Reflections on Barry W. Boehm's "A spiral model of software development and enhancement"

Andreas Nilsson and Timothy L. Wilson
Umeå School of Business, Umeå University, Umeå, Sweden

Abstract

Purpose – The purpose of this paper is to review the content, contributions and subsequent developments of the seminal paper by Barry Boehm, "A spiral model of software development and enhancement" written in 1988. The relationships of this paper to software development, agile projects, real options and present practice are put into perspective.

Design/methodology/approach – Basically an essayist approach is taken. First, the contents of Boehm's paper are reviewed and then associated with subsequent developments.

Findings – Review of the paper as published represents a documentation of cutting-edge software development as it existed at the time. Fundamentally it suggests the viability of a non-linear, customer-influenced, development approach.

Practical implications – This basic approach illustrated in the spiral model of course has found its way into complex project approaches and management.

Originality/value – This paper follows the lines of increasing attention to classics, which is the purpose of this special issue of the journal. In particular, attention is called to the transition of thought on projects and project management from supplier-oriented, linear processes to customer/client-influenced, non-linear ones.

Keywords Project management, Computer software, Agile production, Non-linear approaches, Agile processes, Real options

Paper type Conceptual paper

Introduction

It is imagined that in this special issue there are a variety of papers that have been instrumental in individuals' development. Our attention to Boehm (1988a, b) came about as an ongoing interest in projects, project organizations and project management. As background, in the early 1990s, research was started in the management department here on the study of projects and some implications of their use as organizational approaches and elements of change. By investigating different types of projects, an attempt was made to develop the sociology of temporary organizations involved in projects, while also adding portfolio, general management and strategy issues to the body of knowledge[1]. The initial interest in Boehm was associated with a study on the development of video games (Zackariasson, 2003). Those developments are organized around projects, but in the words of an executive

The authors wish to acknowledge the contribution of Ms Donna Beck, Engineering Librarian at Carnegie Mellon University, Pittsburgh, PA, who was instrumental in leading them to a number of references used in the revision of the original paper. The authors also appreciate the contributions of the anonymous reviewers and special editors of this journal for their insight in revising the paper.

International Journal of Managing Projects in Business
Vol. 5 No. 4, 2012
pp. 737-756

1753-8378
DOI 10.1108/17538371211269031

"a milestone model of product development had basic shortcomings." That is, they did not work. Because of that, an alternative approach was developed at the company under study that satisfied the milestone, cost, and time constraints imposed by the publishers who funded development, but met creativity aspirations of the developers (Walfisz *et al.*, 2006). The description of game development as a spiral eventually led to Boehm (1988a, b) in the literature search. Aside from the similarity in spirals and the aspects of portions of his observations, his apparent frustration with the waterfall model and the use of a quote describing users who do not know what they want, but want it done anyhow were striking.

To put the spiral model into perspective, one might turn to Boehm's own assessment in subsequent papers co-authored by him. In Boehm and Papaccio (1988) they noted:

> The spiral model (also) accommodates any appropriate mixture of specification-, prototype-, simulation-, automatic transformation-oriented approaches to software development where the appropriate strategy of considering the relative magnitude of program risks and the relative effectiveness of various techniques in resolving risks.

Ten years later Boehm's group reflected (Boehm *et al.*, 1998):

> The original spiral model uses a cyclic approach to develop increasingly detailed elaborations of a software system's definition, culminating in incremental releases of the system's operational capability. Each cycle involves four main activities:
>
> (1) Elaborate the system or subsystem's product and process objectives, constraints, and alternatives.
>
> (2) Evaluate the alternatives with respect to the objectives and constraints. Identify and resolve major sources of product and process risk.
>
> (3) Elaborate the definition of the product and process.
>
> (4) Plan the next, and update the life-cycle plan, including partition of the system into subsystems to be addressed in parallel cycles. This can include a plan to terminate the project if it is too risky or infeasible. Secure the management's commitment to proceed as planned.

Subsequent to these beginnings, there have been developments that affect the outlook on projects with regard to flexibility. The purpose of this paper therefore is to review the content, contributions and subsequent developments of the seminal paper by Boehm (1988a, b), "A spiral model of software development and enhancement." It is suggested that it is a classic. It has stood the test of time; the Boehm paper was published 24 years ago and still is the foundation of the approach to software development in particular and complex projects in general. Further, it has been widely quoted. Despite being published in a rather modest journal (*IEEE Computer*), the latest review of Google Scholar indicates that the proposed classic has been quoted in nearly 4,000 (3,589) academic publications. Finally, it was ground breaking. His paper not only described software development and the inherent risk associated with projects; it anticipated agile development projects and contained the elements of real options – all of which are important today. Perhaps more important to project conduct, planning and practice, it deals with non-linear processes and prototyping.

This paper is structured in the following way. First, a short description is presented of the author and the content of the paper. After that, some materials are taken out of context and associated with subsequent developments in project management.

In that regard, Boehm kept active after the seminal paper. Two books are available that illustrate his contributions and thinking – one on his lifetime contributions (Selby, 2007) and another published with Boehm and Turner (2004) on agile development. Finally, some ideas are presented for future development and research.

About the author

The paper was written by Barry W. Boehm who is presently TRW Emeritus Professor of Software Engineering and Director, Center for Software Engineering, University of Southern California. To put his career in perspective, in his assessment of the development of software approaches and processes, Glass (1998) identified three eras – the pioneering era (1955-1965), the stabilizing era (1965-1980) and the micro era (1980 to the present)[2]. For each of these eras he entered his personal recollections as well as "hard" facts of things happening in the industry and solicited memories from 14 other contributors. In the pioneering era, he shared a cubical with four or five other programmers; there was nothing electronic in the cubicles and all development was done on cards for the mainframe. In the stabilizing era, his number of office sharers dropped to three and IBM made a big splash with its 360. Development was still for mainframes and turnaround was slow by present standards (overnight), but the value of software became recognized – a few software houses began to offer products. It was not until 1982 in the micro era that he got his first computer on his desk and had his own office. In his words, "IBM became king of the mainframe market, only to find the mountain much smaller than it had imagined. [. . .] Microsoft is king of micro software" (and Bill Gates became the richest man in the world).

Barry Boehm has gone through each of these eras. In his chapter in Glass (1998, pp. 67-97) he recalled his experiences starting with a summer job while still a student at Harvard in 1955 at General Dynamics through his experiences at Rand. It was at Rand that he was permitted to finish his PhD in mathematics at UCLA. The Rand work primarily dealt with rocket development and usage. In the early 1970s, he actually tried to get out of software because he sensed that he was "basically providing services to engineers and system analysts. [. . .] At the time, it seemed to me that they were doing more useful and satisfying things than programmers were." His career took a turn when Rand lent him to the Air Force to run a two year mission analysis on its gaps in information processing. Instead of super computing and large screen displays, it was found that the biggest information processing problems were in the software area. Upon completion of that duty, he left Rand to take a position at TRW as its director of software research and technology in 1973. His mission was "to come up with an integrated set of procedures, methods and tools that would bring the software problem under control" (p. 95).

In Selby's (2007) anthology of Boehm's lifetime contributions to software development, management and research, Professor Boehm is characterized as one:

> [. . .] who helped pioneer the field of software engineering and crystallize it into a mature scientific and engineering discipline. Barry's leadership, experience and wisdom profoundly impact software developers, managers, and researchers worldwide, and his spirit exemplify the continual innovation and drives the software engineering field forward[3].

That compendium contained 43 papers organized into nine chapters (plus his summary reflections) – each of these chapters was introduced by one of his peers. In these introductions his contributions are put into perspective: "Barry lays the foundation for

software architecture as well as the framework for quantitative analysis in software engineering" (Bernstein, 2007, p. 1); "Barry Boehm's seminal ideas and extensive publications establish the field of software economics" (Selby, 2007, p. 87); "The hallmark of Boehm's work is the way in which he draws insights by analyzing data collected from real projects" (Pyster, 2007, p. 227); "Barry's mix of the intellectual and emotional factors has resulted in breakthrough ideas and workable models that have educated us on more economically feasible approaches to exploiting software" (Royce, 2007, p. 315)[4]; "Not tracking a risk on Barry's top-ten list was an a priori indication that was not being focused on all the areas where it was needed" (DeMarco, 2007, p. 383); "The articles in this chapter provide an excellent example of why Barry Boehm occupies a nearly unique position as a bridge between the world of software engineering research and software development practice" (Osterweil, 2007, p. 299); "Boehm's articles reek of realism. He has been there, done that" (Brooks, 2007, p. 573); "He [. . .] predicts the future in terms of the systems we might expect and the techniques needed to build them" (Basili, 2007, p. 627); "Barry Boehm's writings on value-based software engineering provide a principled new framework through with to critically understand and improve a wide range of traditional concepts, practices, methods and tools in software engineering" (Sullivan, 2007, p. 731).

Professor Boehm has served on the boards of several scientific journals, including *IEEE Computer*, the *IEEE Transactions on Software Engineering*, *IEEE Software*, *ACM Computing Reviews*, *Automated Software Engineering*, *Software Process*, and *Information and Software Technology*. He has also served as Chair of the Air Force Scientific Advisory Board's Information Technology Panel, Chair of the NASA Research and Technology Advisory Committee for Guidance, Control, and Information Processing, and Chair of the Board of Visitors for the CMU Software Engineering Institute. He is a Fellow of the primary professional societies in computing (ACM), aerospace (AIAA), electronics (IEEE), and systems engineering (INCOSE), and a member of the National Academy of Engineering. His current research interests focus on value-based software engineering, including a method for integrating a software system's process models, product models, property models, and success models called Model-Based (System) Architecting and Software Engineering (MBASE). His contributions to the field include the Constructive Cost Model (COCOMO), the spiral model of the software process, the Theory W (win-win) approach to software management and requirements determination, the foundations for the areas of software risk management and software quality factor analysis, and two advanced software engineering environments: the TRW Software Productivity System and Quantum Leap Environment[5].

Re-reflections

Dichter and Pease (1995, pp. 22-36) have put events in a historical context with regard to the waterfall model and Boehm's subsequent development of the spiral model. These authors wrote:

> The waterfall development model is still the most accepted model today in large software organizations. [. . .] The process steps in a waterfall development model usually include requirements analysis, design, code and unit test, integration test and acceptance test.

Referring to representing the software development cycle as a waterfall, Hatley and Pirbhai (1988) were quoted as saying that:

> [. . .] this view obscures the true nature of systems development: it has always been an iterative process in which any given step can feed back and modify decisions made in a preceding one. Failure to recognize this fact has been a major difficulty in project management, giving an erroneous feeling of warmth (We are done with requirements! We can proceed with design!).

Dichter and Pease further relate, "As you can see, it isn't the waterfall method that is so bad, but the bad assumption that its 'water' flows down only to the next step."

Around the same time Boehm was developing the rapid prototyping model, other engineers were coming to similar conclusions (Dichter and Pease, 1995, p. 27)[6]. Connell and Shafer (1989) developed the technique called structured rapid prototyping. That is, in their book *Structured Rapid Prototyping*, they developed an approach that evolved into a development process. The primary technique was called evolutionary rapid prototyping process (ERPP). The fundamental difference between ERPP and the spiral model was that, according to Connell and Shafer, if you have the correct tools, you do not need to "build one to throw away." In ERPP, one was to evaluate the prototype in the tuning phase to see if it meets the user requirements and the requirements for performance and reliability under loaded conditions (they call this "stress testing" and "performance testing"). It was asserted that one also should evaluate the prototypes in terms of maintainability and error handling, since most prototypes still have some "quick-and-dirty" code in them. Another method, called Evo (evolutionary delivery method) described in *Principles of Software Engineering Management*, by Gilb (1988), was also an iterative approach of that period[7].

Boehm's (1988a, b) paper, however, has survived as relevant; the others not so much so. Dichter and Pease (1995, p. 26) go on:

> Barry Boehm developed the spiral model as a controllable way of doing rapid prototyping. [. . .] In the spiral model, you still have the same steps as you would in a waterfall model, but you do some of them more than once. Each repeated step is a closer approximation to the desired result – such as a new component of the final product. The expression "Design a little, code a little, test a little" applies to rapid prototyping and to the spiral model.

In assessing the early spiral model Royce (2007, p. 315), in one word, stressed the need for flexibility in approaches. In his words, "The absolute best thing about software is that it is soft – it can be programmed to do almost anything and changed over time." Frankly, in the initial exposure to Boehm's paper, that aspect of the paper affected the research here. In the approach to the present paper, however, interest was drawn to Figure 1 in Boehm's paper, which really summarizes the article (shown here in simplified form as Figure 1). It is a construct that snakes through a matrix of which the vertical axis is "cumulative cost" and the horizontal one "review"[8]. The spiral shape forces one to alter a stereotypically, mental image of projects as linear processes to an appreciation of their frequently curvilinear nature.

The significance of the model is dictated by the four quadrants defining the diagram – starting upper left and moving clockwise:

(1) determine objectives;

(2) evaluate alternatives, identify, resolve risks;

(3) develop, verify next-level product; and

(4) plan next phase.

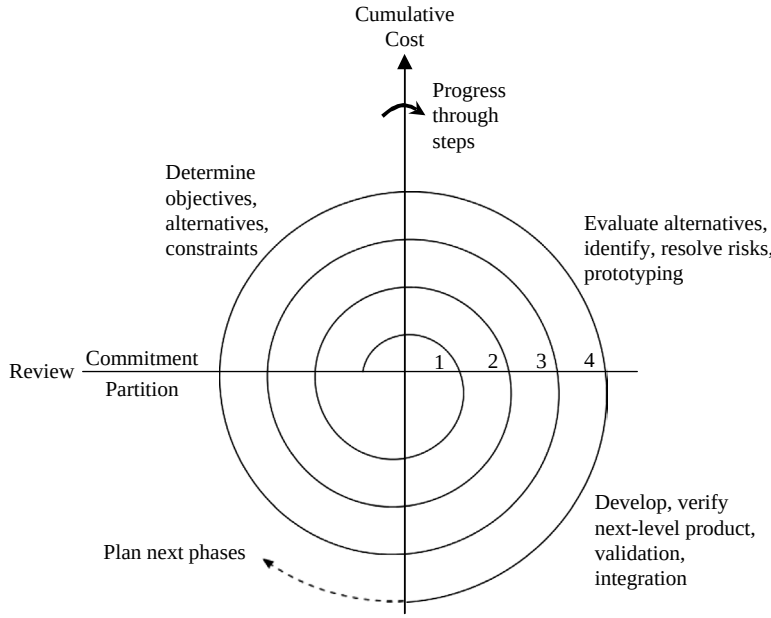

Figure 1.
Boehm's spiral model of the software process

Source: Adapted from Boehm (1988a, p. 64)

The radial dimension of the spiral represents the cumulative costs incurred during the project; the angular dimension represents the progress made in completing each phase of the spiral. Four phases are sketched in the model and the natures of these phases are reproduced in Table I here. It is observed that the first phase is very basic and each additional phase becomes more complex until in the forth the final product is produced. Each stage or cycle starts with a risk analysis and develops a prototype. Dichter and Pease (1995, pp. 19-20) put prototyping in perspective – they can help eliminate project risks, especially those arising from unknowns.

The second observation made in re-reading the paper for this article is that the author stresses the risk-driven nature of the model. In his words, "The major

Phase 1	Phase 2	Phase 3	Phase 4
Risk analysis	Risk analysis	Risk analysis	Risk analysis
Prototype 1	Prototype 2	Prototype 3	Operational prototype
Concept of operation	Software requirements	Software product design	Detailed design
Requirements plan and life-cycle plan	Requirements validation	Design validation and verification	Detailed design
	Development plan	Integration and test plan	Code
			Unit test
			Integration and test
			Acceptance test
			Implementation

Table I.
The phases in the spiral software process

distinguishing feature of the spiral model is that it creates a *risk-driven* approach to the software process rather than a primarily *document-driven* or *code-driven* process." By "risk" Boehm clarifies his ideas of their nature in the article; his top ten include:

(1) personnel shortfalls;
(2) unrealistic schedules and budgets;
(3) developing the wrong software functions;
(4) developing the wrong user interface;
(5) gold plating;
(6) continuing stream of requirements changes;
(7) shortfalls in externally furnished components;
(8) shortfalls in externally performed tasks;
(9) real-time performance shortfalls; and
(10) straining computer science capabilities.

Risk analyses are now routinely done before initiating a project (Nicholas, 2001, pp. 306-39). In that regard, although some of Boehm's risks are specifically associated with software development, over half could apply to any project. Thus, the ideas of starting with a risk analysis, as well as identifying general risks, are supported (perhaps initiated) by this paper.

Finally, and this subject will be discussed subsequently, it is striking that Boehm came close to anticipating the use of real options (MacMillan *et al.*, 2006) in judging the veracity of a project. That is, the implementation of the first phase and each subsequent phase provides greater insight into the overall value of the project. This approach reflects Boehm's interest in economics going back to his early career (Boehm, 1984). He wrote (Boehm, 2002):

> In 1968, I became head of a group at Rand called Computer Systems Analysis. Besides doing defence command-control and information systems analyses, we were involved in helping state and local governments make better use of computers and software. Several of these experiences convinced me that getting a software system's *economic* (emphasis added) and systems analysis right was at least as important as getting its programming right.

Content and structure

Although the paper was written fairly early in the contributions from Boehm's career cited by Selby (2007)[9], it came after Boehm had 33 years of experience in developing software. Consequently, Boehm was describing not only the evolving (at the time) approach to development, but also reflecting on his past experiences. It also was driven by the experience in developing software; Royce (2007, p. 316) indicates:

> At the time that Barry's spiral model article was published, TRW was several spirals into one of the best software development case studies in the defense industry, the Command Center Processing and Display System Replacement (CCPDS-R, a very large US Air Force command and control system).

The article itself took as its starting point four software process models existing at the time before describing the spiral model as one solution to many of the problems with

the existing models. After describing how that model worked, its advantages and difficulties were evaluated and its implications discussed.

The first four pages of the article are devoted to describing a brief history of four software development approaches: code-and-fix model, stage-wise and waterfall models, evolutionary development model, and transform model. These approaches may be of interest to the history of development at the time, but repetition here delays getting to the point. The approaches are thus digested and summarized in Table II for the interested reader[10]. The first column in this table identifies the model; the second gives a short description of the model and the third the difficulties in using the model.

The remedy for the weaknesses of these four approaches was assessed to be Boehm's spiral model, which simultaneously captured the strengths of each of the other models and, as he suggested, reduced to the proper model where appropriate. The author expanded on this thought by explaining how the model worked differently in different settings and adjusted to "become" other models when they were best suited. Hence, when there was low risk for interface problems and high risk in changes in a budget or schedule, the model worked similarly to a waterfall model. On the other hand, if there was a low risk in for example losing the budget and a high risk in user interface, it would work in a manner similar to an evolutionary development model.

Boehm appeared to be well aware of difficulties in applying the model. He therefore suggested three major issues to resolve before it could be applied universally. First, it needed to incorporate the special features of contract software, which required a special approach to the project; second, the model needed project teams experienced in evaluating risks, and third, the steps in the model needed to be developed to give all project participants support.

Model	Brief description	Apparent difficulties
Code-and-fix	Two steps: first, write some code and second, fix problems in the code. Basically, it is the software analog of Peters (1992, pp. 577-85) try it, break it, touch it mode of operations	The code quickly becomes messy; the product often does not meet the requirements, and it is expensive to fix the code once established
Stage-wise and waterfall models	Defining features of feedback loops between successive stages and prototyping built into the model. Emanating from 1956, suited for large projects, and called the approach of the 1970s	Much documentation needed in the approach, which slowed development and could lead to elaborately specified approaches but in wrong order
Evolutionary development	Incrementally stage-wise development where each increment has its objective and next step is commenced as the previous is finished	Lack of initial direction made coding difficult; the coding potentially comes in wrong order; adhering to inter-linkage with other systems often comes too late
Transform model	Establishes a satisfying program with (1) formal specifications, (2) automatic transformation into code, (3) iterative loops to improve code, (4) exercise of the resulting product and (5) an iterative loop to adjust to outer requirements	Best used with small programs. Also, difficulties as it requires that other systems be flexible and faces problems in upgrading

Table II.
Brief description of alternate approaches at the time

Source: Boehm (1988a, b)

The article finished with four concluding remarks:

(1) The spiral model is more adaptable than other models, which is valuable in large and complex projects. The use in developing the US Air Force command and control system validates this conclusion.

(2) The use, so far, of the model has been successful. The highlight from this section asserts "all of the projects using the system have increased their productivity at least 50 percent".

(3) Some more work is still needed on the model. The author noted, "The three primary challenges involve matching to contract software, relying on risk-assessment expertise, and the need for further elaboration of spiral model steps".

(4) Parts of the model could be useful if implemented/incorporated in other models. In particular, the ideas of a risk management plan are compatible with current process models and are very helpful in overcoming major sources of project risk.

Key themes or relevance to project management

The "out of the box" idea expressed by Boehm (1988a, b) was that software projects were not Cartesian-oriented, Gantt-chart types of things. Put another way, the basic idea of the spiral approach was that projects would proceed sequentially with an output after each step that would satisfy an agreed-upon need of the customer. This flexibility in approach did a number of things. First, it was associated with risk and that in itself was a step forward in project management; it is a universal feeling that assessment of risk is a critical first step in the initiation of a project – to the extent that a formal document may be produced in large projects. The second thing that comes out of his paper with regard to project management is that it is possible to develop output of a qualitative, even unspecified nature. The quote "I'll know it when I see it" characterized this aspect of the work. In the experience with video game development noted previously, games were observed to be developed that were "fun to play" and "immersed the user" (Walfisz *et al.*, 2006). Finally, the whole idea of a non-linear approach has caught on that encompasses project management in general. Lenfle and Loch (2010, p. 40) have indicated a need to revisit a stage-gate approach (Cooper, 1994)[11] to project management that traces back to US Secretary of Defence McNamara in the 1950s; Collyer and Warren (2009, p. 359) have expressed a similar observation and cite examples in automobile manufacturing, project selection and prescription drug development.

The recent development in project management of agile methods in order to handle risk and complex, unpredictable environments goes back to Boehm. As a foundation, sometimes expressed (Abrahamsson *et al.*, 2010; Collyer and Warren, 2009; Lenfle and Loch, 2010; Rising and Janoff, 2000; Williams, 2010) and sometimes not, the methods share a common foundation in the spiral model of Boehm (1988a, b). This development has been an area pursued by Boehm himself, particularly in software (Boehm and Turner, 2004). These authors wrote, "Agile methods are an *outgrowth* (emphasis added) of rapid prototyping [...]" (p. 16) and "In general, agile methods are very lightweight processes that employ short iterative cycles; *actively involve users* (again, emphasis added) to establish, prioritize and verify requirements [...]" (p. 17). One side (agile v. plan driven) was not taken in the approach to development, although there is an apparent fondness for an agile process. Although Boehm (1988a, b) is seen as a key milestone in the development of the agile method, his ideas actually

suggest the reconciliation between agile and waterfall, instead of merely arguing in favor of agile over waterfall. In the paper he wrote, "[. . .] the spiral model can accommodate most previous models (including waterfall) as special cases [. . .]" (p. 64). In Boehm and Turner (2004), they summarize their approach in the preface (p. xix):

> It (the book) is based on our own development experiences, [. . .] We discuss the subject absent a need to choose sides. Our goal is to help you (the reader) gain the understanding and information you need to integrate the approaches in a manner that best fits your business environment.

The Boehm paper also formed the basis for real option treatment that evolved later in project management. This contribution might be viewed as being the other book end to the risk awareness approach. Briefly put, options provide the right "but not the obligation" to invest further in a project after a certain stage is reached. Their value, therefore, is driven by the possibility of achieving a large upside gain combined with the fact that companies can usually abandon their projects before their investment in them has cost too much, thus limiting the downside (Van Putten and MacMillan, 2004). The approach thus finds application in companies involved in high technology, high risk developments. In her interview with the *Harvard Business Review* (Nichols, 1994), the chief financial officer at Merck said:

> Option analysis, like the kind used to value stock-options, provides a more flexible approach to valuing our research investments than traditional financial analysis because it allows us to evaluate those investments at successive stages of a project. [. . .] When I look at investments from this perspective, I am able to value the project if it is successful. [. . .] But I am also able to value the project if it doesn't result in a new product, because I know we will have gained some new scientific information that will help us in the future.

Impact and diffusion

Early on Oman and Lewis (1990, p. 225) wrote that the paper addressed the question, "How can we combine the effectiveness of rapid prototyping with the consistency of the waterfall model?" Barry Boehm's answer, "A spiral model of software development and enhancement," is a new risk-driven approach to software development. They continue:

> Note, however, that Boehm's model accommodates all existing methods. For example, you can go through several iterations of the spiral and then proceed to a more traditional specification document. His key point is that at each iteration there should be a risk analysis prior to building the prototype. Throughout the paper – in his examples, discussion, and figures – risk analysis and assessment is prominent. Although sketchy on implementation details (Boehm admitted his model needs further elaboration), the paper gives us enough information to start using his methods in our own software development efforts.

In Walker Royce's (Selby, 2007, pp. 315-16) assessment of the Boehm paper, he wrote:

> Barry's insight was a micro-level improvement. He added a whole dimension to process thinking. [. . .] No matter where you are looking for modern software process guidance, the roots of today's methods flowed through the spiral model foundations.

Put another way, this paper was indeed a seminal article and forms the basis for software thinking and development to this day. Royce also notes, however, that the approach was such a change for management thinking that, "It has proven surprisingly hard to implement within the real-world constraints and organizational inertia that most of us live within." In spite of this difficulty, it was as if the world was waiting for

the approach. The paper appeared in 1988 – two years later the approach was in a text on software development (Ould, 1990 quoted in Deakins and Dillon (2005)).

Brooks (2007, p. 573) expressed this opinion of Boehm's contribution:

> A tremendously important and contribution is the *spiral model* (emphasis added) of software development and his continued, vigorous, and effectively advocacy for replacement the disastrously bad waterfall model with spiral models. Key to this crucial way of thinking is the early recognition that requirements must evolve as development proceeds and that both requirements and interfaces must evolve as prototypes are built and used.

In reflecting upon Boehm's (2000) paper on managing expectations he emphasizes this importance of prototypes, "If a picture is worth a thousand words, a prototype can be worth a thousand pictures" (Brooks, 2007, p. 576).

To complete the further impact of the spiral model for this paper, the Thompson Reuters ISI Web of Knowledge[12] database was used to trace the diffusion of the paper into the subsequent academic literature. Since the paper was published in 1988, there have been 235 citations in journals and conferences registered in the database with apparent peaks in 1993, 1999 and 2009 corresponding to approximately five, ten and 20 year "rediscovery" of the important aspects of the paper. The top ten papers citing Boehm and then being frequently re-cited fall mainly into three categories – further software development (Beck, 1999; Schreiber *et al.*, 1984; Rising and Janoff, 2000), associated topics (Studer *et al.*, 1998; Mili *et al.*, 1995; Boehm and Papaccio, 1988; Boehm *et al.*, 1998) and elements of prototype usage (Nissen, 1998; Hartson and Hix, 1989; Baskerville and Stage, 1996). Schreiber *et al.* (1984), for instance, indicated "In CommonKADS, project management and development are separated. [...] The development process (of CommonKADS) proceeds in a cyclic, risk driven way similar to Boehm's spiral model". Baskerville and Stage (1996) suggested that the spiral model was a constructive attempt to employ risk analysis as a means to determine the relevance of either a specification or a prototype. It was also a source of inspiration for other approaches such as a contingency model that determines a proper mix of prototyping techniques and more structured approaches based on uncertainty and complexity. Finally, Rising and Janoff (2000) in describing Scrum wrote:

> Barry Boehm's spiral model certainly addresses the element of risk in software development. The essential ideas behind the spiral model are exactly the same as those in scrum – just speeded up!

Table III summarizes the nature of these papers and the reference to Boehm's spiral model.

Because it takes some time for a paper's value to be recognized and then get cited, more recent papers, although less frequently cited, reflect further where the spiral concept is going. The concept is still being used in software development, but now in games (Stacey and Nandhakuman, 2009) and also in elements of prototype usage (Lu *et al.*, 2009; Khoo *et al.*, 2009; Lindgaard, 2009; Hansen and Rennecker, 2010).

Ideas of spirals and prototypes have also found their way into general PM practice in situations where flexibility is required. Collyer and Warren (2008), for instance, wrote:

> The "waterfall" lifecycle has strictly limited overlap between phases, and high levels of planning and process control. This is suitable for projects with a well understood scope and enabled using proven technologies. This approach is favoured in the construction industry. In unpredictable environments waterfall does not allow sufficient adaptability to permit

Year	Rank	Authors	Title	Comment
1988	3	Boehm and Papaccio	Understanding and controlling software costs	Reflects upon the spiral model, "The spiral model (also) accommodates any appropriate mixture of specification-, prototype-, simulation-, automatic transformation-oriented approaches to software development where the appropriate strategy of considering the relative magnitude of program risks and the relative effectiveness of various techniques in resolving risks"
1989	9	Hartson and Hix	Toward empirically derived methodologies and tools for human-computer interface development	Reported on the fourth generation development of human-computer interfaces and proposed a star model. Contrasted with Boehm – "for development of all software" and "includes prototypes and testing phases". Star specific and continuous. References two early papers (Ramamoorthy *et al.*, 1984; Swartout and Balzer, 1982) as relevant to non-linear approaches
1994	5	Schreiber *et al.*	CommonKADS – a comprehensive methodology for KBS development	"In CommonKADS, project management and development are separated. [. . .] The development process (of CommonKADS) proceeds in a cyclic, risk driven way similar to Boehm's spiral model"
1995	2	Mili *et al.*	Reusing software – issues and research directions	Reuse involves both products and processes. Contrasts waterfall life cycle with spiral and prototyping. All involve some measure of analysis, design, coding and testing. WLC implies entire system is analyzed before any part is designed or implemented. Both spiral and prototyping prescribe analysis ← → testing on system increments
1996	10	Baskerville and Stage	Controlling prototype development through risk analysis	The spiral model is a constructive attempt to employ risk analysis as a means to determine the relevance of either a specification or a prototype. It has also been a source of inspiration for other approaches such as a contingency model that determines a proper mix of prototyping techniques and more structured approaches based on uncertainty and complexity
1998	1	Studer *et al.*	Knowledge engineering principles and methods	All the development activities, which result in the stepwise development of the different (KBS) models, are embedded in a cyclic and risk driven life-cycle model similar to Boehm's spiral model
1998	6	Boehm *et al.*	Using the win-win spiral model: a case study	Resolves two difficulties in spiral model implementation by adding stakeholder interests and life-cycle objectives
1998	8	Nissen	Redesigning reengineering through measurement-driven inference	Uses a spiral to describe second wave of BPR – refers reader to Boehm to explain "common notation for evolutionary processes"

Table III.
Summary of most quoted papers associated with Boehm's spiral model

(*continued*)

Year	Rank	Authors	Title	Comment
1999	4	Beck	Embracing change with extreme programming	In citing the antecedents to XP, Beck says, "Barry Boehm's spiral model was the initial response to the waterfall"
2000	7	Rising and Janoff	The Scrum software development process for small teams	"Barry Boehm's spiral model certainly addresses the element of risk in software development. The essential ideas behind the spiral model are exactly the same as those in scrum – just speeded up!"

Table III.

> maximisation of benefits. Novice managers might believe this approach is lower in risk but in environments with high levels of unknowns it can have a higher risk of failure because of the time and effort required to be invested before environmental incompatibility is discovered. High levels of control inhibit the adaptability needed to maximise business benefits in dynamic environments. [...] In the rolling wave approach the plan for each phase is completed at the end of the preceding phase. This allows for improved environmental adaptation. With the "iterative" approach all phases run in order many times over. Successive releases evolve into a more complete product. This is a good way to reveal unknowns and adapt to a changing environment. Iterative is also known as "spiral" or "incremental". When there is limited knowledge about how a product might interact with its environment, an "iterative" approach is an effective way to test and collect that information, and minimising resource expenditure on bad choices. Some versions of the "iterative" approach use feedback as the primary control mechanism, rather than planning. The feedback is driven by regular tests and releases of the evolving product. Agile development for instance tries to keep scope small, and to deliver early and often. It focuses more on communication than process.

Examples were cited in automobile manufacturing, project selection and prescription drug development to illustrate their point.

Along similar lines Lenfle and Loch (2010) have written:

> Similar criticisms (of a phased approach) were voiced in the field of software development against the traditional "waterfall model," another application of the phased approach. The emphasis on complete system definition before entering development proved impracticable in innovative projects, leading to potential managerial disasters.

Examples in this case were drawn from the history of the Manhattan project during Second World War and the more recent NASA developments.

Issues and insights

From the start, there have been two issues associated with the spiral model. They were/are:

(1) Implementation of the approach is vague. Boehm (1988a, b, p. 70) himself wrote, "In general, the spiral model process steps need further elaboration to ensure that all software development participants are operating in a consistent context."

(2) The prototypes commonly are "throw aways" (Dichter and Pease, 1995, p. 27). They are developed primarily to determine the risks for the next step. One argument for following the Connell and Shafer approach was that prototypes were built in their tuning stage to see if they met user requirements (quoted in Dichter and Pease (1995, p. 27)).

One might also note that the model describes a process and is not well-suited to controlling a project. In a field study (Nilsson, 2008) associated with the development of software, this observation was made on Scrum[13]:

> Scrum is a model of how to run a project, it says nothing about how to develop software but about how the interaction and the actual work in the team will be commenced (Edstedt, 2011, translated from the original Swedish).

Further, agile methods derived from the spiral model have their limitations. Boehm and Turner (2004, p. 39) write, "[. . .] the plan-driven world is considerably ahead of the agile world in dealing with quality or non-functional requirements such as reliability, throughput, read-time deadline satisfaction or scalability."

The model thus remains relevant for its conceptual value. It sees a certain class of projects, software initially but other complex projects later, which are better treated as customer/client-influenced, non-linear ones. That has been a big step in project management.

It is suspected that future developments will further validate a spiral model approach. Collyer and Warren (2009, p. 359), for instance, have written the following about development of prescriptive drugs:

> Researchers cannot just sit down and write a plan guaranteed to deliver a cure for cancer. They experiment, identifying likely possibilities, and methodically eliminate dead ends. The time spent testing the ideas that don't work out is just as important as the time spent testing the ones that do. The ability to select more promising ideas is enhanced by the elimination of others.

The process generally is expensive, fraught with risks, takes a lot of time and requires patient, trial and error on the part of participants.

For purposes of this paper, the steps might be put into a spiral formalism. The process is initiated by understanding the disease, the pre-discovery step (Innovation.org, 2007). The goal in this step is to understand the disease and choose a target molecule[14] that seems instrumental in affecting the disease. This work can go on for years by scientists in pharmaceutical research companies, government, academic and/or for-profit research institutions who contribute to this basic research. In Boehm's model, this phase of the work might be the inner-most step in the process or the trigger for the spiral to start.

One of the initial steps toward commercialization would be the discovery step, which could take three to six years, covering 5,000-10,000 compounds. The goal would be to find a drug candidate by creating a new molecule or selecting an existing molecule as the starting point. Tests of various types are conducted on that molecule and then optimized to make it work better. Supposing success in this step, the more promising molecules would go into preclinical tests. Researchers in these tests ascertain the safety and effectiveness in the lab and in animal models. The goal is this series of tests is to determine if the drug is safe enough for human testing and under normal situations, the number of alternatives might be reduced to 200-250 compounds. Subsequently, the process would lead to pre-clinical trials where the prototypes would be reduced to five candidates or so. From here, the FDA becomes involved through its investigation of new drugs (IND) procedures to final review. Table IV summarizes the process as it might be described in a spiral-type format developed in Table I.

Of course even this approach is idealized as Collyer and Warren (2009) suggest. Under the best of circumstances Innovation.org (2007) indicates the process is a lengthy

Phase 1 – discovery	Phase 2 – pre-clinical	Phase 3 – clinical trials	Phase 4 – process development
Risk analysis – general project considerations + safety concerns	Risk analysis – general project considerations + safety concerns	Risk analysis – general project considerations + safety concerns	Risk analysis – general project considerations + investment options and reproducibility concerns
Prototype 200 – 250 promising compounds	Prototype 3 – reduction to five compounds	Operational prototype	Manufacturing process
Drug requirements	Product design – safety and effectiveness focus	Detailed design of trials	Integration and test
Requirements validation	Design validation and verification	Detailed design of final product	Acceptance test
Development plan	Test plan for FDA	Validation for FDA	Distribution Ongoing coverage of results

Table IV.
The phases in the development of a prescriptive drug as a spiral process

one – perhaps 14 years through FDA testing, and that does not assure success. Naik (2012) in summarizing promising research on Alzheimer's indicates "Over the years, drugs in about a half-dozen late-stage human trials have failed to make the cut". Thus, the process must recycle from stage 3 back to stage 1 and start all over again.

Avenues for future research

The development of Boehm's paper, indeed this approach, argues strongly for publications associated with an experiential base. In pulling together his model, Professor Boehm had been through the rigors of code-and-fix, stage-wise and waterfall models, evolutionary development, and transform models. Coming off the success of the TRW CCPDS-R project, which was treated as a spiral, he elected to share his insight with us – and we are the better for it. It also argues for incorporation of people and thought outside of product management. Boehm's academic background was as a mathematician; one does not know what his thought process was, but it was likely different than those of us who have grown up and developed as project managers and engineers. Boehm, as a mathematician, was probably more comfortable with spirals than most of us might be and thus the incorporation of that structure into management theory. In applying to the general projects sphere, that recognition argues for a projects-as-practice approach for future research in project development and management. The importance of this approach has been recognized and a special issue of the *International Journal of Managing Projects in Business* on exactly that topic will be published soon. The treatment of creative projects, which appear to be neither Cartesian nor spiral, could use a Boehm type breakthrough. It has been suggested that Simon's (1996, pp. 162-3) description of an artist at work is useful (Walfisz *et al.*, 2006), but the field could surely put to good use a graphical model of the creative process.

Second, the paper came out of recognition of good, or at least better, practice in software development at the time. Royce (2007, p. 316) suggests that the output of the TRW project was instrumental in the development of the spiral. Boehm made a note on artistic license when it came to the spiral figure (p. 65) suggesting everything might not have been as smooth in the TRW project as it seems. In the research for this paper,

frequent citations were found on the spiral model as a concept, but none where the spiral was actually drawn from information gathered on a project. Insofar as we tend to understand best those phenomena that we measure, it would be interesting to see an empirically drawn spiral for a project. Perhaps this task would be a suitable project for a graduate student interested in projects.

Along other lines, Boehm was interested in risk management and started each of the spiral cycles with assessment of risk. In the implications section of the paper, he suggests that:

> Even if an organization is not ready to adopt the entire spiral approach, one characteristic technique that can easily be adapted to any life-cycle model provides many of the benefits [...] This is the Risk Management Plan [...]

He continued his work in that area. Boehm and Ross (1989) argued that IS project risk management could be used to avoid disasters, reduce rework, help balance work and effort, and stimulate win-win situations. Central to the issue of improving project performance is the application of risk management processes to reduce human error and promote reliability (Boehm, 1991). Nevertheless, this focus on risk seems to have slipped over time. For instance, in a study built on these observations, Kutsch *et al.* (2012) found IS project managers disengaged from prescribed risk management process of identification, analysis and response. That is, the findings revealed that prescribed risk management processes were established in every project, but significant differences were found in the way that individual projects accomplished risk identification, analysis and response. The major obstacle in increasing risk management reliability was less the issue of not detecting changes in the environment (risk), but of actions taken to prevent risks from happening. It was concluded that the key challenge for IS project risk management lies in finding an appropriate balance between routine-based and mindfulness-based reliability. If project risk management and its underlying processes are not to be discredited, the behavior of project managers when confronted by uncertainty should be considered and action taken to discourage project managers' irrational intentions to merely assess, but not to actively manage risk (Kutsch *et al.*, 2009). Undoubtedly work of this type should be continued.

Fourth, the other bookend to risk in the spiral model is the element of prototype usage as suggested by Boehm (1988a, b). See, for instance, Lu *et al.* (2009) – computer-based training tools; Khoo *et al.* (2009) – entertainment systems; Lingaard (2009) – computer interactions; Hansen and Rennecker (2010) – interpretive team interactions. The topic of prototype itself is well developed. If one googles the term in Google Scholar, one finds the usual eight citations per page (frequently in prescription drug development as covered here) for as many pages as one wants to search. Likewise, if one uses the ordinary Google search engine, one finds a very nice Wikipedia article that covers the nuances of prototype usage across a variety of applications[15]. Further, the linkage of prototype and product management produces a like number of citations – many of the consulting type. Consequently, further research in the area will have a rich background on which to draw, but will likely be constrained to specific applications of interest.

Finally, in their review of flexible, novel projects Lenfle and Loch (2010), a need is seen for:

- theory development in decision making in flexible situations;
- budgeting approaches for learning in projects;

- consideration of project management contributions to strategy; and
- the development of templates for flexible processing.

It would be difficult not to generally concur with that list, and when completed could very well be a classic of the Boehm stature.

Notes

1. www.fek.umu.se/project/ (accessed July 28, 2011).
2. The "present" being 1998, the year in which the book was published.
3. Quote taken from the author's dedication of the book – no page number.
4. This chapter is the one in which the spiral model paper appears.
5. http://csse.usc.edu/people/barry.html (accessed March 1, 2011).
6. Hartson and Hix (1989) reference two earlier papers (Ramamoorthy *et al.*, 1984; Swartout and Balzer, 1982) relevant to the need to treat non-linear approaches in software development.
7. Noted as an alternate process model in the Boehm (1988a, b) paper.
8. Note the mixed nature of the axes – one quantitative and the other qualitative, which further supports a qualitative interpretation of the diagram.
9. Number nine in the 43 "lifetime contributions" cited by Selby (2007).
10. Dichter and Pease (1995, pp. 24-5) added a fifth process – the "V development model", which was really the waterfall model in "V" shape that showed the opportunity to recycle between certain stages.
11. "Stage-gate" in general corresponds to "waterfall" in software.
12. www.webofknowledge.com
13. Note Rising and Janoff's reflection above on Scrum, "The essential ideas behind the spiral model are the same as those in Scrum – just speeded up!"
14. "Molecule" is the term used in the reference. It is the general chemical entity that affects the "virus" or "gene" carrying the disease.
15. en.wikipedia.org/wiki/prototype (accessed April 7, 2012).

References

Abrahamsson, P., Oza, N. and Siponen, M.T. (2010), "Agile software development methods: a comparative review", in Dingsoyr, T., Dyba, T. and Moe, N.B. (Eds), *Agile Software Development: Current Research and Future Directions, Conference Proceedings*, pp. 31-59.

Basili, V.R. (2007), "Software engineering state of the art and practice", in Selby, R.W. (Ed.), *Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management and Research*, Wiley, New York, NY, pp. 627-32 (Chapter 8 – Introduction).

Baskerville, R.L. and Stage, J. (1996), "Controlling prototype development through risk analysis", *MIS Quarterly*, Vol. 20 No. 4, pp. 481-504.

Beck, K. (1999), "Embracing change with extreme programming", *IEEE Computer*, Vol. 32 No. 10, pp. 70-7.

Bernstein, L. (2007), "Software architecture and quality", in Selby, R.W. (Ed.), *Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management and Research*, Wiley, New York, NY, pp. 1-4 (Chapter 1 – Introduction).

Boehm, B.W. (1984), "Software engineering economics", *IEEE Transactions on Software Engineering*, Vol. 10 No. 1, pp. 4-21.

Boehm, B.W. (1988a), "A spiral model of software development and enhancement", *IEEE Computer*, May, pp. 61-72.

Boehm, B.W. (1988b), "Understanding and controlling software costs", *IEEE Transactions on Software Engineering*, Vol. 14 No. 10, pp. 1462-77.

Boehm, B.W. (1991), "Software risk management: principles and practices", *IEEE Sofrware*, Vol. 8 No. 1, pp. 32-41.

Boehm, B.W. (2000), "Unifying software engineering and systems engineering", *Computer*, Vol. 33 No. 3, pp. 114-16.

Boehm, B.W. (2002), "Early experiences in software economics", in Broy, M. and Denert, E. (Eds), *Software Pioneers*, Springer, New York, NY, pp. 632-40.

Boehm, B.W. and Papaccio, P.N. (1988), "Understanding and controlling software costs", *IEEE Transactions on Software Engineering*, Vol. 14 No. 10, pp. 1462-77.

Boehm, B.W. and Ross, R. (1989), "Theory – W software project management: principles and examples", *IEEE Transactions on Software Engineering*, Vol. 15 No. 7, pp. 902-6.

Boehm, B.W. and Turner, R. (2004), *Balancing Agility and Discipline: A Guide for the Perplexed*, Addison-Wesley, Boston, MA.

Boehm, B.W., Egyed, A., Kwan, J., Port, D. and Madachy, R. (1998), "Using the WinWin spiral model: a case study", *IEEE Computer*, Vol. 31 No. 7, pp. 33-44.

Brooks, F.P. Jr (2007), "Software and systems management", in Selby, R.W. (Ed.), *Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management and Research*, Wiley, New York, NY, pp. 573-8 (Chapter 7 – Introduction).

Collyer, S. and Warren, C.N.J. (2008), "Project management approaches for dynamic environments", *International Journal of Project Management*, Vol. 27 No. 4, pp. 81-92.

Collyer, S. and Warren, C.M.J. (2009), "Project management approaches for dynamic environments", *International Journal of Project Management*, Vol. 27 No. 4, pp. 355-64.

Connell, J.L. and Shafer, L.B. (1989), *Structured Rapid Prototyping*, Prentice-Hall, Englewood Cliffs, NJ.

Cooper, R.G. (1994), "Perspective third-generation new product processes", *Journal of Innovation Management*, Vol. 11, pp. 3-14.

Deakins, E. and Dillon, S. (2005), "A helical model for managing innovative product and service initiatives", *International Journal of Project Management*, Vol. 23, pp. 65-74.

DeMarco, T. (2007), "Software risk management", in Selby, R.W. (Ed.), *Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management and Research*, Wiley, New York, NY, pp. 383-6 (Chapter 5 – Introduction).

Dichter, C. and Pease, M. (1995), *Software Engineering with Perl: Prototyping and Toolsmithing for Better Software-Sooner*, Prentice-Hall, Englewood Cliffs, NJ, pp. 22-36.

Edstedt, P. (2011), Project Manager/Scrum Master, Tieto, personal communication, April 11.

Gilb, T. (1988), *Principles of Software Engineering Management*, Addison-Westley, Reading MA.

Glass, R.L. (1998), *In the Beginning: Personal Recollections of Software Pioneers*, IEEE Computer Society, Los Alamitos, CA.

Hansen, S. and Rennecker, J. (2010), "Getting on the same page: collective hermeneutics in a systems development team", *Information and Organization*, Vol. 20 No. 1, pp. 44-63.

Hartson, H.R. and Hix, D. (1989), "Toward empirically derived methodologies and tools for human-computer interface development", *Man-Made Studies*, Vol. 31, pp. 477-94.

Hatley, D.J. and Pirbhai, I.A. (1988), *Structured Methods*, Dorset House Publishing Co., New York, NY.

Innovation.org (2007), *Drug Discovery and Development: Understanding the R&D Process*, (accessed January 13, 2012).

Khoo, E.T., Merritt, T. and Cheok, A.D. (2009), "Designing physical and social intergenerational family entertainment", *Interacting with Computers*, Vol. 21 Nos 1/2, pp. 76-87.

Kutsch, E., Hall, M. and Lee-Kelly, E. (2012), "Does risk matter? Disengagement from risk management practices in information systems projects", *European Journal of Information Systems* (in press).

Kutsch, E., Geraldi, J.G., Hall, M. and Kelly, L. (2009), "The (non-)adoption of risk management processes in projects", *Proceedings of the 9th Annual Conference of the European Academy of Management* (*EURAM*), *Liverpool, UK*.

Lenfle, S. and Loch, C. (2010), "How project management came to emphasize control over flexibility and novelty", *California Management Review*, Vol. 53 No. 1, pp. 32-55.

Lindgaard, G. (2009), "Early traces of usability as a science and as a profession", *Interacting with Computers*, Vol. 21 No. 3, pp. 350-2.

Lu, C.-C., Kang, S.-C., Shang, H.-H. and Shiu, R.-S. (2009), "Improvement of a computer-based surveyor-training tool using a user-centered approach", *Advanced Engineering Informatics*, Vol. 23 No. 1, pp. 81-92.

MacMillan, I.C., van Putten, A.B., McGrath, R.G. and Thompson, J.D. (2006), "Using real options discipline for highly uncertain technology investments", *Research-Technology Management*, January/February, pp. 29-37.

Mili, H., Mili, F. and Mili, A. (1995), "Reusing software: issues and research directions", *IEEE Transactions on Software Engineering*, Vol. 21 No. 6, pp. 528-62.

Naik, G. (2012), "New attack on Alzheimer's", *Wall Street Journal*, February 10, p. A3.

Nicholas, J.M. (2001), *Project Management for Business and Technology: Principles and Practice*, 2nd ed., Prentice-Hall, Upper Saddle River, NJ.

Nichols, N.A. (1994), "Scientific management at Merck: an interview with CFO Judy Lewent", *Harvard Business Review*, January/February, pp. 88-99.

Nilsson, A. (2008), "Project management practice: observations of work in short-duration projects" ("Projektledning i praktiken: Observationer av arbete i korta projekt"), dissertation, Umeå School of Business, Umeå.

Nissen, M.E. (1998), "Redesigning reengineering through measurement-driven inference", *MIS Quarterly*, Vol. 22 No. 4, pp. 509-34.

Oman, P.W. and Lewis, T.G. (1990), "A renaissance of paradigms challenging the norms", in Oman, P.W. and Lewis, T.G. (Eds), *Milestones in Software Evolution*, IEEE Computer Society, Los Alamitos, CA, p. 225.

Osterweil, L.J. (2007), "Software process: emerging extensions", in Selby, R.W. (Ed.), *Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management and Research*, Wiley, New York, NY, pp. 499-502 (Chapter 6 – Introduction).

Ould, M.A. (1990), *Strategies for Software Engineering: The Management of Risk and Quality*, Wiley, London.

Peters, T. (1992), *Liberation Management: Necessary Disorganization for the Nanosecond Nineties*, Alfred A. Knopf, New York, NY.

Pyster, A.B. (2007), "Software tools", in Selby, R.W. (Ed.), *Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management and Research*, Wiley, New York, NY, pp. 227-30 (Chapter 3 – Introduction).

Ramamoorthy, C.V., Prakish, A., Tsai, W.T. and Usuda, Y. (1984), "Software engineering: problems and perspectives", *IEEE Computer*, Vol. 17 No. 10, pp. 191-209.

Rising, L. and Janoff, N.S. (2000), "The Scrum software development process for small teams", *IEEE Software*, Vol. 17 No. 4, pp. 26-32.

Royce, W. (2007), "Early spiral model", in Selby, R.W. (Ed.), *Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management and Research*, Wiley, New York, NY, pp. 315-18 (Chapter 4 – Introduction).

Schreiber, G., Wielinga, B., DeHoog, R., Akkermans, H. and Vandevelde, W. (1984), "CommonKADS – a comprehensive methodology for KBS development", *IEEE Expert Intelligent Systems and Their Applications*, Vol. 9 No. 6, pp. 28-37.

Selby, R.W. (2007), "Software economics", in Selby, R.W. (Ed.), *Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management and Research*, Wiley, New York, NY, pp. 87-90 (Chapter 2 – Introduction).

Simon, H. (1996), *The Sciences of the Artificial*, 3rd ed., The MIT Press, Cambridge, MA.

Stacey, P. and Nandhakuman, J. (2009), "A temporal perspective of the computer game development process", *Info Systems Journal*, Vol. 19 No. 5, pp. 479-97.

Studer, R., Benjamins, R. and Fensel, D. (1998), "Knowledge engineering: principles and methods", *Data & Knowledge Engineering*, Vol. 25, pp. 161-97.

Sullivan, K.J. (2007), "Value-based software engineering", in Selby, R.W. (Ed.), *Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management and Research*, Wiley, New York, NY, pp. 731-6 (Chapter 9 – Introduction).

Swartout, W. and Balzer, R. (1982), "On the inevitable intertwining of specification and implementation", *Communications of the ACM*, Vol. 25 No. 7, pp. 438-45.

Van Putten, A.B. and MacMillan, I.C. (2004), "Making real options really work", *Harvard Business Review*, December, pp. 134-41.

Walfisz, M., Zackariasson, P. and Wilson, T.L. (2006), "Real-time strategy: evolutionary game development", *Business Horizons*, Vol. 49 No. 6, pp. 487-96.

Williams, L. (2010) in Zelkowitz, M.V. (Ed.), *Agile Software Development Methodologies and Practices*, Advances in Computers, Vol. 80, pp. 1-44.

Zackariasson, P. (2003), "Cyborg leadership: including nonhuman actors in leadership", licentiate thesis, Umeå School of Business, Umeå University, FE-publikationer 2003:172.

About the authors

Andreas Nilsson received his PhD in 2008 on a thesis about project management practice in a software development project. Currently he holds a post-doc position at Umeå School of Business (USBE) studying leadership and delegation. At USBE he also teaches, on campus and on-line, in the areas of leadership, project management, strategy and organization theory at basic and advanced levels. For more information see: usbe.se/ansnin95

Timothy L. Wilson is Adjunct Professor of Marketing and Management at the Umeå School of Business. His research interests are in the general areas of business services, project organizations and management, international business, and regional development. On an annual basis he offers a PhD course in Academic Writing and works with faculty and staff at USBE with their writing and publication efforts (for more information see: usbe.se/tiwi0002). Timothy L. Wilson is the corresponding author and can be contacted at: tim.wilson@usbe.umu.se