Agile Methods: Reconciliation with the Classical SE Method and the Integral Role of the Ergonomist

George M. Samaras
Samaras & Associates, Inc.
george@samaras.eng.pro

Abstract

Agile methods have had a profound and beneficial impact on the development of products and processes. This article suggests that they are an enlightened reaction to the misuse and abuse of the half-century old classical systems engineering method, drawing particular attention to XP, RUP, FDD, and Scrum. The role of ergonomists, in promoting human-centered design throughout the life of the project, is described in the context of the classical method.

1. Introduction

I have, for many years, encouraged my clients to utilize lightweight or agile methods in their (hardware, software, ergonomic, and production) development of medical devices, pharmaceutical equipment, or clinical information systems. I have consistently encountered two principal reactions: (a) "never heard of it" and (b) "not approved by management". The first response, from my perspective, is welcome in that it justifies the presence of a consultant; the second is frustrating, in that I have always viewed all the various methods used in product and process development as subsets of the classical systems engineering method. To my way of thinking, you adjust the methods to the project, NOT the project to the methods. Just as enlightened managers adopt an agile management style, enlightened developers adopt an agile development style. It has been long recognized by contingency theorists that organizations in an uncertain environment should maximize flexibility with decentralized authority structures and highly developed lines of communications [see, e.g., 1].

The first objective of this article is to suggest that "agile" methods arise as a rational reaction (by intelligent individuals actually tasked with developing a product or process) to the misuse and abuse of the classical systems engineering method. An initial sense that this may be a reasonable interpretation can be obtained by the following restatement of the agile manifesto [2]:

- Discovering the real users' actual needs, wants and desires (i.e., human-centered design) has more value than a static contract document intended only to initiate a relationship with a customer. (*Customer collaboration over contract negotiation*)
- Real tools that satisfy the real users' real needs in an intuitive manner and without steep, resource-intensive, learning curves have greater value than volumes of rapidly obsolescing documentation. (*Working software over comprehensive documentation*)
- Every project (that is not a mere identical duplication effort) involves learning – modifying your behavior responding to changes – and this can never occur in a rigid, pre-planned, working environment. (*Responding to change over following a plan*)
- All work occurs in a socio-technical system and the management (read "leadership") of humans and their social interactions is far more complex that the management of the requisite processes and tools. (*Individuals and interactions over processes and tools*)

The second objective of this article is to indicate that the classical method provides an excellent framework from which to achieve human-centered product design, development, production, operation, and disposal.

2. Background

We have recently discussed the classical systems engineering (SE) method, shown that it is a superset of a number of (non-agile) methods proposed over the past two decades, and that it provides a framework for incorporating human factors (ergonomics) knowledge and integrating ergonomists throughout the interdisciplinary development lifecycle of products and processes [3]. The methods previously considered were those of Gould & Lewis (1985) [4], Mantei & Teorei (1988) [5], Nielsen (1992) [6], Kreitzberg (1996) [7], Mayhew (1999) [8], and Endsley (2002) [9]; in all cases, they had an emphasis on human-centered design. This article extends the methodological analysis (study of methods [10]) to agile methods. I begin with a brief review of our previous exposition and then examine how various agile methods relate to the classical SE method.

2.1 The Classical SE Method

SE is a structured, systematic approach to system risk reduction (business-technical-social or cost-schedule-scope-quality) over the full lifetime of the system. Your ability to predict system behavior reliably increases with increasing levels of validation. Un-validated systems have a high degree of uncertainty (complexity) in their behavior; ***validation thus decreases the complexity of system behavior***. SE is a proactive hazard mitigation process, maximizing the likelihood of reducing errors and time to market. It is a structured, ***risk***-based, iterative approach to the conceptualization, research, design, development, test & evaluation (RDDT&E), deployment/operation, and salvage/disposal of products and processes. It is a process that emphasizes transparency and clarity of known objectives and constraints. The SE domain consists of the triumvirate of requirements engineering, compliance engineering, and reliability engineering.

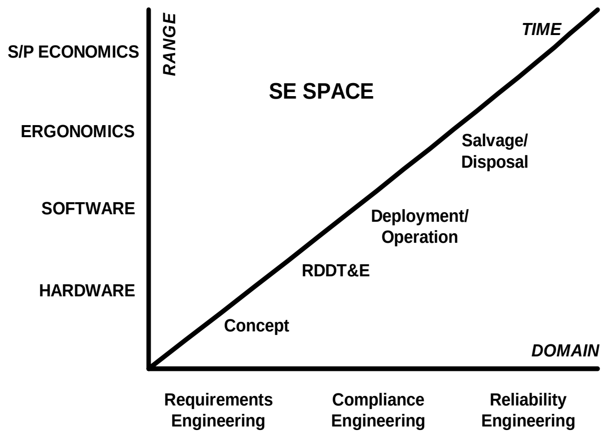

Figure 1: The Microergonomic SE Space

This is the domain of activities regardless of whether we are dealing with product (microergonomic) or organizational (macroergonomic) activities. The microergonomic SE space (Figure 1) includes the activities of hardware engineering, software engineering, human factors engineering, and seller/purchaser economics. The timeline extends from conceptualization to salvage and disposal (lust to dust). I am unaware of any method that cannot be cast in this space. Even software projects that presume to run on absolutely standard hardware and presume to have no human users, still operate within this space.

The SE method is generally represented as a lifecycle as shown in Figure 2. While any putative controversy relating to the microergonomic space (Figure 1) can quickly be dispelled, the misuse and abuse of the SE method arises when we consider the lifecycle in shorthand (condensed) notation.

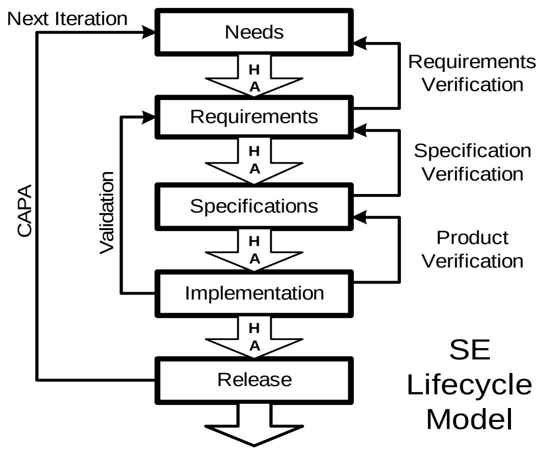

Figure 2: The Condensed SE Lifecycle

This condensed representation of the SE lifecycle does NOT state that ALL the needs must be assessed, then ALL the requirements must be established, then ALL the specifications determined, and then all the implementation work must be accomplished. This is a rigid, mistaken and uninformed reading of the shorthand notation. Real developers with real experience immediately recognize this for the nonsensical interpretation that it is. In the next section we will see how the expanded notation begins to become familiar to agile method practitioners.

The SE lifecycle begins with the initial conceptualization of the system, is continually applied throughout the research, design, development, test and evaluation phases, in the operational phase (with periodic re-validations) and, finally, when the system is obsolete, in the salvage and disposal phase. The lifecycle is characterized by a *feedforward* loop consisting of needs assessments, translation of needs to quantifiable requirements, translations of requirements to quantitative specifications, translations of specifications to a product/process implementation, and deployment (internally or externally) of the product or process. The *feedback* loops consist of validation testing (implementation vs. requirements), verification testing (of requirements, specifications and implementations), incremental hazard analyses (HA), and post-deployment corrective and preventative actions (CAPA). Ergonomic considerations exist throughout the full lifecycle and

© GM Samaras, 2005, All Rights Reserved

ergonomists are able to add value at every step in the process (see section 4).

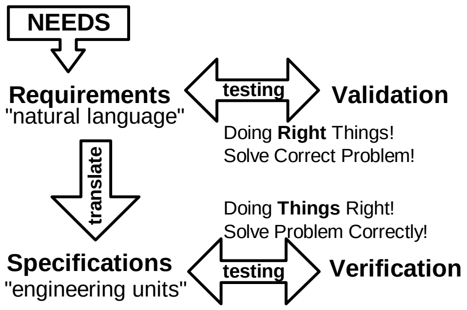

Figure 3: Verification vs. Validation

Another source of misuse and abuse of the SE method centers on the requisite testing. While there has been some legitimate disagreement on when to use which "V" word [11], the underlying meanings remain the same (see Figure 3). The purpose of measurement is to support "management", not "The Management". The text box below elucidates the relationship between management (what each and every developer must do for themselves) and how the process begins with recognition – that is one of the fundamental benefits of using a structured, systematic (nobody said rigid or invariant), and transparent approach. It is worthwhile noting that "*operationally define*" means that you are specifying the test and measurement procedure (known in extreme programming (XP) as "test-first development" [12]). This is traditionally the greatest stumbling block in any management process. Once you can operationally define something, you know a great deal about it.

- One cannot **manage** what one cannot **control**
- One cannot **control** what one cannot **measure**
- One cannot **measure** what one cannot **operationally define**
- One cannot **define** what one does not **know about**.

Table 1

Finally, the requisite amount of documentation (the means of communicating across time and place) is always an important issue in every effort. Documentation may not be synonymous with understanding, but very little understanding occurs and persists without adequate documentation. It cannot be rigidly pre-determined or externally planned as it must vary with a project's evolving characteristics. We have previously argued that the degree of formalization (Figure 4) may vary from 3"x5" cards to massive databases, depending upon the intersection of the **product *criticality*** (e.g., patient safety is at stake or their may be toxic impacts on the environment) and **project *complexity*** attributes (e.g., as dictated by project size, time constraints, team skills, experience, & distribution, as well as organizational readiness, maturity, & culture) [13]. Cockburn [14] describes the criticality dimension as ranging from loss of comfort, to loss of monies (discretionary, then essential), to loss of life.

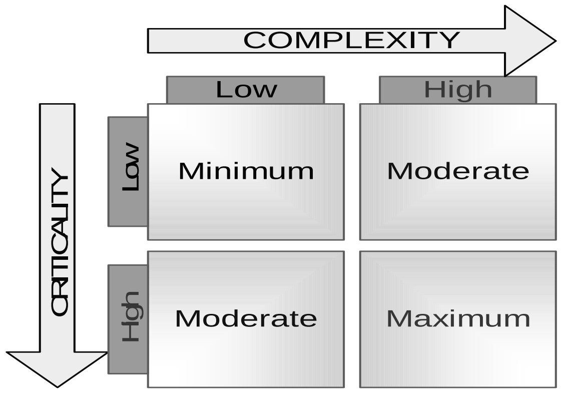

Figure 4: Formalization Position Diagram

Project complexity is related to problem complexity. Therefore, iteratively decomposing (also known as partitioning or elaborating) your problem offers the potential to reduce complexity, but only up to a point, after which further decomposition results in increases in project complexity due to the increased burdens associated with cooperation and coordination. Cockburn [15] describes essentially the same relationship in terms of "problem size versus methodology weight". Therefore, the correct answer to the old adage "How do you eat an elephant?" is "In appropriately-sized bites!" The objective should be to achieve the minimum degree of formalization that is necessary and sufficient for the particular project. This permits the team to concentrate on human-centered design.

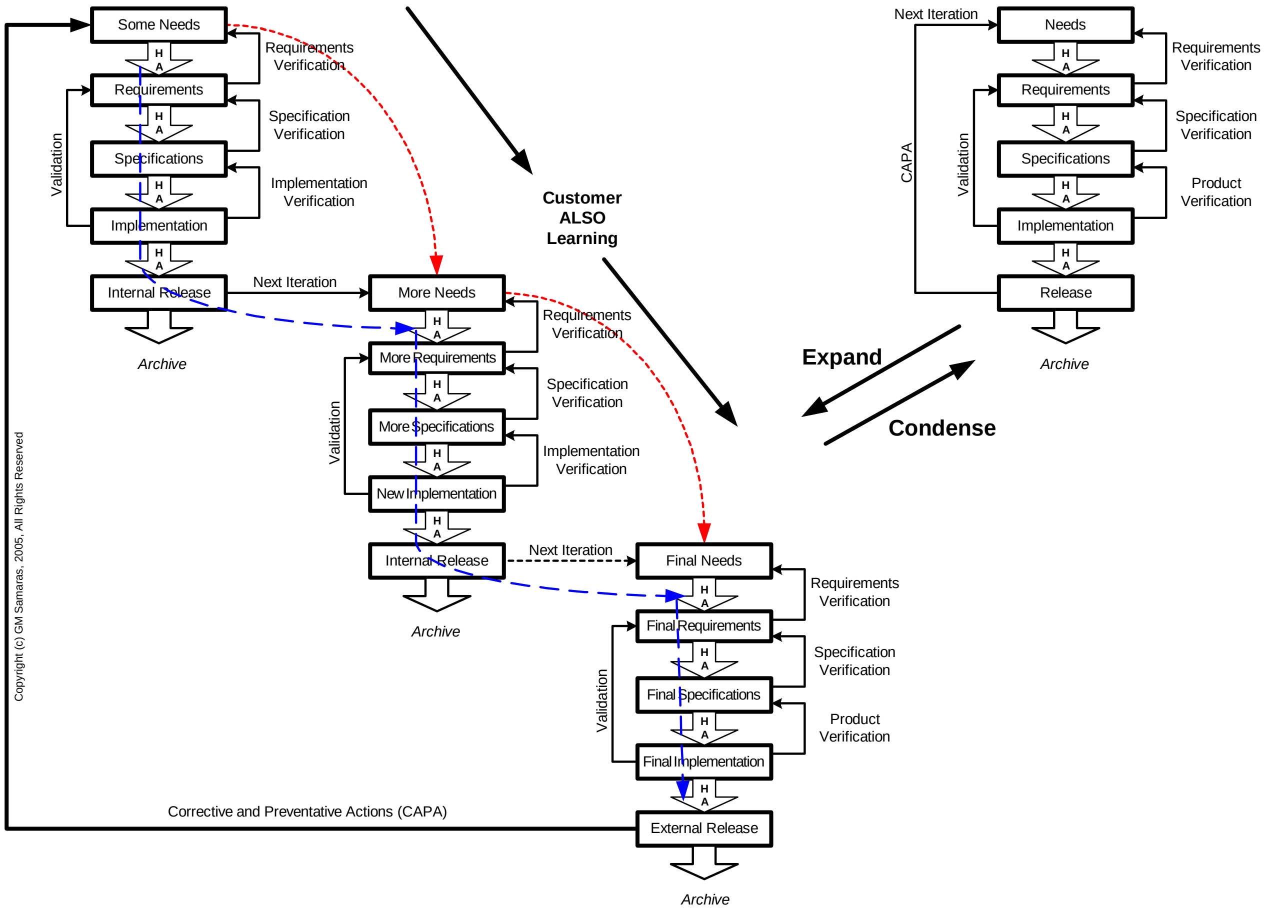

Figure 5: The Expanded SE Lifecycle

3. Expanding the SE Lifecycle

The shorthand SE lifecycle can be expanded as shown in Figure 5, clearly demonstrating the non-serial nature of the method. The expanded form elucidates the iterative nature of the SE method, showing the hierarchy of nested iterations. It illustrates the adaptive nature of the method, making explicit the unstable nature of learning on the part of the users, the team members, and the environment in which both operate. It is worthwhile noting that the CAPA loop in the SE method offers opportunities not only to improve the product, but also to learn how to improve your development processes.

Practitioners of XP will recognize the emphasis on testing (verifications, validation, and hazard analyses) as well as the constraint on focusing on the current iteration (current planning horizon) and internal release of a working implementation, without regard for future iterations. The well-known "pair-programming" paradigm that can achieve near simultaneous "code and code-walkthrough" is part, though not all, of the implementation verification feedback loop. The SE method does not embrace change, it presumes change. XP's four basic activities "coding, testing, listening, and designing" are captured as follows:

XP	SE
Coding	Implementation
Testing	Imp. Verification
Testing	Validation
Listening	Requirements
Testing	Reqs. Verification
Designing	Specifications
Testing	Specs. Verification

Table 2

We can also visualize the SE method modeled as a discrete event process (Figure 6) in a manner similar to the representation of the Rational Unified Process (RUP) [16]. As pointed out by Fowler [17], RUP is a *process framework* and can accommodate a variety of processes, from agile to not-so-agile. In a sense, this same criticism can also be leveled at the SE method – when viewed in shorthand notation. The expanded notation of the lifecycle provides a considerably more detailed "recipe" that may be readily followed and adapted to individual needs.

Needs assessments, requirements formulation and verification occur in the conceptualization phase and the salvage/disposal phase, as well as the main development phase. Practitioners of "Feature-Driven Development" may recognize the conceptualization phase as the time when the overall model is developed, the initial list of "features" is formulated and utilized for project planning purposes. The next phase contains the multiple iterations for "Design by Feature" and "Build by Feature".

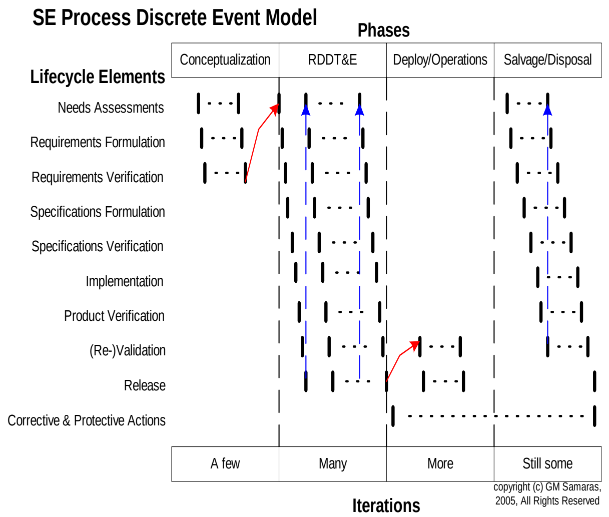

Figure 6: SE Process Discrete Event Model

Scrum [18] practitioners will recognize the evolution of requirements (derived from the evolving needs) as the "Product Backlog". An iteration that delivers an internal release (it must be functional or it will not achieve the "release" stage) may be viewed as a "Sprint". The "Sprint Backlog" is recognized as the new increment of requirements that need implementation within the sprint iteration.

The purpose of this section is not to PROVE that various agile methods are derived from the SE method, but rather to suggest that they are enlightened reactions to the misuse and abuse of the classical method, when it is applied in a manner that ceased to be adaptive and human-centric (both for the customer and the developer). Agile methods are not only useful to the software community; they are also valuable to ergonomists, who in turn may contribute to the development of human-centric software products as envisioned in the classical SE method.

4. The Role of the Ergonomist

If someone hands you a set of project requirements without justifying to you and your team members how they determined the universe of users, sampled the user populations, and assessed the user's needs, they are NOT your friend! These days, the principle locus of failure in product or process development is a lack of human-centered design. No matter how talented and experienced the project team, if you do not meet the users' evolving needs, your efforts will not be properly appreciated. Hitting this moving target can be facilitated by the continuous involvement of ergonomists having "an equal seat at the table".

4.1 Needs Assessments

User needs assessment is a complex activity. It has often been implemented by marketing personnel with *ad hoc* technical support. Common flaws include not selecting the proper target audiences and assuming you already know the user needs. A complete and correct user needs assessment presupposes that you have correctly identified and properly sampled the universe of user populations. It is a central area of expertise and practice in ergonomics. Some examples of needs assessment techniques include:

a Priori Elicitations
- Interviews
- Questionnaires
- Ethno-methodological studies
- Brainstorming
- Problem-domain storyboarding
- Literature reviews
- Ergonomics research

a Posteriori Elicitations
- Prototyping
- Evolutionary (rapid & iterative) development techniques

The identification of compliance and reliability needs may also be developed. Both from a good business practices perspective and from a FDA regulatory perspective, they must be implemented in a statistically valid manner, so that the results truly represent the populations under study.

4.2 Requirements

Defective requirements are the principal cause of incorrect or inadequate systems and failed validations. Properly formulated requirements are natural language statements (e.g., English) that are understandable by the user populations, including the project team and seller and purchaser management. Properly formulated requirements must be traceable to specific user needs, must be clear, complete, and internally consistent, and must be verifiable (you must be able to design a test for it). In order for a requirement to be quantifiable and testable – and thus verifiable – it is imperative that operational definitions of the critical elements incorporated within each requirement exist. Absent operational definitions, there can be no measurements and no verification.

Proper requirement formulation is an inter-disciplinary activity that necessarily includes ergonomics expertise to properly represent the discovered needs of the various user populations. A central activity of ergonomics is properly translating user needs into requirements (and then requirements into specifications). And, when difficult engineering trade-offs are encountered, the ergonomists on the design team must ensure that the user's needs are properly considered - because if they are not, the project team's efforts will fail!

4.3 Specifications

Design specifications are the true basis for the product design and are quantitative product attributes. Once again, the ergonomist can play a crucial role on the project team, directly impacting the work of the rest of the team and the final design of the product:

1. *hardware ergonomics* - the ergonomist not only has access to tabulated human cognitive and perceptual data, and as appropriate, anthropometric data, which can dictate physical specifications, but the ergonomist is trained to properly use these data in the realization of engineering designs.

2. *software ergonomics* - the ergonomist is trained to participate in the design of user interfaces, to conduct task analyses on the proposed logical operation of the product, and to participate in the design of training, operation, and maintenance materials.

3. *environmental ergonomics* - the ergonomist can assist the design team in assessing how known workspace environmental modalities can impact the use and reliability of the proposed design (e.g., effects of temperature, humidity, lighting, ambient noise, and air quality on user fatigue, perceptual, and cognitive abilities).

4. *macro-ergonomics* - some ergonomists can assist the organization in harmonizing the design of the product with the way the purchaser organization does business; from inside their own product development organization, these same ergonomists can be called upon to help harmonize their own organization with the product development process, with the manufacturing process, with the product distribution process, and/or with the product field support process.

4.4 Implementations

The ergonomist can add significant value to iterative pre-production (e.g., internal releases) and mass production/distribution activities. In the pre-production stage, the ergonomist can provide a number of analytic evaluations of the product including heuristic analyses, managing expert reviews, and conducting laboratory-based usability analyses. As required by the FDA Quality System Regulation [19], test procedures that are appropriate for their intended use (validated test procedures that possess the appropriate sensitivity, specificity, and reliability), properly calibrated equipment, and tests that are statistically valid must be employed for usability studies. In the production phase, the ergonomist can assist in job redesign, the development of job aids, as well as recommendations on environmental and organizational issues that would enhance the productivity and job satisfaction of production personnel.

4.5 Compliance Engineering

There exist a large number of ergonomics standards; they address various aspects of the profession's activities and they are not generally well-known outside the profession. The ergonomist on the project team plays a critical role in identifying, interpreting, and designing the product to conform to these constraints.

4.6 Reliability Engineering

Ergonomists are trained to use analytical and laboratory techniques to discover subtle - but potentially more hazardous - use errors. With these same analytical and laboratory techniques, putative mitigations can be evaluated and the residual risks can be properly assessed.

Risk reduction is managed through risk identification, risk assessment, risk mitigation, and then *re-assessment* of residual risks. All members of the design team, including the ergonomist, utilize standard risk analytic techniques (e.g., fault tree analysis, failure mode effects and criticality analysis, or hazard and operability studies). However, the ergonomist begins not from an analysis of the mechanical or electronic parts or from an analysis of the program structure, but rather from a task and function analysis; the focus is *the interface between the tool and the user*.

Unlike the other members of the design team, the focus is on:

1. *hardware issues* (e.g., size, feel, color, and arrangement of physical controls and displays and the impact on their use with and without surgical gloves);

2. *software issues* (e.g., mental workload issues, logic of operations issues, training materials, etc.);

3. *environmental issues* (e.g., the crisis of a patient in cardiac arrest, the boredom and reduced vigilance at the end of a shift, light levels during day and night operations); and

4. *organizational issues* (e.g., purchaser organization administrative procedures for handling/using product and for scheduling work time, including multiple shifts, etc.)

5. Conclusions

As with all methods, the half-century old classical systems engineering method is susceptible to misuse and abuse. Abuses include rigid (inflexible) implementation, sequential (non-iterative) implementation, and planning outside the process. Misuses include: believing that the role is more important than the individual and that individual strength, weakness, and personality are irrelevant; expecting that data will be continuous rather than a discrete series, and that the environment will be static.

The development of products and processes in the real world is fraught with uncertainty. Management theorists have emphasized, for at least four decades, the importance of differentiation and integration in enterprises involved in product and process development. When development methods do not yield the requisite results, enlightened developers will seek new approaches. It is suggested that "agile" methods are a reasoned reaction to the misuse and abuse of the classical systems engineering method.

The agile methods developed by the software community have value for ergonomists. Conversely, applying these methods, ergonomists can contribute to system development and add value to software development and software development processes.

6. References

1 Lawrence P, Lorsch J., Organization and Environment, Cambridge, MA: Harvard University Press, 1967
2 Manifesto for Agile Software Development, Available from: www.agilemanifesto.org, Accessed February 3, 2005.
3 Samaras GM, Horst RL A systems engineering perspective on the human-centered design of health information systems, Journal of Biomedical Informatics, 2005, 38(1):61-74.
4 Gould JD, Lewis C, Designing for Usability: Key Principles and What Designers Think, Communications of the ACM, 1985; 28(3): 300 - 311.
5 Mantei MM, Teorey TJ, Cost/Benefit Analysis for Incorporating Human Factors in the Software Lifecycle, Communications of the ACM, 1988; 31(4): 428-439.
6 Nielsen J, The Usability Engineering Life Cycle, IEEE Computer, 1992 Mar; 12-22.
7 Kreitzberg C, Logical User-Centered Interaction Design Methodology from Cognetics Corporation – cited in Shneiderman B. Designing the User Interface: Strategies for more Effective Human-Computer Interactions, 3rd ed. Reading, MA : Addison-Wesley; 1998, p. 104-7.
8 Mayhew DJ, The Usability Engineering Lifecycle, San Diego: Academic Press; 1999.
9 Endsley MR, From Cognitive Task Analysis to System Design: Designing to Support Situation Awareness, CTA Online Seminar, 2002
10 The term method-ology is used here in the same fashion as bi-ology, psych-ology, soci-ology, etc. It refers to the study of methods, rather than to the methods themselves, which are "ways of doing things".
11 Grady JR, System Validation and Verification, New York, NY:CRC Press, 1998, pp. 3-4
12 Beck K, Extreme Programming Explained, New York, NY:Addison-Wesley, 2000, p. 117
13 Horst RL, Samaras GM, Validation engineering in the Ergonomics of Medical Systems: Application Perspectives, Proc. HFES 47th Annual Meeting 2003, pp. 1458-62
14 Cockburn A, Agile Software Development, New York, NY:Addison-Wesley, 2002, p.152
15 Cockburn A, Agile Software Development, New York, NY:Addison-Wesley, 2002, p.151
16 Kruchten P, The Rational Unified Process, New York, NY:Addison-Wesley, 1999, p. 62.
17 Fowler M, The New Methodology, Available from: www.martinfowler.com/articles/newMethodology.html, Accessed March 10, 2005
18 Schwaber K, Beedle M, Agile Software Development with Scrum, Upper Saddle River, NJ:Prentice-Hall, 2002,

19 FDA Quality System Regulation, US FDA 21 CFR Part 820, October 1996.