Engineering Agile Systems: Creative-Guidance Frameworks for Requirements and Design

Rick Dove
Stevens Institute of Technology
Castle Point on Hudson, Hoboken, NJ 07030
rick.dove@stevens.edu

Abstract

Industrial-led research in the early nineties, searching for new competitive paradigms, launched today's interest in agile systems with a study centered at Lehigh University. Subsequent research went on to identify metrics, modalities, and engineering principles observed in systems that exhibited high degrees of adaptability in the face of unpredictable demands and requirements. This paper reviews that research and its findings as applicable to the engineering of agile systems. The confusion of agile systems with rapid-response systems is addressed, as is the confusion of *agile-systems* engineering with agile *systems-engineering*. Early framework tools for the engineering of agile systems are reviewed, and two additional framework tools are proposed to improve their use as creative-guidance.

Introduction

It is evident when speaking with people about *agile systems engineering* that some hear a hyphenated *agile-systems* (Fricke et al. 2005) while others hear a hyphenated *systems-engineering* (Beck et al. 2001, Hari et al. 2005, Boehm 2006). The meanings are very different as discussed by (Haberfellner et al. 2005). It is also evident that agility is too often equated with speed, and not much else. These confusions need cleared up, especially now that agile systems engineering (un-hyphenated) has become a topic of interest in systems engineering curricula. Stevens Institute of Technology, for one, offers a graduate certificate in *Agile Systems Engineering and Design* in their Systems Development and Operational Effectiveness (SDOE) professional education program.

This paper will attempt to clear up the confusion by reviewing the origins and purpose of agile-systems interest that developed in the early nineties, and the subsequent research findings of the Agility Forum, and then advance that foundation further with lessons learned while applying that knowledge to the engineering of agile systems and the education of would be agile-systems engineers.

Background. Agility as a system characteristic was first introduced in the widely circulated final report of the 1991 study at Lehigh University (Nagel et al. 1991), which focused on enterprise competitiveness. This industry collaborative project was motivated by Toyota's *lean* manufacturing initiative as exposed by (Womack et al. 1990), which put US manufacturers at a competitive disadvantage. Lean codified new system design and operational principles for manufacturing that delivered advantageous values. While US manufacturers struggled to understand and adopt these foreign concepts, the Lehigh study was funded by the US

Department of Defence to identify what would next define global competition, and to build the road map for a US head start.

The study concluded that the frequency and variety of technology and market change was accelerating, and already surpassing the abilities of organizations to adapt. Viable enterprise would need systems that could respond effectively to unpredictable requirements on shorter and shorter notice. Among other deliverables, the study produced a 15-year-out (2005) projection of life in three different industries, depicting the nature of agile enterprise in operation. The problem was defined and the objective was envisioned with that study, but no engineering clues were offered for designing and operating these magical agile systems.

Lean focused on manufacturing systems. Agility focused on enterprise systems, with manufacturing systems included as a subset. Lean system principles were aimed at minimizing operational costs by removing all unnecessary sources of cost, with buffer inventory a prime target. Agile systems, in contrast, were aimed at maximizing response options should something unpredictable occur. In a strict sense, early lean concepts produced a fragile system: one highly dependent on just-in-time raw material and work-in-process arrival. Agile systems, in contrast, pursued a just-in-case path, raising early concerns about the affordability of such insurance.

Lean concepts had been evolving at Toyota for over fifteen years at that time, and were expressible as system design and operating principles. Agile systems, on the other hand, were no more than a compelling objective. In 1992 the Agility Forum was created to solve this lack of systems knowledge. The Forum was set up at Lehigh University under industry leadership with contributed and volunteer help, and was chartered to facilitate a collaborative knowledge development activity within industry. DARPA (ARPA at that time) came in subsequently with significant funding. At its high point the Forum involved over 1,000 participants from over 250 organizations, and generated a solid body of agile systems research and knowledge. This was basically accomplished by active practitioners in industry with pressing problems in search of agile solutions. Forum staff led development of a research agenda, recruited and organized the working groups, and facilitated this grass-roots research.

The Forum dissolved in 1998, mission completed. The interest in agility and agile systems today is a self-perpetuating result of that work in the nineties. Largely the process was successful because it employed real people with real problems in search of real solutions. The Forum's focus on collaborating practitioners, however, ignored traditional academic methods and publication channels. Much of what was developed as a knowledge base is unknown to many in the academic community today as a result. With systems engineering ascending as an academic pursuit, this body of knowledge offers a head start for advancing agile-systems engineering disciplines and identifying further research.

Agile Competitors and Virtual Organizations (Goldman et al, 1995) and *Cooperate to Compete* (Preiss et al. 1996) introduced much of the early Forum thought on agile enterprise strategy and vision. *The Agile Virtual Enterprise* (Goranson, 1999) and *Response Ability* (Dove 2001) came along later once system principles and metric understandings emerged. Other influential contributions to agile enterprise concepts include Clock Speed (Fine 1998), and Adaptive Enterprise (Haeckel 1999).

This author was co-principal investigator on the original Lehigh study, subsequently led the development and management of the Agility Forum's research agenda and collaborative research processes, and has continued independent research and application since. A brief review of the key systems engineering concepts that came from the Forum's work is in order before discussing what the author believes is needed next in knowledge development, and proposing some preliminary ideas that have emerged from recent applications experience.

Reviewing Prior Work

This review focuses on that originating foundation work of the nineties, and attempts to display the key points with brevity, as they are expounded on elsewhere. The reader looking for more detail is directed, with reference, to source material readily available in the Internet and covered extensively in (Dove 2001b). A comprehensive review of subsequent developments is beyond the scope and page limitations of this paper, but the interested reader is referred with respect to (Beck et al. 2001, Haberfellner et al. 2005, Hari et al. 2005, Boehm 2006 and Fricke et al. 2005).

The Problem. The study in 1991 observed that the pace of change in technologies and markets had suddenly gotten ahead of the typical enterprise's ability to keep up. It wasn't that an enterprise didn't know what it should do next (some didn't, of course), but rather an inability to respond effectively to what it knew was needed. Enterprise systems (with culture included) could not be re-engineered to accommodate new requirements fast enough. Respected established companies were loosing market share and going out of business, while new companies designing new systems for the new requirements were thriving...or so it seemed, until the next change hit these newer companies that had equally intractable systems.

Flexible systems, those that could respond in multiple pre-planned ways, were generally inadequate. The nature of future requirements could not be sufficiently predicted to have a pre-planned response ready to go. A new approach was needed that could somehow extend the useful life of existing systems, re-tasking them rather than replacing them, reconfiguring them rather than redesigning them.

Though much of this paper will draw from the domain of enterprise systems, it is at core domain agnostic. The life cycle of requirements for sustainable systems-viability is an issue in virtually all systems domains, for reasons that include increasing systems complexity, increasing customer expectations and impatience, increasing knowledge of the possible, and increasing advances in technology.

The Search. Sufficient urgency existed to demand a quick and actionable answer to the problem. Hundreds of participants from all varieties of organizations joined in a search activity. They were asked to look in their own organization, or others they had knowledge of, for systems (loosely defined) of any kind that responded effectively to unpredictable requirements, with some consistency (Benson et al. 1995 and Dove 1995). Over the course of six years hundreds of these systems were examined in collaborative on-site structured-workshops for common enabling and defining characteristics. Three principal questions were seeking answers:

- What metrics define *effective* response?
- What kinds of dynamics must be accommodated by *effective* response?
- What common system-design principles facilitate *effective* response?

Metrics. Looking for *effective* response was not how it started. Initially it was thought that the key lay in fast response - fast enough to thwart the threats to continued viability. This almost panic sense of urgency was myopically defining agility as *fast reaction*. In time neither *fast* or *reaction* proved sufficient. On the metric side the search for fast response evolved into a multi-dimensional metric for *effective* response when it became clear that the cost of response was equally important in an environment that demanded a new response with increasing frequency. Break the bank on the first response-at-any-cost and it's game over. Once that was understood it was easy to recognize that both quality of response and the scope of capable response were also necessary metrics.

Quality addresses the predictability of response capability: can it be done on time, on budget, and on spec...repeatably. Scope addresses the range of capable response, and is the principle distinguishing factor from a system that is simply flexible. Scope is bounded by mission, so that an effective response is one that can address new requirements predictably, affordably, and timely, anywhere within the mission space. An agile systems-engineering process, for instance, does not have to address projects outside of the organization's mission space; but it should never flub or have to decline a desirable opportunity within the mission space, for lack of capability. Table 1 summarizes these four metrics.

Table 1: Agility is effective response to opportunity and problem, within mission ... always

An *effective* response is one that is:
▪ Timely - fast enough to deliver value
▪ Affordable - at a cost that leaves room for an ROI and another response
▪ Predictable - can be counted on to meet all expectations
▪ Comprehensive - can satisfy anything and everything within mission boundary

Dynamics. The quest for agility was initially driven by the inability to react effectively when an unanticipated response was required. It didn't take long to realize that reactive response competency was a defensive posture, however, and that proactive response competency had an equally necessary role. Reactive response capability enables sustainable viability. Proactive response capability enables innovation. This two-category realization immediately helped clarify the nature of response proficiency in the systems being examined, and opened the door for a finer grain demarcation. In the end four sub-categories of proactive response and four sub-categories of reactive response were defined to channel system requirements analysis.

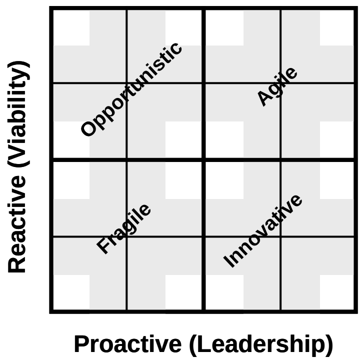

Figure 1: Response Proficiency Space

Table 2 summarizes the response categories of the requirements analysis framework. There is nothing absolute about the words chosen as labels. This framework is simply a tool to help spread thought in a variety of different directions for requirements analysis and development. The framework evolved over years of employment and adjustment in systems analysis and has proven to be effective. Greater detail on metrics, response categories, and the nature of their employment can be found in (Dove 1999 and Dove 2001).

Table 2 - Response Dynamics Categories

Proactive Dynamics	Reactive Dynamics
Creation/Elimination - Create something new or eliminate something that exists.	**Correction** - Rectify a dysfunction.
Improvement - Incremental improvement, generally a continual open-ended activity.	**Variation** - Real-time operating change within the mission.
Migration - Anticipated fundamental change, often with infrastructure.	**Expansion/Contraction** - Increase or decrease of existing capacity.
Modification - Addition or subtraction of unique capability.	**Reconfiguration** - Reorganize resource or process relationships.

Design Principles. Metrics define the objective of agility: *effective response*. If it walks like a duck, quacks like a duck, and so forth, it is a duck. There are surely many ways to design systems that meet the objective. The conclusions of the research in the nineties, however, provides a specific set of ten design principles. These ten principles were observed repeatedly to some extent or another in all of the "highly adaptable" systems that were analyzed, and were believed responsible for that adaptability. For compatible but somewhat different conclusions see (Fricke et al. 2005).

What analysis found was fairly consistent:

- One general strategy:
  1. Reusable modules, reconfigurable in a scalable framework
- Two general concepts:
  1. Scalable framework
  2. Pool of modules
- Three general capabilities:
  1. Assembly of new system configurations from existing modules
  2. Augmentation of module pool with new module types or versions
  3. Evolution of framework to accommodate new requirements
- Ten general design principles

  1-10 See Table 3

Table 3: Agile-System Design Principles

Evolving Standards (Framework) - Frameworks standardize inter-module communication and interaction; define module compatibility; and are monitored/updated to accommodate old, current, and new modules.
Self-Contained Units (Modules) - Modules are encapsulated, distinct, separable, self-sufficient units cooperating toward a shared common purpose.
Plug Compatibility - Modules share defined interaction and interface standards; and are easily inserted or removed.
Facilitated Reuse - Modules are reusable/replicable; and responsibilities for ready re-use/replication and for management, maintenance, and upgrade of component inventory is specifically designated.
Redundancy and Diversity - Duplicate modules are employed to provide capacity right-sizing options and fail-soft tolerance; and diversity among similar modules employing different methods is exploited.
Elastic Capacity - Module populations may be increased and decreased widely within the existing framework.
Flat Interaction - Modules communicate directly on a peer-to-peer relationship; and parallel rather than sequential relationships are favored.
Deferred Commitment - Module relationships are transient when possible; decisions and fixed bindings are postponed until immediately necessary; and relationships are scheduled and bound in real-time.
Distributed Control and Information - Modules are directed by objective rather than method; decisions are made at point of maximum knowledge; information is associated locally, accessible globally, and freely disseminated.
Self-Organization - Module relationships are self-determined; and component interaction is self-adjusting or negotiated.

High Concept. In 1997, well down the systems analysis and principles-extraction path, a three-day workshop stumbled upon a key revelation at a unique low-volume, high-variety, auto-body, after-year GM stamping plant. Unlike most auto-body stamping plants servicing a few current year models in high volume production, this plant inherited the after-model-year responsibility for hundreds of car models. They had to service orders as small as a mere 100 parts. Auto-body parts are not simply stamped metal, but must be assembled with welds, glue, fabric, insulation, mechanisms for windows and other such to make a finished replacement part. This is usually done on dedicated highly-automated 24/7 assembly lines - impractical for high-variety small-lot orders. This plant was a wealth of unique agile-systems concepts as a result (Dove 2001).

The revelation came when their assembly line methods were analyzed (Dove 1997). In the usual sense, they didn't have an assembly line. What they had instead was a way to construct just-in-time a custom assembly line for any of the thousands of assembled parts they might have to build. They did this with a highly disciplined management and configuration process for the various tools and work surfaces that could be quickly configured into a new assembly line. They were masters at managing, configuring, and updating a sizable pool of modules, assembly-system assembly-instructions, and a framework of assembly areas with a standardized utility infrastructure. Tool modules, such as control units and automated machines for bending and shaping metal, were inherited from a foreign world, and reworked for standardized interfaces before joining the module pool. This was an ongoing process as the plant inherited new model responsibilities multiple times per year. A new order saw a new assembly-system constructed on the spot, then disassembled and replaced by another in the virtual twinkling of an eye. Module inventory was constantly updated to accommodate new assembly responsibilities.

The revelation was that this just-in-time assembly-line process were a good metaphor for agile systems.

Switch gears to imagine an agile systems-engineering process (note hyphen placement) modelled on this metaphor. New projects require a team of competent resources with various types of specialty skills, a knowledge base of prior work that might have reusable application, tools and services of various types, facilities appropriate for development and testing, outside contractors with appropriate capabilities, and so on. All could be consider resource modules that must be immediately available and ready to work productively together when needed. These modules will need plug-compatible interfaces for all other modules and infrastructure connections they are likely to interact with. Somewhere this inventory of modules is managed to make sure it has capacity, capability, and readiness at all times. And somewhere sits the system assembler that constructs the perfect system-engineering system on demand. Carry on down that path and it will become clear that agile systems-engineering is much more than faster spirals and better requirements.

Maturity. Another interesting pattern emerged: competency appears to progress in a common sequential pattern, as shown in Figure 2. This is especially evident among the four metrics. Speed is generally the first focus for system response. Once that is sufficient, response cost becomes the issue of contention, followed by quality and eventually scope. A similar progression has been observed among the proactive and reactive response categories.

This maturity progression was used to generate an agile enterprise reference model by evaluating the agile state of 24 business practices at Remmele Engineering (Dove at al. 1996). No claim is made that higher maturity scores equate with better performance, anymore than higher maturity among teenagers equates to higher intelligence.

The analysis research exhibited this same dawning or awareness, looking simply for fast-

Maturity	Working	Metric	Response Competencies	
Stage	Knowledge	Focus	Proactive	Reactive
0 Accidental	Examples	Pass/Fail	None	None
1 Repeatable	Concepts	Time	Creation	Correction
2 Defined	Metrics	Cost	Improvement	Variation
3 Managed	Rules	Quality	Migration	Expansion
4 Mastered	Principles	Scope	Modification	Reconfiguration

Figure 2: Common Maturity Progression for Response-Proficiency Development

response systems initially. What becomes evident with experience is that sustainable effective response speed is a product of agility, not the other way round. A sustainable system is fast because it is agile, it is not agile because it is fast.

Efficacy. The response dynamics categories combined with the response metrics formed an effective Response Proficiency Framework tool that was used throughout the research to analyse the values delivered or sought by existing systems. The Design Principles Framework also proved its efficacy as an analysis tool, illuminating the enabling mechanisms of agility. Both frameworks began as sparsely populated hypotheses that gradually became richer and more useful as analysis work confirmed their usefulness and refined their nature.

The Design Principles Framework was the later of the two to mature. This occurred in a series of 3-day on-site workshops during 1997 (Dove 1998) with a group of common participants from a variety of companies - all were interested in vetting or adjusting these frameworks for use as creative guidance tools for design synthesis. The workshops were structured to first analyze two on-site systems that exhibited agile characterises to establish the expression of framework concepts clearly in mind. Then a candidate for design synthesis was presented by the hosting organization - either as a problem system that needed to become more agile or as a soon-to-be developed system that had to be agile.

The group employed the Response Proficiency Framework as a requirements development tool, using it as a means to identify the range and nature of the dynamics the candidate system would have to cope with. Then the Design Principles Framework was employed to guide the development of design strategy. This synthesis exercise generally occurred on the final day of the three day workshop, and often had the camel-committee flavour at days end. Nevertheless, two values emerged: for the host, structured brainstorming provided relevant actionable ideas for later consideration by responsible system engineers; and for the participants (those that came to three or more of the workshops) a new systems insight generally developed - it was like a pair of x-ray glasses became a permanent fixture, revealing at a glance a system's architectural bones that facilitated agility and the broken bones that impeded it.

Exercised with a little repetition, these two frameworks as analysis tools have developed knowledge at the level of insight (Dove 1998). The world isn't seen the same way after. As synthesis tools, however, they are not as easily employed by everybody. The frameworks are generalized abstractions of concepts that can be recognized easily when seen in an existing system. But the very generalization and abstraction that makes them universally applicable in analysis can also distances them from ready expression in design strategy. This is not dissimilar from trying to teach art appreciation as a means to create an artist - a useful and necessary early step, but not enough for many.

Bridging Frameworks

Two framework tools that bridge the domain-agnostic proficiency and design frameworks to a domain-specific application are proposed here. At this time it is anticipated that a sponsored series of workshops focused on agile cyber-security in the electric utility sector will begin vetting these proposed tools in 2006 .

The Reality Factors Framework bridges the Response Proficiency Framework into requirements development. The Design Strategy Framework bridges the Design Principles Framework into security-system design. These bridge frameworks are domain specific, but once constructed they can be reused for new system engineering projects within the same domain.

Reality Factors Framework. This framework categorizes uncontrollable and unpredictable factors in the system's environment that should be mitigated or nullified by the system design. Framework categories are those recognized as sources of dynamic problems in the real environment, as opposed to an idealized or well-behaved environment. Human factors and Murphy's law are good guiding principles to drive the formulation of this framework.

The seven categories shown in Table 4 were originally developed for analyzing security strategy requirements (Dove 2005), and are generalized here to be usable in virtually all business process domains. A similar concern with the mismatch between reality and general security strategy was the subject of (Workshop 2003).

Technology Pace - Issues of decreasing technology life cycles and increased technology variety.
Systems Complexity - Issues of systems predictability, unintended consequences, and interconnection.
Agile Enterprise - Issues of agile enterprise operation and strategy caused by change and response.
Globalization - Issues of interaction with other-cultural and other-national business organizations and personnel.
Human Behavior - Issues of individuals acting unpredictably, inconsistently, and irrationally.
Organizational Behavior - Issues of organizations acting unpredictably, inconsistently, and irrationally.
Threat Sources - Issues of unpredictable or uncertain acts of aggression.

Table 4: Reality Factors Framework - Domain Specific to Business Process Systems

When this framework is employed it should be expanded by the person(s) doing the requirements development task in order to guide the development process and the stakeholder interviews and probes. An example of this expansion is beyond the scope of this paper, but can be found online at (Dove 2004) applied to cyber-security in the electric utility industry. In usage, each of the Reality Factors is individually investigated for each of the Response Proficiency areas.

Design Strategy Framework. This is a strategy-dependent framework, with categories that are specific to the type of system under consideration. The framework shown in Table 5 deals with categories specific to an agile security strategy-system, and was first utilized in the development of a proprietary security strategy (document unavailable) for a semi-conductor foundry in Malaysia in 2002.

Proactive Domains	Reactive Domains
Vulnerability/Risk Anticipation – Identify pending changes in vulnerability and risk before occurrence	**Detection** – Detect intrusion and damage quickly.
Prudence – Correct vulnerabilities before exploitation, sense indirect indicators of pending exploitation	**Containment** – Minimize potential damage scope.
Transformation – Change randomly the elements/nature of security system.	**Mitigation** – Minimize potential damage magnitude.
Threat Anticipation – Identify and counter threats and risks before exploitation.	**Assessment** – Understand what has been damaged and how.
Migration – Continuous upgrade of security strategy and components.	**Recovery** – Repair damage quickly.
Accountability (Proactive) – Identify perpetrators with traps, glass houses, disinformation, etc, before damage.	**Accountability (Reactive)** – Identify the perpetrators forensically, after damage.

Table 5: Strategy Design Framework - Domain-Specific to Agile Cyber Security Systems

The only generalization for this framework is that it has both reactive and proactive branches to ensure that it is structured to address both reactive and proactive requirements issues. A strategy framework does not necessarily need to have an equal number of subcategories under each - though experience has shown that a mirror-image effect typically evolves the result to an equal number.

Similar to the process for requirements development, this bridge framework is taken one category at a time as all of the design principles are considered for application.

Conclusion

Research into highly adaptable system has shown that their values include speed of response, but that they rely on a sustainable ability to deliver this speed by also delivering values in cost of response, quality of response and scope of response. An example of an agile *systems-engineering* process as an instance of *agile-systems* engineering was outlined as a structural pursuit that could be fast because it was agile, rather than being agile because it was fast.

Tools exist for guiding the creative engineering of agile systems. Early generic ones that are good for building engineering mind sets on structure and principles were reviewed, and two new tools were discussed that show potential to bridge the gap between generic concepts and domain specific application. These new frameworks are recent developments that will be refined with a few years use, as the original two frameworks were.

References

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Andrew, H., Jeffries, R., Kern, J., Marick, B., Martin, R., Mellor, S., Schwaber, K., Sutherland, J.and Thomas, D., " Manifesto for Agile Software Development," The Agile Alliance, 2001, www.agilemanifesto.org.

Benson, S., Dove, R., Drake, W., Fiore, T., Goldman, D., Goranson, T., Hartman, S., Parunak, V., Scaringella, S., Seshadri, Raja., and Turner, B., "Agile Practice Reference Base ", Agility Forum, Bethlehem, PA, May 1995.

Boehm, Barry, "Some Future Trends and Implications for Software Engineering Processes," Systems Engineering, Vol. 9. No. 1, Wiley Periodicals, Spring 2006.

Dove, R., "Best Agile Practice Reference Base - 1994: Challenge Models and Benchmarks", Proc. 4th Annual Agility Conference, Agility Forum, Bethlehem, PA, Mar 1995, www.parshift.com/Files/PsiDocs/Rkd5Art1.zip

Dove, R., Hartman, S., and Benson, S., "An Agile Enterprise Reference Model, With a Case Study of Remmele Engineering," Agility Forum, Bethlehem, PA, 1996, www.parshift.com/docs/aermodA0.htm.

Dove, R., "Assembly Lines Built Just In Time," *Automotive Manufacturing and Production*, Gardner Publications, August 97, www.parshift.com/Files/Essays/Essay033.zip.

Dove, R., "Realsearch—A Framework for Knowledge Management and Continuing Education," *Proc. IEEE Aerospace Conference,* Vale, CO, March 1998, www.parshift.com/Files/PsiDocs/RealSrch1.zip.

Dove, R., "Knowledge Management, Response Ability, and the Agile Enterprise," *Journal of Knowledge Management*, Emerald Group Publishing, Bradford, UK, March 1999, pp. 18-35, www.parshift.com/Files/PsiDocs/Rkd9Art1.pdf.

Dove, R., "Design Principles for Highly Adaptable Business Systems, With Tangible Manufacturing Examples," *Maynard's Industrial Handbook*, pps 9.3-9.26, McGraw-Hill, New York 2001, www.parshift.com/Files/PsiDocs/Rkd8Art3.pdf.

Dove, R., *Response Ability—The Language, Structure, and Culture of the Agile Enterprise*, Wiley, New York, 2001, www.parshift.com/ResponseAbility/Preface.htm.

Dove, R., "Enterprise Agility *Is* Risk Management," Daily IssueAlert, UtiliPoint International, 11/19/04, www.parshift.com/Files/Essays/Essay066.pdf.

Dove, R, "Frameworks for Analyzing and Developing Agile Security Strategies - Oriented for the Energy and Utility Sector," Agile Security Forum, 1/22/05, http://www.agilesecurityforum.com/docs/AsfPaperSixFrameworks.pdf

Fricke, E. and Schulz, A.P., "Design for Changeability (DFC): Principles To Enable Changes in Systems Throughout Their Entire Lifecycle," *Systems Engineering*, Vol. 8, No. 4, pps 342-358, Wiley Periodicals, 2005.

Goldman, S.L., Nagel, R.N., and Preiss, K., *Agile Competitors and Virtual Organizations*, Van Nostrand, 1995.

Goranson, H.T., *The Agile Virtual Enterprise - Cases, Metrics, Tools*, Quorum Books, Westport CN, 1999.

Haberfellner, R. and de Weck, Olivier, "Agile SYSTEMS ENGINEERING versus AGILE SYSTEMS Engineering," Proc. International Symposium of the International Council on Systems Engineering (INCOSE), July 2005, www.mit.edu/~deweck/PDF_archive/3%20Refereed%20Conference/3_59_INCOSE-2005-AGSEvsEAGS.pdf.

Haeckel, S., *Adaptive Enterprise*, Harvard Business School Press, Boston, 1999.

Hari, A., Cropley, D.H. and Zonnenshain, A., "Agile Systems Engineering for Creative Anti-Terror Solutions," Proc. System Engineering / Test and Evaluation Conference (SETE), Brisbane, Australia, November 7-9, 2005. www.unisa.edu.au/seec/pubs/05papers/haria.pdf

Nagel, R., Dove, R., Goldman, S., and Preiss, K., *21st Century Manufacturing Strategy: An Industry-Led View*, DIANE Publishing Company, Coolingdale, PA, 1991 (available at Amazon.com or from dove@parshift.com).

Preiss, K., Goldman, S.L., and Nagel, R.N., *Cooperate to Compete*, Van Nostrand Reinhold, New York, 1996.

Womack, J., Jones, D., and Roos, D., *The Machine That Changed The World*, Macmillan, New York, 1990.

Workshop on Scalable Cyber-Security Challenges in Large Scale networks: Deployment Obstacles, Large Scale Networking Coordinating Group of the Interagency Working Group for Information Technology Research and Development, March 13-14, 2003, Lansdowne, VA, cs-www.cs.yale.edu/homes/jf/LSN-report.pdf

Biography

Rick Dove is an Industry Fellow at Stevens Institute of Technology teaching graduate courses in Agile-Systems Engineering. He is Chairman of Paradigm Shift International, a consulting and special projects group specializing in agile enterprise strategies and systems. He has run companies producing software, manufacturing machinery, fine wine, rapid manufacturing services, strategic planning services, and interim executive services. He has led engineering, R&D, IT, information security, sales, and marketing in a variety of companies. He was co-principle investigator of the 1991 project at Lehigh University that gave birth to the concept of Agile Enterprise, and led the subsequent Agility Forum's research and industry involvement activity. He is author of two books and over 175 publications. A full resume and publication list can be found at http://www.parshift.com/Files/PsiDocs/RkdBio.pdf