MANAGING COMPLEX SYSTEMS
ABOUT THE AUTHOR

Since 1989, Howard Eisner has served as Distinguished Research Professor and Professor of Engineering Management and Systems Engineering at The George Washington University, Washington, DC. For the prior thirty years, he held various positions in industry, including president of two systems and software engineering companies (Intercon Systems Corporation and the Atlantic Research Services Corporation) and as a member of the board of three companies. He is a Life Fellow of the Institute of Electrical and Electronics Engineers (IEEE) and a member of several engineering honor societies. Dr. Eisner has written two books on systems engineering and related topics and a book on personal and corporate reengineering. He holds a B.E.E. from the City College of New York, and M.S. from Columbia University, and a Doctor of Science from The George Washington University.
Contents

Preface ix

1 Systems and Thinking 1
  1.1 Building and Managing Complex Systems 2
  1.2 Some Results of Thinking Outside the Box 3
  1.3 Thinking in Relation to Specific Issues 4
  1.4 Conclusion 11
  References 11

2 Building and Managing Systems 13
  2.1 Some Basics of Systems Engineering 14
  2.2 Some Basics of Project Management 16
  2.3 Complex Systems 17
  2.4 System of Systems Engineering 21
  2.5 Summary 22
  References 22

3 Problems to Ponder 24
  3.1 Problem Areas: Systems 25
  3.2 Problem Areas: People 27
  3.3 Problem Areas: Software 29
  3.4 Problem Areas: Management 32
  3.5 Summary 34
  References 35
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 The Inventive Mind</td>
<td></td>
</tr>
<tr>
<td>4.1 Management Thinking</td>
<td>37</td>
</tr>
<tr>
<td>4.2 Scientific and Technical Thinking</td>
<td>42</td>
</tr>
<tr>
<td>4.3 Not Everyone Is an Einstein (Or Needs To Be)</td>
<td>44</td>
</tr>
<tr>
<td>4.4 The Next Nine Chapters</td>
<td>45</td>
</tr>
<tr>
<td>References</td>
<td>45</td>
</tr>
<tr>
<td>5 Perspective 1: Broaden and Generalize</td>
<td></td>
</tr>
<tr>
<td>5.1 Architecting a Complex System</td>
<td>47</td>
</tr>
<tr>
<td>5.2 Using Functional Decomposition to Explore Strategies</td>
<td>49</td>
</tr>
<tr>
<td>5.3 System of Systems Engineering</td>
<td>50</td>
</tr>
<tr>
<td>5.4 IBM’s View of the World</td>
<td>51</td>
</tr>
<tr>
<td>5.5 Haloid’s Bold Steps</td>
<td>52</td>
</tr>
<tr>
<td>5.6 From PERT to GERT to Stochastic Networks</td>
<td>53</td>
</tr>
<tr>
<td>5.7 Reinventing Yourself</td>
<td>53</td>
</tr>
<tr>
<td>5.8 Summary: A Meeting</td>
<td>54</td>
</tr>
<tr>
<td>References</td>
<td>55</td>
</tr>
<tr>
<td>6 Perspective 2: Crossover</td>
<td></td>
</tr>
<tr>
<td>6.1 Peoplesoft</td>
<td>58</td>
</tr>
<tr>
<td>6.2 Software Reuse</td>
<td>58</td>
</tr>
<tr>
<td>6.3 Developer Off-the-Shelf Systems</td>
<td>60</td>
</tr>
<tr>
<td>6.4 Accounting Firm Consultation</td>
<td>61</td>
</tr>
<tr>
<td>6.5 Building and Managing New Systems</td>
<td>62</td>
</tr>
<tr>
<td>6.6 Summary: A Managing New Systems</td>
<td>70</td>
</tr>
<tr>
<td>References</td>
<td>72</td>
</tr>
<tr>
<td>7 Perspective 3: Question Conventional Wisdom</td>
<td></td>
</tr>
<tr>
<td>7.1 Large and Complex Government Systems</td>
<td>75</td>
</tr>
<tr>
<td>7.2 Conventional Wisdom Changes with the Times</td>
<td>78</td>
</tr>
<tr>
<td>7.3 More Challengeable Conventional Wisdom</td>
<td>81</td>
</tr>
<tr>
<td>7.4 Summary: Two Meetings</td>
<td>82</td>
</tr>
<tr>
<td>References</td>
<td>83</td>
</tr>
<tr>
<td>8 Perspective 4: Back of the Envelope</td>
<td></td>
</tr>
<tr>
<td>8.1 What Can Fit on the Back of an Envelope?</td>
<td>86</td>
</tr>
<tr>
<td>8.2 Some Examples of Back-of-the-Envelope Results</td>
<td>86</td>
</tr>
<tr>
<td>8.3 Constructing What’s on the Back of the Envelope</td>
<td>92</td>
</tr>
<tr>
<td>8.4 What Does It All Mean?</td>
<td>95</td>
</tr>
<tr>
<td>8.5 Summary: A Meeting</td>
<td>95</td>
</tr>
<tr>
<td>References</td>
<td>96</td>
</tr>
</tbody>
</table>
## CONTENTS

13.3 The Persistence of Alternatives  
13.4 Building a System  
13.5 Architecting a System  
13.6 Additional Views of Architectures  
13.7 Another Government Perspective  
13.8 Summary: A Meeting  
References

14 Thinking in Groups  
14.1 The Delphi Process  
14.2 Groupthink  
14.3 de Bono and His Thinking Hats  
14.4 Advocacy vs. Inquiry  
14.5 SAST  
14.6 Team Syntegrity  
14.7 Facilitation  
14.8 Self-Directed Work Teams  
14.9 Synectics  
14.10 No More Teams: Collaboration  
14.11 Conclusion  
References

15 Widening the Circle  
15.1 de Bono and Lateral Thinking  
15.2 TRIZ  
15.3 Thinking Like Leonardo  
15.4 The Art of Problem Solving  
15.5 Einstein  
15.6 Breakthrough Thinking  
15.7 Creativity  
15.8 Cogito Ergo Sum  
15.9 Thinking About the Future  
15.10 Thinking for a Change  
15.11 Some Genius Attributes  
15.12 Systems Thinking  
References

16 Final Thoughts and a Test  
16.1 More Extraordinary Thinkers  
16.2 Is There Life Outside the Box?  
16.3 Test Yourself  
16.4 Final Words  

Index
Preface

This book is about two primary topics: systems and ways of thinking. We are part of and use several systems just about every day: from the highway system, to the telecommunication system, to the banking system, to the medical system, to many others. This central fact supports the notion that it is important to better understand how to build and manage such systems. Improvements can pay large dividends in all our lives and provide benefits measurable in economic as well as quality-of-life terms.

Such improvements are not, however, easy to make. Even if you are inside a company making a special product, it is often a long road from thinking about a product improvement to getting it out the door to your customers. The realities of how we conceive of and actually make new or improved products often conspire against us. So a large number of people in various companies and parts of government have put out the challenge for themselves and their associates—let’s see if we can find a way to “think outside the box.” This book is partly a response to that challenge.

The specific story of this book started some years ago when I was asked to teach a course in systems engineering at a relatively large systems integration and engineering company. About 30 employees were sponsored and made up a cohort that was being given this out-of-the-ordinary treatment. The course was a normal offering as the first course in systems engineering and part of my overall responsibility in the Engineering Management and Systems Engineering Department in the School of Engineering and Applied Science at The George Washington University. As lead professor in systems engineering, I decided to visit with the company manager who wanted this special program for his people. I asked him what he was looking for in the course and its delivery. He gave two answers. Since systems engineering was a core competency in the company’s ability to do systems integration, he wanted to make sure that his people had a formal knowledge of systems engineering, along with some practice in execution. I more-or-less
expected that answer, but then he proceeded to a second answer: “I want my people to be better able to think outside the box,” he declared. I took that as a very serious answer, and it motivated me to dig more deeply into its various meanings.

My several-year research activity into what it might mean to think outside the box included a reexamination of my experiences over about a 45-year period. During the first 30 of these years I worked in industry, starting as a research engineer and finishing as the president of two high-tech companies. For the past 16 years I have been a professor in academia, where I served as an educator, researcher, and consultant to several companies and groups. Reviewing these experiences was enormously rewarding as well as lots of fun. It’s not often that one gets a chance to review so many years of work with a focused objective in mind. A key question was: Could I identify specific cases where I concluded that I might be thinking outside the box, as well as to the contrary?

Along with the somewhat personal retrospective noted above, I also scoured the literature and found it expansive. Relevant material was everywhere and took me through a time line from da Vinci’s era to the twenty-first century in terms of both technical and management thinkers. A few even took me back to our very earliest days as, for example, I contemplated what went wrong as some tried to build the Tower of Babel. I’m happy to report that we seem to have been able to do a lot better since that time.

As I was bringing together and organizing my personal experiences along with examples from our history and the literature, a list of some dozen ways to think outside the box emerged. All had at least some foundation in my own experience since I was able to connect to that perspective in a deeper way. In 1999 and 2000 I wrote two articles for the Washington Business Journal on the matter of thinking outside the box. Each article described briefly five ways to do this type of thinking. The reader response to these articles was about 10 times the response I typically got from my other articles on the “technology manager.” This was additional encouragement; people certainly were very interested in this subject.

I followed that with a talk on this topic as the invited lecturer at an evening professional society meeting and dinner. Here again, a lot of interest was displayed, as I had the additional opportunity of a Q & A session as part of the evening activity. All of this convinced me that it was time to think seriously about how to write a book that included these experiences and my newly found set of expanded ways of thinking. The key to moving forward was to link the matter of thinking outside the box with my strong interest in systems and how we might be able to improve how we build and manage systems. I also wanted the book to be as useful as possible, sacrificing much of the academic perspective in favor of real-world practicality. Time will tell whether or not that goal has been achieved.

As to the particulars, this book is organized into 16 chapters. In the first four chapters we explore such topics as the nature of large-scale systems and complexity, thinking inside and outside the box with respect to specific systems issues, some basics of systems engineering and project management, system of systems engineering, typical systems problems faced by management, and examples of the inventive mind in both management as well as scientific domains.
Chapters 5 through 13 then focus on nine specific ways to think outside the box. Examples are used to a large extent to show how these thinking approaches have led to better solutions in the past. In this context, such solutions tend to contribute to improvements in how we build and manage systems of various types.

In Chapter 14, ways of thinking are expanded specifically to group situations. Such scenarios can be enhancers or inhibitors, depending on the group dynamics. Of particular interest is how to achieve the former and avoid the latter.

Chapter 15 widens the circle by presenting approaches that have been suggested by others to expand one’s ways of thinking. These are not included as one or another of the main nine perspectives in Chapters 5 through 13 since the latter are derivative of more personal and direct experiences of mine. References are provided so that readers who wish to explore other approaches may do so.

Chapter 16 is the concluding chapter. It reiterates the main points and perspectives with respect to building and managing complex systems and thinking outside the box. It also includes a test for readers to obtain insights into whether or not they are ready to embrace thinking outside the box as well as some of the consequences of doing so.

Having looked at all of the above, a few words are still in order about who this book might be for. Several audiences are envisioned:

1. People with management responsibilities of various types with respect to building and managing large-scale complex systems
2. Nonmanagers trying to contribute to the building of systems, who are looking for new ways to approach the problems they encounter
3. People who are simply seeking to expand their patterns of thinking in any domain of their lives
4. University students who may be able, relatively early in their careers, to master new thinking perspectives that will help them be more successful in school and after they graduate

Relating to item 4 above, a standard 15-week university course may be constructed as follows:

<table>
<thead>
<tr>
<th>Class Sessions</th>
<th>Book Chapters</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 and 2</td>
</tr>
<tr>
<td>2</td>
<td>3 and 4</td>
</tr>
<tr>
<td>3 through 11</td>
<td>5 through 13</td>
</tr>
<tr>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>13</td>
<td>15</td>
</tr>
<tr>
<td>14</td>
<td>16</td>
</tr>
<tr>
<td>15</td>
<td>Final exam</td>
</tr>
</tbody>
</table>
At a personal level, I wish to dedicate this book to four people. The first is my wife, June Linowitz, who continues to be supportive of my various writing endeavors and who, without much apparent effort, exhibits prodigious skills at thinking, both inside and outside the box. The other three people are my daughter, Susan Rachel Lee, and my sons, Seth Eric and Oren David. All are in their 40s and display considerable wisdom in conducting their lives. And they are passing this on to their children, my grandchildren, who are Gabriel, Jacob, Rebecca, Benjamin, and Zachary. The wheel continues to turn. Thank goodness it is (mostly) round rather than multipolygonic. There are still some bumps in the road.

Howard Eisner

Bethesda, Maryland
The main focus of this book is to define and explore ways of thinking that have been shown to have a positive impact on building and managing complex systems. This chapter is largely introductory, defining some basic aspects of systems as well as issues that relate to the construction and management of these systems. Some alternative approaches to thinking about these issues are also explored.

Complex systems will be interpreted very broadly and will include both physical (mostly hardware and software) groupings of equipment to serve a purpose and sets of procedures that are carried out by people or machines, or both. An example of the former is our national air traffic control (ATC) system that is used to control our aircraft as they fly across the country as well as in and out of terminal areas. For the latter, one might think of an airline reservation system, which is largely procedural but is executed on a large scale by both people and machines. Complex systems tend to be relatively large, with lots of internal and external interfaces. Part of our job in thinking about such systems is to try to reduce the complexity by focusing on fundamentals and simplification. But that is getting a bit ahead of ourselves.

Very large and complex systems include, as examples, the following:

- Our national aviation system
- Our national telephone system
- Our stock market system
- Our legislative system, the means by which we enact our laws
Our electricity delivery system
Our interstate highway system

Smaller, but still seriously complex systems could be represented by:

- Our traffic control systems, which control the flow of airplanes and automobiles
- The aircraft and automobiles themselves
- Information systems of various kinds, such as Microsoft Office or a company’s inventory control system

It is important to note that some systems are related directly to a special or novel idea and that such systems would probably not exist without that idea. For example, when you send a FedEx package from the east to the west coast, you are using a large and complex system (mostly transparent to you) of airplanes, trucks, control systems, tracking systems, delivery personnel, and others. The fact that all packages go to Memphis, Tennessee, overnight and fan out from there is a fact of the design of that system. That design is based on the singular idea that this approach is efficient and cost-effective. Without that idea, another system configuration might have been selected. In other words, for many systems, it is the breakthrough idea that is behind the system, and as such, places us in search of these types of ideas as we approach the design issue. In this book, these types of ideas represent thinking outside the box.

1.1 BUILDING AND MANAGING COMPLEX SYSTEMS

We have a variety of tools that we use to perform the tasks of building and managing complex systems. In the building part, a set of activities known as systems engineering is dominant, giving us guidance on how to construct a cost-effective system [1.1–1.4]. A crucial part of that process is the architecting of the system, which is a top-level structure for the system [1.4, 1.5]. For example, the personal computer today generally has an open architecture, such that there is considerable interoperability between components. The same is true, in the main, for other consumer electronics, such as TV sets, VCRs, DVD players, and stereo (audio) systems. Unfortunately, there are lots of system architectures that do not adequately consider interoperability matters.

For the managing function with respect to complex systems, all of our institutions study and implement the best approaches they can think of in terms of management practices. There is an extensive literature that is available to help, from Jack Welch in regard to how he managed GE [1.6], to Harold Geneen and how he built ITT [1.7]. And there is no dearth of “gurus” such as Peter Drucker [1.8], Tom Peters [1.9], and W. Edwards Deming [1.10], who try to point us in the right direction with respect to the fine art of management. In
relation to the field of project management, even I have joined the fray [1.4, Chaps. 3–6].

In both building and managing these systems, we are constantly in search of the new and better idea that will allow us to make critical improvements in their design and operation. The new and seminal idea is the “Holy Grail” that takes us to the next level, so that we may be in a position to do better than our competitors. Our overall industrial system works that way, and is supported in part by an elaborate system that is an example of how we value the new and better approach: our patent system. Without that system for recording and regulating better ideas, it would probably be a free-for-all in industry and a sure road to disaster. In other words, we continue to seek, keep track of, and value the better idea, referred to in this book as what results from thinking outside the box.

1.2 SOME RESULTS OF THINKING OUTSIDE THE BOX

In very specific terms, it is clear that thinking outside the box often leads to a new system that is better than previous systems. This can be illustrated by the following short list of inventions and better approaches that all will recognize:

- Light bulb
- Airplane
- Transistor
- Electronic chip
- Copying machine (i.e., xerography)
- Digital computer, from personal computer to supercomputer
- Atomic bomb
- Internal combustion engine
- Artificial heart
- Nuclear power plant

Although it is the new and seminal idea that sets the stage for major advances in building and managing systems, such an idea is often insufficient. Usually, there are many hurdles to jump over and potholes to avoid between the thought and its implementation. Bringing an idea from its conception to its fruition in the real world is often a long journey that requires lots of determination and stick-to-tiveness. A relevant observation in this regard is that genius is a matter of 1% inspiration and 99% perspiration (attributed to Thomas Edison). Although modest gains, one day at a time, can be made just by showing up, breakthrough gains don’t come that easily. So even if you have a demonstrably wonderful new idea, you must keep in mind that you’re still only a short way down the road toward the goal of converting the idea into a successful product or service that is implemented in the real world.
1.3 THINKING IN RELATION TO SPECIFIC ISSUES

Table 1.1 lists a dozen issues together with ways of thinking about these issues that might be considered both inside and outside the box. These are explored next, one issue at a time. They are meant simply to illustrate, in a concrete way, how thinking inside and outside the box may differ from one another.

### Table 1.1 Examples of Thinking Inside and Outside the Box

<table>
<thead>
<tr>
<th>Issue</th>
<th>Thinking Inside the Box</th>
<th>Thinking Outside the Box</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Integration of stovepipes</td>
<td>100% of all systems must be integrated</td>
<td>Integrate what it is cost-effective to integrate</td>
</tr>
<tr>
<td>2. Best of best of breed</td>
<td>Optimizing subsystem choices will optimize the overall system</td>
<td>May not work; there is no guarantee</td>
</tr>
<tr>
<td>3. Measurements</td>
<td>Measure as much as you can think of</td>
<td>Measure a minimum set that works and “tells the story”</td>
</tr>
<tr>
<td>4. Getting back on schedule</td>
<td>Add more people on the project</td>
<td>Adding more people is likely to make situation worse</td>
</tr>
<tr>
<td>5. Requirements change and volatility</td>
<td>Requirements are to be taken as fixed and inviolate</td>
<td>Requirements can, at times, be variables</td>
</tr>
<tr>
<td>6. Reserves on a project</td>
<td>All levels of management need to have dollar reserves</td>
<td>Project manager needs enough money to get the job done</td>
</tr>
<tr>
<td>7. Customer negotiation</td>
<td>Promise whatever the customer appears to want</td>
<td>Try to underpromise and overdeliver</td>
</tr>
<tr>
<td>8. Dealing with customers</td>
<td>The customer is always right</td>
<td>The customer can often be wrong</td>
</tr>
<tr>
<td>9. Overall approach</td>
<td>Do it right the first time (DIRFT) [1.11]</td>
<td>Provide continuous improvement and iteration</td>
</tr>
<tr>
<td>10. Employee trust</td>
<td>Employees cannot be trusted to know how the company is really doing</td>
<td>Have the obligation to tell the truth and focus on company well-being</td>
</tr>
<tr>
<td>11. Work task strategy</td>
<td>Never do work unless you can profit from it</td>
<td>Invest in key areas for the future health of the company</td>
</tr>
<tr>
<td>12. Processes and products</td>
<td>Get the process right and the products will always be right</td>
<td>The right process still doesn’t guarantee the right product</td>
</tr>
</tbody>
</table>

### 1.3.1 Integration of Stovepipe Systems

There are a large number of “stovepipe” systems in operation today, each of which carries out a discrete function. Examples include a human resources personnel tracking system, an inventory control system, and a finance and accounting system. When a new manager comes upon such a scene, he or she often leads the charge toward the integration of the stovepipes, believing that an integrated system is certainly going to be more cost-effective. The knee-jerk goal is often something like the following: We need to integrate the stovepipes to the maximum extent,
approaching 100% integration. One might say that this knee-jerk reaction to the existence of a set of stovepipes is the rule today rather than the exception.

A deeper look, however, suggests that the integration of stovepipes may be a good idea, but it also may be a bad idea. Further, the so-called goal of 100% integration may be a very bad idea. An out-of-the-box approach might well be to integrate the stovepipes to whatever extent is appropriate and cost-effective, based on the specific circumstances attendant on the stovepipes in question. This author has seen an advanced Navy information system start down the road toward the integration of stovepipes and then be given up after three years, at which time they proceeded to “disintegrate” the stovepipes. The problem, as defined, was just too difficult and expensive.

More will be said about this particular out-of-the-box perspective later in this book. For the time being, can you think of situations for which the integration of stovepipes might well be a less than wonderful idea? If so, make a note of your thoughts and hold on to them until you read the next section.

1.3.2 Best of “Best of Breed”

Inside-the-box thinking assumes that a system composed of the integration of a set of best-of-breed systems will necessarily be the best system that can be constructed. After all, what can be better than the “sum” of “bests”? A simple example should demonstrate that such a system can be a terrible choice in the sense that another approach can easily be seen to be superior.

Let us assume that a government agency has done several in-depth studies to determine the best-of-breed systems that carry out certain functions. The functions and the best of breed answers are as follows:

<table>
<thead>
<tr>
<th>Functions</th>
<th>Best of Breed Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word processing</td>
<td>Wordperfect</td>
</tr>
<tr>
<td>Spreadsheet</td>
<td>Lotus 1-2-3</td>
</tr>
<tr>
<td>Presentation manager</td>
<td>Powerpoint</td>
</tr>
<tr>
<td>Database management system</td>
<td>Oracle</td>
</tr>
</tbody>
</table>

It should be noted that each of these best-of-breed systems is produced by a different company:

- Wordperfect is made by Corel.
- Lotus 1-2-3 is made by Lotus.
- Powerpoint is made by Microsoft.
- Oracle is made by Oracle.

Anyone familiar with software products recognizes the difficulty of trying to integrate software from different companies, even if the source code were accessible and modifiable. The costs of doing so would be prohibitively high,
especially in light of the fact that another, more than feasible alternative is readily available. Those familiar with Microsoft’s Office system (or Lotus’s Smartsuite) can see immediately that an integrated system with the four functions above is available as a commercial off-the-shelf (COTS) product. Thus, abandoning the integration of disparate systems in favor of Microsoft Office is a clear and highly cost-effective alternative solution. This simple example demonstrates that leaping to the conclusion that a system composed of a set of best-of-breed systems will be the best choice may not be the right answer, even though one’s intuition might point in that direction.

1.3.3 Measurements

This example focuses on measurements programs that are related to the building or managing of a product or service. Measurements are also sometimes called metrics. The inside-the-box approach is to measure everything that you can think of. That way you are not likely to miss anything important, one can reason. As an example, the Department of Defense, in relation to establishing management indicators for software, came up with the following measurement areas [1.12]:

1. Requirements volatility
2. Software size
3. Software staffing
4. Software complexity
5. Software progress
6. Problem change report status
7. Build/release content
8. Computer hardware resource utilization
9. Milestone performance
10. Scap/rewrok
11. Effect of reuse

Looking at this list, it is evident (1) that lots of measurements are being made, and (2) that it will take a lot of effort to make such measurements. The size and efficacy of a measurements program are, of course, related to the size of the system being developed, but still, the list of measurements above suggests a considerable effort at a very sizable cost. Outside-the-box thinking tries to focus on the minimum measurement profile that is sufficient to “tell the story” and be implementable within the constraints of a real-world situation. That includes the practicality of both making the measurements and making changes, given that the measurements suggest that there are problem areas.

This “minimalist” approach is supported, if you will, by some of the thinking in the ubiquitous and far-reaching Department of Defense. In the so-called 5000 series [1.13] dealing with the acquisition of systems, the directive and instruction of the
reference are both scaled down, pointing to performance and capabilities-based acquisitions that are tailored to the situation at hand. I tend to agree with this idea for most situations, reckoning that there should be an appropriate match between a measurement program and the specific needs of that program rather than going automatically to the default solution that says: “Measure everything that you can think of.” A good measurements program should be designed rather than be the consequence of constructing a mindless, long laundry list.

1.3.4 Getting Back on Schedule

Many large-scale software development projects seem to go off the track after awhile, as manifested by falling behind schedule. This means that the actual achievement of various milestones is later than that called for by the original schedule. One inside-the-box reaction is to add personnel to the project in an attempt to get back on schedule. This reaction flies in the face of Brooks’s law [1.14], which states that the addition of personnel will probably make the project schedule problem even worse rather than better. Part of the reason for this apparent anomaly is that productive people currently working on the project will have to stop what they’re doing in order to bring the new people “up to speed.” This diversion of their time and attention makes the schedule problem worse, among other negative effects.

Is there a potentially better solution to this type of problem? A few other approaches suggest themselves, including (1) changing the technical approach, which may not be sound; (2) recognizing that the present personnel level may not be sufficient in terms of the technical challenge and its degree of difficulty; and (3) the original schedule may not be realistic.

1.3.5 Requirements Change and Volatility

Inside-the-box thinking assumes that the up-front set of defined requirements is basically inviolate and “set in concrete.” This point of view often follows from the need to be definitive in a contract document between customer and contractor. However, since requirements are usually defined very early in a program, it makes sense that they will probably have defects, often significant ones. Out-of-the-box thinking recognizes the fallibility of the people who develop this early statement of requirements and provides the means by which requirements can be changed in an organized and disciplined manner. This is not the same as requirements creep. Sensible changes to requirements should be made when both parties agree that such changes will improve the system development process or the product.

Support for the above can be found in both the “spiral” approach to building systems [1.15] and in aspects of the acquisition process in the government. With respect to the latter, we find the following phrase: “consistent and continuous definition of requirements.” As systems are built, they are acquired in increments.
This incremental approach suggests further that requirements may change as part of the process, for purposes of refinement and improvement.

1.3.6 Reserves on a Project

Conventional wisdom might suggest, in the execution of a project, that each layer of management set aside a reserve to try to assure project success. That’s an interesting mainstream idea but it can easily lead to counterproductive results. Suppose that a project manager (PM) reports to a program manager who reports to a division director who reports to a vice president. If the latter three executives each set aside a 10% dollar reserve, we can see that the PM is left with only about 73% of the original budget to work with. As a conscientious PM, he or she will then attempt to complete the work with about three-fourths of the original estimate of funding. This, in turn, can place enormous stress on the PM as well as on every member of the team over the entire duration of the project. This can easily lead to several negative consequences.

First, let us assume that by lots of extra hours and good PM management, the project is completed without use of the reserves. As the people on the project discover that all those extra hours were likely not to have been essential, they will conclude that they were taken advantage of, unless they get a bonus for the work. If not, a lasting negative feeling will persist.

Now assume that the project is not completed: that is, the PM runs out of money prior to completion. The reserves may or may not have been employed in time to preclude schedule and technical progress slippages. If not, a failure scenario has taken place, even though that was not the intention. The bottom line? Don’t give the PM a very nearly impossible job to do and try to be a hero by making new funds available later. It’s a strategy that has a good chance of backfiring. Instead, give the PM all the money necessary to get the job done.

1.3.7 Customer Negotiation

There are literally hundreds of books that can be consulted about how to negotiate with your customer, who is looking to you, and possibly others, to build and manage a system for them. Conventional wisdom appears to tilt in the direction of promising essentially whatever the customer wants. It is easy to see that this strategy can often lead both parties astray.

An alternative out-of-the-box approach was suggested to me by my oldest son. He had been quite successful as a software development manager at a large and significant company. One day I asked him to cite the most important reason for his success. His answer was: “Try to underpromise and overdeliver.” This meant relatively hard-nosed up-front negotiation, but it paid handsome dividends to both parties down the road. At project completion, everyone tended to be pleased. Another way of expressing this approach is simply to manage expectations. Many managers work themselves and their people to a frazzle and still