SEI SERIES IN SOFTWARE ENGINEERING

Creating and Using Virtual Prototyping Software

Principles and Practices

Douglass E. Post

Richard P. Kendall

Creating and Using Virtual Prototyping Software

This page intentionally left blank

Creating and Using Virtual Prototyping Software

Principles and Practices

Douglass E. Post
Richard P. Kendall

Addison-Wesley

Boston • Columbus • New York • San Francisco • Amsterdam • Cape Town
Dubai • London • Madrid • Milan • Munich • Paris • Montreal • Toronto • Delhi • Mexico City
São Paulo • Sydney • Hong Kong • Seoul • Singapore • Taipei • Tokyo

Cover credits:
Helicopter: KateChris/Shutterstock
Cruise ship: nan728/123RF
Concorde airplane: agsaz/Shutterstock
Passenger plane: iurii/Shutterstock
Sailing yacht: Alvov/Shutterstock
Race car: Scott Betts/123RF

Visit us on the Web: informit.com/aw.

Library of Congress Control Number: 2021947676

ISBN-13: 978-0-13-656693-9
ISBN-10: 0-13-656693-6

ScoutAutomatedPrintCode

Editor-in-Chief
Mark Taub

Executive Editor
Haze Humbert

Development Editor
Mark Taber

Managing Editor
Sandra Schroeder

Senior Project Editor
Lori Lyons

Copy Editor
Krista Hansing
Editorial Services

Production Manager
Remya Divakaran/
Codemantra

Indexer
Timothy Wright

Proofreader
Abigail Manheim

Compositor
Codemantra

Pearson's Commitment to Diversity, Equity, and Inclusion

Pearson is dedicated to creating bias-free content that reflects the diversity of all learners. We embrace the many dimensions of diversity, including but not limited to race, ethnicity, gender, socioeconomic status, ability, age, sexual orientation, and religious or political beliefs.

Education is a powerful force for equity and change in our world. It has the potential to deliver opportunities that improve lives and enable economic mobility. As we work with authors to create content for every product and service, we acknowledge our responsibility to demonstrate inclusivity and incorporate diverse scholarship so that everyone can achieve their potential through learning. As the world's leading learning company, we have a duty to help drive change and live up to our purpose to help more people create a better life for themselves and to create a better world.

Our ambition is to purposefully contribute to a world where:

- Everyone has an equitable and lifelong opportunity to succeed through learning.
- Our educational products and services are inclusive and represent the rich diversity of learners.
- Our educational content accurately reflects the histories and experiences of the learners we serve.
- Our educational content prompts deeper discussions with learners and motivates them to expand their own learning (and worldview).

While we work hard to present unbiased content, we want to hear from you about any concerns or needs with this Pearson product so that we can investigate and address them.

- Please contact us with concerns about any potential bias at https://www.pearson.com/report-bias.html.

To our wives, Susan Post and
Linda Richards

Contents at a Glance

Contents

This page intentionally left blank

Figure List

This page intentionally left blank

Preface

Why This Book?

The purpose of our book is to provide guidance for engineers and scientists who want to acquire or develop software to improve the competitiveness of their engineering and scientific research organizations with virtual prototypes. Virtual prototypes replace physical prototypes in the product development process. They are an extension of computer-aided design that allows product developers not just to visualize product designs, but to accurately *predict* their performance, using physics-based software tools, before anything is manufactured. When applied to natural systems such as the weather, they allow forecasts of behavior to a level of accuracy never before possible.

Engineering design and scientific research have always involved working with abstractions of new products or the objectives of research. For the development of complex products, the abstraction may start with a mental image of the new product that can be then translated into a drawing or physical model. For scientific research, the abstraction may be an expectation of the result of an experiment or an observation of natural phenomena. As science and technology have progressed from ancient times to the present, the usefulness of these abstractions has grown immensely.

This progress has accelerated dramatically during the last 50 years or so. Computing power has grown explosively from 1 floating point operation per second (FLOP) at the end of World War II to 10^{17} FLOPS today. The consequences of this are almost as dramatic as the stunning increase in computing power. It is now possible to use computers to design and accurately predict the behavior of complex products such as supersonic jet aircraft and to accurately forecast the behavior of complex natural phenomena such as the weather.

Although computers have played a role in product design for decades, especially in microelectronics, current engineering design methods still rely largely on an experiment-based "design, build, test" paradigm. Scientific and engineering research is based on theoretical studies, physical models, and experiments, from tabletop scale to very large experimental facilities such as high-energy accelerators and large terrestrial and satellite-based telescopes. To improve the competitiveness of their organizations, engineers and scientists are increasingly turning to computational methods to analyze and predict the performance of new products and

conduct scientific research. This has become substantially more useful due to the introduction of multiphysics modeling software coupled with high-performance computing. The union of these two in a virtual prototype makes accurate predictions of full-scale system performance possible. For many applications, this was not the case even at the beginning of this century. This computational product development paradigm is much faster, much less expensive, and much more flexible than experiment-based methods.

The computational science and engineering literature contains references to virtual prototypes, digital surrogates, digital twins, mirrors, simulations, and various synonyms. The concepts behind these terms are roughly equivalent. We use them to refer to physics-based, mathematical representations (often referred to as *models*) of physical objects or natural physical systems captured in a digital form that can help predict their behavior or state. We use all these terms in this book, depending on the context. For example, virtual prototyping is the process of developing virtual prototypes and using them to investigate systems of interest. A digital surrogate is a computer model of a specific product or natural system, just as is a virtual prototype. The term *digital surrogate* often connotes more persistence than the term *virtual prototype*. A digital twin is a digital surrogate tied to a specific instance of a product or system of interest; it persists throughout the life of the product or system. Although we don't discuss it in detail, others have extended these concepts to include biological systems, including human societies, collections of microorganisms, and predator–prey behavior in natural ecosystems.

Historical Perspective

Although the advent of practical electronic computers occurred only recently (since World War II), the use of mathematical abstractions of systems for design and prediction has a long and distinguished history dating as far back as the Babylonian astronomers (800–400 BCE).

Babylonian astronomers used empirically based mathematical models and their 400 years of astronomical measurements of the motions of the moon and planets to predict astronomical events such as eclipses and planetary motion. Similar to modern digital surrogates, their "model" was validated by their data. However, unlike the Greeks, the Babylonians had no concept of a general model for the heavens. Their approach was purely an empirical mathematical exercise to predict eclipses, phases of the moon, and other events important for their astrology. They believed, as did many ancient peoples, that events in the heavens were clues from the gods that foretold future earthly events.

The Greek astronomer Claudius Ptolemaeus (Ptolemy), in Alexandria, Egypt (150–170 CE), developed a geocentric model of the heavens. It is one of the most familiar examples of an ancient virtual prototype: *virtual* in the dictionary sense of capturing the essence but not the appearance of the system. Most dictionaries now also list *digital* as an alternative meaning of *virtual*. Ptolemy's model is described in his manuscript, *The Almagest*. In the geocentric Ptolemaic model, the spherical Earth is motionless and the fixed stars, planets, moon, and sun revolve around Earth in various complicated orbits (epicycles). This was a very sophisticated model—and the most accurate mathematical model for the heavens for its time. As planetary and astronomical observations improved from the 1st century to the 16th century, the geocentric Ptolemaic model became increasingly unwieldy. However, it lasted 1,450 years, until the Copernican Revolution (circa 1550 CE), when Nicholas Copernicus published his heliocentric model for the heavens. His successors (Galileo, Kepler, Newton, Gauss, and others) ignited the scientific revolution. Newton was among the first to appreciate that the laws of physics were universal. Newton applied his three laws of motion and the law of gravity to the heavens and calculus (which he invented for this purpose) to predict the motion of the planets.

Over the next several centuries, other mathematicians and scientists developed physics-based, mathematical models to explain electromagnetics, fluid flow, and myriad other physical phenomena. The predictions of these models were initially calculated by hand, which severely limited their accuracy and scope. To help with his planetary orbit calculations, Gauss memorized a four-place log table. Until computers came along, calculating problems with realistic geometries and materials was extremely difficult for mortals who were less mentally endowed.

Although it developed a little later than the methods to calculate planetary motion, the mathematics of weather prediction were worked out reasonably well by 1910. Highly accurate practical weather predictions required computers that emerged only in the latter half of the 20th century. Virtual prototyping in the digital sense played a significant role in the Manhattan Project and a major role in the hydrogen bomb project. Both were (and remain) major drivers of the development of high-performance computing. Virtual prototyping has continued to play a major role in the U.S. nuclear weapons community, especially now that the U.S. is no longer conducting nuclear tests.

Computers became available to the general scientific and engineering community by the late 1950s. Computational engineering and computational science then started to grow rapidly as the advantages of virtual prototypes became apparent. Today the individual components of computational science (such as finite element-based fluid flow solvers) and computational engineering software tools are relatively mature. A major remaining frontier is the development and deployment of large-scale, multiphysics research and design tools, which can include *all* the important physical effects, together with realistic geometries and materials.

The Key Role of Software

Software applications are an essential part of this capability. They are in many ways the key part, despite being the least visible. At least for now, computers are general-purpose machines that can be used for handling accounting, creating animated movies, or predicting the weather. Although high-performance computers are far from simple, the needed computing power can be purchased from many cloud vendors. Neither they nor the networks to reach them need to be owned. Computing has thus become a commodity. For organizations that want to own and operate their own computing facilities, a plethora of vendors specialize in setting up and supporting the operation of computer centers. Many good engineering and scientific software packages are available from commercial vendors and other sources, yet the software needs for many applications are unique and not available from the marketplace.

This is especially true for the software that makes the virtual prototyping paradigm work. This software is not yet a commodity, especially at the multiscale, multiphysics, high-performance computing, system-of-systems level. We were motivated to write this book because it is potentially difficult to acquire and use the right software tools to successfully implement the virtual prototyping paradigm. In the following chapters, we share our experiences and lessons learned from initiating and executing the Computational Research and Engineering Tools and Environments (HPCMP CREATE, or just CREATE) program. One of the authors (Post) initiated and led CREATE for its first 12 years; the other (Kendall) has been a senior member of the CREATE team since 2007. CREATE was formed in 2006 by the U.S. Department of Defense (DoD) High Performance Computing Modernization Program (HPCMP) to introduce the virtual prototyping paradigm to the DoD acquisition community. The goal for CREATE was to develop, deploy, and sustain a suite of 11 physics-based, high-performance computing software tools for the development of digital surrogates for DoD air vehicles, naval vessels, ground vehicles, and radar antennas. A secondary goal was to provide a model for how the DoD high-performance computing community could develop its own multiphysics software for engineering design.

Our insights are captured here in a set of software engineering and software program management practices and principles that we found to be successful in CREATE. They are also tempered by our experiences from our 50-year careers in computational engineering and computational science in the U.S. DoD, the U.S. Department of Energy (DOE), and other federal agencies; U.S. industry; and academia. With colleagues, we have made formal and informal assessments and case studies of both successful and unsuccessful programs to develop and deploy physics-based software for product development and scientific research in areas such as weather prediction, computational chemistry, atomic and molecular physics, oil

and gas production, earthquakes, astrophysics, cosmology, plasma physics, controlled fusion, and the design of nuclear and conventional weapons.

While we focus describing the ways we overcame the challenges faced by the CREATE program, similar challenges will be faced by any organization seeking to use physics-based modeling and simulation for product development or scientific research.

Final Note

This is a "how to" book about creating and managing programs to develop the software needed to make the paradigm shift to digital product design and performance analysis, which is the heart of virtual prototyping. We focus on establishing and executing a successful virtual prototyping program. Included are the following topics:

1. A brief history of virtual prototyping and computing (Chapter 1)
2. The ecosystem required to support product development and scientific research (Chapter 2)
3. Successful examples of virtual prototyping (Chapters 1 and 5)
4. Choosing between licensing and internal development of virtual prototyping software (Chapter 3)
5. The software tools that play a role in creating virtual prototypes (Chapter 4)
6. Managing the risks typically encountered in software acquisition and development over the long term (Chapters 3, 8, 9, and 10)
7. Retaining control over the intellectual property created by a virtual prototyping program (Chapter 3)
8. Determining the software requirements (Chapters 6 and 8)
9. Providing the necessary software tools and hardware infrastructure for software development (Chapters 3, 7, and 8)
10. Demonstrating the value of the virtual prototyping to the sponsor (Chapter 5)
11. Development and marketing of a successful proposal to establish a virtual prototyping software development program (Chapter 6)
12. Building and sustaining support for a virtual prototyping program (Chapter 7)

13. Best practices for managing virtual prototyping software development and testing (Chapters 8, 9, and 10)
14. Recruitment and retention of the right workforce (Chapter 11)
15. Opportunities for virtual prototyping in science and engineering, and the challenges posed by architecture changes of next-generation computers (Chapter 12)

We assume that our readers are professional scientists or engineers who have gained extensive skills and experience in the details of basic science and engineering science. Hundreds of thousands of books and papers thoroughly describe those topics. Surveying the software development and engineering literature, we found only one book on software engineering principles and practices for software for scientific research, and none on the software engineering principles and practices for developing physics-based software for product development. We offer this book to help fill that near vacuum.

Acknowledgments

The CREATE program was successfully launched in 2007 by the Office of the Secretary of Defense (OSD) due to the efforts of many scientists and engineers from the Department of Defense (DoD) and other agencies and institutions. We have space to acknowledge only a small portion of the key contributors to CREATE during its first few years and to its subsequent rapid success.

Aviation: Robert Meakin and his deputy, Chris Atwood, established the CREATE Aviation program. Edward Kraft, Scott Morton, Nathan Hariharan, Andrew Wissink, and Roger Strawn made key contributions to the establishment and sustainment of the CREATE Aviation program that included fixed-wing aircraft for the U.S. Air Force and NAVAIR and rotorcraft for the U.S. Army, Air Force, and Marines. They developed and utilized advanced computational algorithms and software architectures that enabled the CREATE Aviation program tools to make groundbreaking advances in aircraft simulation and design analysis. Theresa Shafer provided a useful case study on the utility of Kestrel for flight certification of small UAVs, enabling NAVAIR to acquire the UAVs for their sailors and marines.

Ships: Richard Vogelsong, Tom Moyer, Myles Hurwitz, Scott Littlefield, Adrian Mackenna, Joseph Gorski, and Jon Stergiou provided the vision and leadership for the formation and sustainment of the CREATE Ships program to address key naval ship design issues, including ship vulnerability, maneuvering, seakeeping, and operations.

Radio Frequency Antennas: Kueichien Hill, John D'Angelo, and Michael Gilbert founded and sustained the CREATE Radio Frequency Antenna program, which has played a strong role in electromagnetics and radar and communications antenna design for DoD systems.

Geometry and Meshing: Saikat Dey and Ted Blacker founded the CREATE Geometry and Meshing program, which provided the essential capability for users to generate the geometries and meshes that are the starting point for design and analysis. Saikat Dey, Eric Mestreau, and Romain Aubry sustained this program.

Program Leadership: Chris Atwood served as the deputy CREATE Director before returning to Silicon Valley. Dr. Meakin succeeded Doug Post as the CREATE Director.

Program Management: Kevin Newmeyer, Scott Sundt, Portia Bell, Paula Gibson, Loren Miller, Larry Votta, and Linda Park freely shared their technical,

organizational, and program management experience and skills with the CREATE team. This really helped the CREATE program succeed in the long run.

DoD HPCMP: Cray Henry, Director of the DoD High Performance Computing Modernization Program (HPCMP) from 2004 through 2011, provided essential support and leadership needed by the CREATE program during its formative years. Nathan Hariharan, Louis Turcotte, Robert Meakin, Scott Morton, CDMR Kevin Newmeyer (USN, ret.), Denise O'Donnell, and Capt. Scott Sundt (USN ret.), from the DoD HPCMP, reviewed the draft manuscripts for book and gave us their comments and suggestions.

The Carnegie Mellon University Software Engineering Institute (CMU/SEI): Director Paul Nielsen and SEI Chief of Staff John Bramer provided IPA appointments, support for writing this book, and strategic and technical advice and support that enabled Doug Post and Chris Atwood to be successful participants in the HPCMP and the CREATE program. Doug is particularly grateful for John's gift of a copy of Hugh Montgomery's book *Bureaucratic Nirvana, Life in the Center of the Box (the Pentagon)*, an essential guide to successfully navigating the DoD bureaucracy.

Charles Holland (now CMU/SEI, but OSD when CREATE was approved) played a key role in paving the way for the CREATE program within the OSD and supporting the CREATE book project by CMU/SEI.

The authors are also grateful to their SEI/CMU colleagues Charles Holland, Harry Levinson, Gerald Miller, and William Thomas, who reviewed drafts of the book. Todd Loizes (CMU/SEI) and David Biber (CMU/SEI) provided graphics assistance that was essential for developing many of the figures and diagrams used in the book. Copy editing by Gerald Miller (CMU/SEI) was also a highly necessary and appreciated contribution.

Douglass Post is grateful for the mentoring and strong support by Paul Rutherford while Doug was a post-doc and research staff member of the Princeton Plasma Physics Laboratory. This was especially helpful during the early years of his career, when he was making a transition from experimental atomic physics to tokamak modeling, and also later when he led the ITER Physics Group during the ITER Conceptual Design Phase.

About the Authors

Douglass E. Post, Carnegie Mellon University Software Engineering Institute

Dr. Douglass Post was the Chief Scientist of the DoD High Performance Computing Modernization Program from 2005 to 2014 and was a Principal Researcher at the Carnegie Mellon University Software Engineering Institute from 2016 to 2020. His major professional interest for almost 50 years has been the development and application of physics-based software to solve basic and applied atomic, molecular, plasma and controlled fusion physics problems, as well as physics and engineering design problems. In 2005, he established and led the DoD Computational Research and Engineering Acquisition Tools and Environments (CREATE) program, a multi-year DoD program to develop and deploy physics-based computational engineering software for the design of ships, air vehicles, ground vehicles, and RF antennas. He was the Associate Editor-in-Chief of the AIP/IEEE publication *Computing in Science and Engineering* for 20 years. Doug received a Ph.D. in physics from Stanford University in 1975. He led the tokamak and atomic and molecular physics modeling group at Princeton Plasma Physics Laboratory (1975–1993), the International Thermonuclear Experimental Reactor (ITER) Physics Project Unit (1988–1990), and the ITER Joint Central Team In-vessel Physics Group (1993–1998). More recently, he led the Lawrence Livermore National Laboratory A Division (1998–2000) and Los Alamos National Laboratory (2001–2003) with X-Division programs to develop physics-based computer simulations for the design of nuclear weapons. He is the author of over 250 scientific and engineering publications and a Hertz Fellow, a Fellow of the APS, ANS, and IEEE; an AIAA Associate Fellow; and the recipient of ASNE 2011 Gold Medal.

Richard P. Kendall, DoD HPCMP CREATE Program

Dr. Richard Kendall is a software engineering consultant with the DoD High Performance Computer Modernization Program (HPCMP). He has more than 50 years of experience in applications of computational mathematics to engineering problems. After graduating from Rice University in 1972, he joined Esso Production Research Co. (now Exxon Mobil) as a Senior Research Mathematician. There he was directly involved in the development of physics-based reservoir simulators, the enablers of early virtual prototypes of petroleum reservoirs. During the early 1980s, he helped

start an ISV to develop high-performance software modeling tools for the international oil market, specifically state-owned companies that lacked the expertise of oil industry leaders such as ExxonMobil. This startup was eventually sold to Western Atlas International (now Schlumberger), where Dr. Kendall became the Senior Vice President and Chief Operating Officer of the Western Atlas Software Division. In the mid-1990s, he joined Los Alamos National Laboratory as a team leader for Oil & Gas Programs. In 1996, he was appointed Director of the Computational Testbed for Industry. This DOE User Facility provided U.S. companies such as Procter & Gamble access to state-of-the art supercomputers of the 1990s (for example, the Thinking Machines CM-5). Later at Los Alamos, he was appointed Chief Information Officer of the Laboratory. In 2007, he followed Dr. Douglass Post to the HPCMP CREATE program, where he still consults. He is a member of IEEE and SPE/AIME and has published in the fields of numerical analysis, petroleum engineering, information technology, and software engineering.

Chapter 1

The Power of Physics-Based Software for Engineering and Scientific Research

Nearly all mature organizations reach a time when the ways in which they have been developing products or conducting scientific research are no longer competitive. The organizations that survive (and thrive) find new and better ways to operate. In this chapter, we describe the paradigm of virtual prototyping that provides a better way to develop products and conduct research.

1.0 A New Product Development Paradigm

Since the end of World War II in 1945, computing power has increased exponentially, from about 1 floating point operation per second (FLOPS) to more than 10^{17} FLOPS (Strohmaier 2015). The capability to store, access, distribute, and share data has increased concomitantly. These technologies are enabling engineers and scientists to use computers to make revolutionary advances in engineering and science.

One consequence of this explosive growth in computing power is that the experiment-based "design, build, test, iterate" model (Post 2014), with roots in the 19th century Industrial Revolution, is no longer competitive in the modern world for the design and manufacture of many complex products. Engineers are using virtual prototypes of complex products to dramatically reduce the risk, time, and cost of developing them, while also improving their performance and quality. Examples range from the design of golf clubs and automobile tires, to the analysis of automobiles crashes, design and performance predictions for naval vessels, commercial

and military aircraft, and rocket engines. The prototypes are also often referred to as "digital models" or "digital surrogates" (Forrester 2008 and Saddik 2018). These different terms are often used as synonyms. However, there might be many different virtual prototypes, but only a few digital surrogates might exist, based on the final designs and software capabilities.

Scientists are using digital surrogates to conduct new and ground-breaking fundamental scientific research that is not possible with traditional methods that are based only on physical experiments. In the scientific research arena, the use of virtual surrogates plays a key role in understanding supernova explosions, predicting the weather, and designing new materials (Council 2010 and Dongarra et al. 2003).

1.1 Computational Engineering and Virtual Prototypes

Engineering can be defined as follows:

> *The use of science and mathematics to design, construct, or manufacture physical systems.*

Computational engineering is an extension of classical engineering that makes it possible to shift from physical prototypes to virtual prototypes. Computational engineering adds:

> *Digital models and simulations, often coupled with high-performance computing, to solve complex physical problems arising in engineering analysis and design.*

Traditional engineering product design involves the repeated iteration of four steps: 1. requirements and conceptual design, 2. detailed design, 3. construction of physical prototypes, and 4. experimental testing of the physical prototypes. Figure 1.1 depicts this process. The design analysis was initially carried out using hand calculations and heuristics, then calculators, and, more recently, computers (Consortium 2015 and Paquin 2014).

If the experimental physical prototype tests are successful, the product proceeds to manufacture. The manufactured products are tested experimentally to determine whether they meet requirements.

The manufacturing process is guided by engineering drawings, which can be digital, paper documents, and possibly even a physical model of the product.

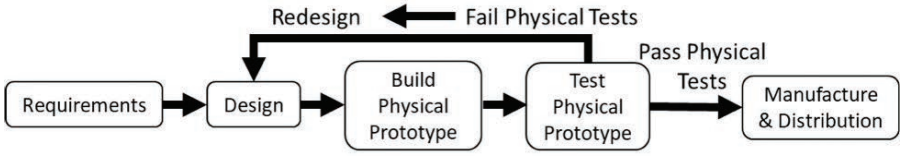

Figure 1.1 *Traditional product development process based on physical prototypes*
(Courtesy Software Engineering Institute)

Computational engineering exploits the use of computers to supplement or even replace the use of physical prototypes with virtual prototypes for the development of complex products (Consortium 2015, Post 2015, and Post 2009). Figure 1.2 illustrates this process. The starting point of a virtual prototype can range from a 2-D CAD drawing to a 3-D NURBS (non-uniform, rational B-spline) description of the product geometry, including all the associated metadata needed to completely describe the product and its attributes. The latter is called the digital product model. It can also include the history of the design process, a complete record of the analysis of the product design, and the data needed to start product manufacture.

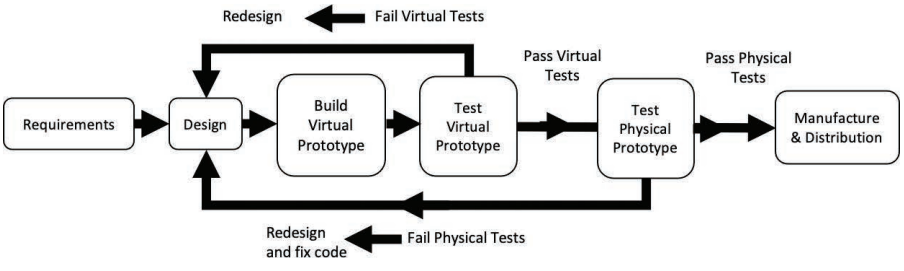

Figure 1.2 *Computational engineering product development process based on virtual prototypes*
(Courtesy Software Engineering Institute)

Figure 1.3 compares the two approaches.

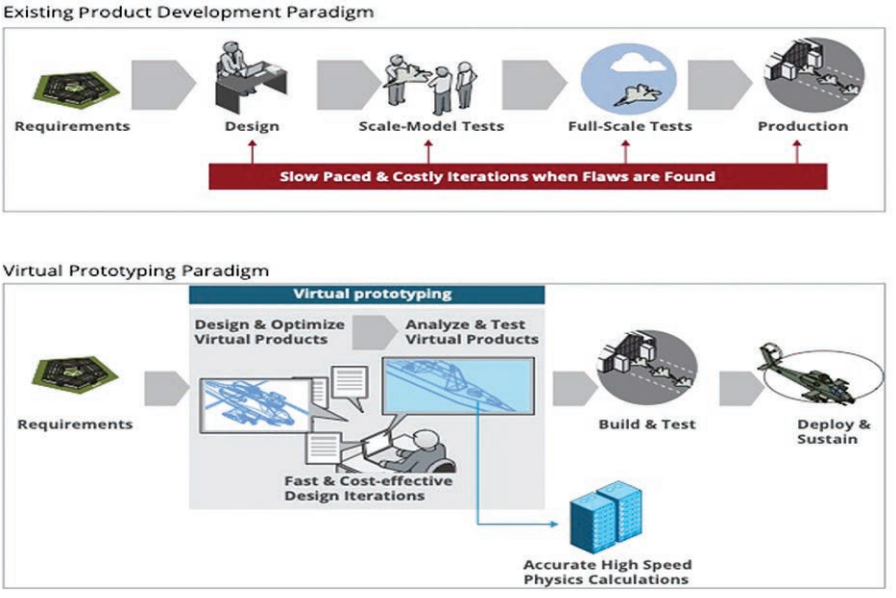

Figure 1.3 *Schematic comparison of historical empirical iterated "design, build, test" paradigm and the virtual prototyping paradigm*

(Courtesy Software Engineering Institute)

What does it mean to pass a "virtual" test? Chapter 10, "Verifying and Validating Science-Based Software," discusses this in detail. However, remember that Mother Nature always casts the final vote. Her laws of physics enable us to predict her vote before we build a physical prototype of the product and test it. As our understanding and experience grow, the need for physical confirmation decreases.

Although the widespread use of computational engineering is relatively new, the use of computers to design and analyze the performance of products, or at least components of them, dates at least as far back as the early days of the Manhattan Project during and just after World War II (Dyson 2012, Ford 2015, and Atomic 2014). From the end of World War II to the present, the U.S. Department of Energy nuclear weapon laboratories have relied on virtual prototypes to maintain and optimize the designs of nuclear weapons (Energy 2019). When high-performance computers became generally available (1960s onward), industry and government began to use them to develop new, more complex products (Council 2005, Post et al. 2016, and Paquin 2014).

More recently, the concept and capabilities of computational engineering (and science) have been extended to include the whole process of product development and scientific research. This is expressed with terms such as *digital engineering* and the

digital thread (Fei Tao 2019 and Kraft 2016). Starting with the digital product model, the artifacts (surrogates and design metadata, along with maintenance and operational history) of the entire product lifecycle are linked together as a digital thread.

Computational engineering extends the classical engineering approach to make use of software and computers to model the effects that determine the behavior or performance of a product before it is manufactured. Only in the last decade have high-performance computers reached the petascale capability (10^{15} FLOPS) to handle all the relevant aspects of the design of very complex products such as aircraft and ships. However, even high-performance computer workstations are often adequate for designing and testing smaller, simpler products. The utility of computational engineering is not limited to designing complex machines such as aircraft. It is also being used to design consumer products such as bicycles, golf clubs, bleach and detergent bottles, and even hip replacement joints and other medical appliances.

The strong need for a new approach for the product development of complex systems based on computational engineering can be illustrated by the history of three of the most recent U.S. Department of Defense (DoD) major multibillion-dollar air vehicle procurement programs described in a recent Hudson Institute report (Greenwalt 2021):

- F-35 Joint Strike Fighter (JSF)
- F-22 Raptor Stealth Fighter
- V22 Osprey Tilt-Rotor

Following the present "design, build, test, repeat" (see Figure 1.1) approaches that rely on the development and testing of physical prototypes for these three aircraft, DoD contractors now need more than 20 years from project start (contract award) to deliver fully operational aircraft to the DoD. In 1975, they needed only about 5 years. The Joint Strike Fighter (F-35) program started in 1996. The first fully operational F-35 was delivered 15 years later in 2011 (F-35 2021). The costs almost doubled from project start to acceptance by the Air Force (Wheeler 2012). Without improvement in aircraft procurement methods, it will continue to take the U.S. longer to develop new aircraft. The F-22 program started about 1986. Delivery of the first fully operational aircraft occurred in 2005, nearly 20 years later (F-22 2021). The number of F-22 fighters that the DoD ultimately purchased dropped from 750 to 183, due to the increase in price of each airplane and the delay in the program (Ritsick 2020). The final cost was $340 million for each F-22. The V-22 Osprey Tilt-Rotor program started about 1983. The first full operational capability (FOC) aircraft was delivered in 2005, more than 20 years later (V-22 2021).

Before 1975, DoD defense contractors generally took no longer than 5 to 7 years to develop and deliver new aircraft. The Lockheed F-117 Nighthawk Stealth bomber program, the General Dynamics F-16 Fighting Falcon, and the McDonnell Douglas F/A-18 Hornet were begun in the 1970s and delivered in 5 to 6 years. The McDonnell Douglas F-15 Eagle, started in 1967 was delivered in 1976, 9 years later.

Other types of complex aircraft have not experienced this level of growth in development time. During the this period (1967 to 2011), the time to market for new commercial aircraft such as the Boeing 737 (delivered in 1967), the Boeing 767 (in 1982), and Boeing 787 (in 2011) rose by only two years, from approximately 5 years to 7 years. Similarly, the time to market for new automobile and truck models has remained 4 to 6 years. Commercial aircraft and automobiles and trucks are quite complex, with massive amounts of embedded software, high reliability requirements, and strong cost constraints (Greenwalt 2021).

The recent history of Goodyear Tire and Rubber Co. provides a compelling example of the benefits of computational engineering for reducing the "Time to Market." Facing fierce competition from Europe (Michelin) and Japan (Bridgestone), Goodyear decided in 1992 to build physics-based tire design software (which we henceforth often refer to as *tools*). Goodyear entered into an ongoing collaboration with Sandia National Laboratory that enabled the company to combine its knowledge of tires and its materials with Sandia's knowledge of finite element algorithms for massively parallel computers to develop this capability. In 2003, Goodyear used the tire design tool to reduce its time to market by a factor of 4 (Miller 2010, Miller 2017, and Council 2009). By enabling designers to generate and analyze more design alternatives, the tire design tool also allowed Goodyear to increase the number of new products per year, from 10 to 60. The Goodyear annual reports began to refer to the company's "New Product/Innovation Engine." This story is being repeated in many industries, such as Ford automobiles (Kochhar 2010), Whirlpool refrigerators (Gielda 2009), and Procter and Gamble shampoos and hand lotions (Lange 2009).

Today the use of virtual prototypes can supplement and, in some cases, even replace the use of physical prototypes (Post 2015). Design engineers can construct and store thousands or even millions of 3-D virtual prototypes of a potential product in a design option tradespace. The performance and behavior of each virtual prototype can be quickly assessed using simplified physics-based software. Further analysis can identify the most promising design options. More sophisticated high-fidelity computational analysis tools can then be used to accurately predict their performance. Finally, if needed, a physical prototype of the final design can then be constructed and tested for final verification of the design before manufacture.

Another advantage is that virtual prototyping can greatly accelerate product innovation. Final design decisions can be postponed until later in the design process, when more information is available through virtual testing of the design candidates.

Usually this can be done orders of magnitude more quickly and cheaply with virtual prototypes than with physical prototypes. Virtual prototypes make it possible for designers to learn by developing and testing a much larger tradespace of designs. Goodyear used virtual prototyping to increase its innovation rate by a factor of 6, from 10 new tires per year to 60 new tires per year. Virtual prototyping enabled the company to rapidly learn through failure, similar to the Silicon Valley mantra (Petroski 2006 and Post 2017).

Numerous studies (Augustine 2007, Oden 2006, and Glotzer 2009) have highlighted the risks to U.S. international competitiveness if U.S. industry falls behind the rest of the world in the use of computational engineering. It is widely acknowledged by senior U.S. industry and DoD leaders (Cordell 2018) that, to remain competitive, U.S. industry and the U.S. government and its contractors must reduce their time to market, costs, and risks while producing high-quality, market-worthy products.

1.2 Computational Science and Digital Surrogates

Science can be defined as follows:

> *The systematic study of the natural world to identify general laws and principles.*

Computational science is defined as follows:

> *The use of computers, models, and simulations to develop a quantitative understanding of natural phenomena.*

Scientific research generally involves four major steps (see Table 1.1):

Table 1.1 *Four Major Elements of Scientific Research*

1. **Observations** of natural systems
2. **Controlled experiments** with the real system or models of the system
3. Development of **mathematical theories** of the behavior of natural systems
4. **Predictions** of system behaviors, based on the mathematical theory and machine learning algorithms trained on observations of natural systems

The first three elements are iterated and can occur in any order. However, predictions require a model and data of sufficient accuracy and breadth. Figure 1.4 illustrates the scientific research process based on these four steps.

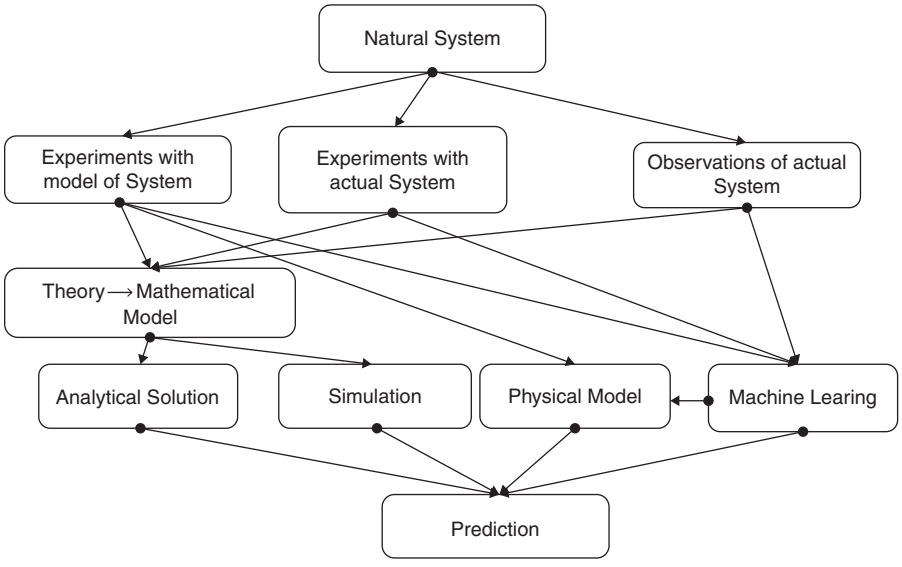

Figure 1.4 *Schematic illustration of the computational science research process (Courtesy Software Engineering Institute)*

Computational science has joined experiments and theory as a third leg of the stool that supports scientific advances.

Today observations of natural phenomena such as terrestrial and space weather, planetary systems, chemical reactions, genetic evolution, geologic phenomena, fires, and hundreds of others increasingly rely on the collection and analysis of extremely large data sets. Generally, understanding and predicting the consequences of these natural events require detailed solutions of complex, nonlinear mathematical models that can be obtained only with computers. Additionally, recent advances in general-purpose graphical processing units (GPGPUs) and other accelerator chips optimized for the rapid processing of streaming data flows, together with the development of computational software for multilayer neural networks, have enabled the analysis of very large data sets (many millions of elements) (Krizhevsky 2012 and Rumelhart 1986). Together with the explosion of data from hundreds of different kinds of sensors (cameras, microphones, pressure transducers, magnetism sensors, and so on), computers can be trained using various machine learning algorithms on large sets of data from observations of the natural system and experiments for both real and

model systems. This is opening up a revolution in artificial intelligence and can supplement physics-based algorithms.

Investigating the behavior of natural systems focuses on analyzing and developing digital models of natural systems of interest. The models serve as digital surrogates for the actual natural systems. This approach is especially useful for natural systems upon which scientists have little or no influence, such as the weather, climate, and stellar formation and evolution. Scientists can conduct controlled virtual experiments by inserting different physics models into the calculation to see which physics models and sets of initial conditions provide the best match to the observed data. The best physics models can then be used to predict the future behavior of the system. A key advantage of using models for the study of natural systems is that it is often possible to study the details of the behavior of a virtual system's internal components. This is often very difficult or impossible for many highly interesting real systems (such as supernovae, the interior of the sun, or even large weather systems).

A key difference between computational engineering and computational science is its goals and approaches. One goal of computational engineering is the design of specific, complex products. The software application is a tool for developing the design. All the physics and solution algorithms in the software application are known, verified, validated, and accredited. The outcome of the calculations is the product design. In contrast, the primary goal of computational science is the discovery of knowledge. The purpose of the simulation is to determine which, if any, of a set of candidate physics or other scientific models best fit the observational or experimental data. If none of the models fits the data, new models are needed.

This is the current situation in astrophysics. The universe is now thought to consist of 5% ordinary matter and energy, 27% an unknown type of matter called dark matter (because we can't see it) (Trimble 1987), and 68% an unknown form of energy known as dark energy, which we also can't see (Frieman 2008). The evidence for dark matter is this: Dark matter is needed to explain why spiral galaxies don't fly apart instead of just rotating. There just isn't enough observed matter to provide the gravitational attraction needed to keep them from flying apart. The evidence for dark energy is that it is needed to explain why the universe is expanding at a rate that continues to increase instead of decreasing as would be expected from gravitational attraction of the observed matter (and dark matter) in the universe. The evidence for the existence of dark matter and dark energy is indirect. It is partly based on the failure of the known physics models to explain the observations described previously. Our lack of any fundamental understanding of 95% of the matter and energy in the universe makes dark matter and dark energy major focal points for current scientific investigation and study.

Weather prediction (meteorology) is a good example of a successful applied computational science research program based on digital surrogates of a complex natural system. The science of weather forecasting began in roughly 1860 with the

establishment of the United Kingdom (U.K.) Meteorological Office (MET) by Admiral Robert FitzRoy (Blum 2019). Earlier in his career, FitzRoy was the captain of the *HMS Beagle* when it carried out a survey of the southern part of South America, with Charles Darwin as the naturalist (1831 to 1836). FitzRoy based his forecasts on weather data sent to him by telegraph from various upwind reporting stations around the U.K., especially those on the western and northern shores. Knowledge of basic principles of meteorology steadily improved during the latter half of the 19th century and the first half of the 20th century (Fleming 2016 and Sawyer 1962), but it still roughly followed FitzRoy's method of collecting and analyzing upwind weather data (air temperature, air pressure, humidity, wind speed and direction, and precipitation, for example). Weather prediction began to improve more rapidly in the latter half of the 20th century due to the exponential growth of computing following the end of World War II, together with more extensive and more complete data collection and analysis. "Modern 72-hour predictions of hurricane tracks are more accurate than 24-hour forecasts were 40 years ago" (Alley 2019). Alley and his collaborators credit today's accuracy to improvements in computing power, better understanding of the major physics effects, better data collection, and better computing techniques.

Computational techniques are also proving useful for soft sciences, which involve complex interactions among living systems (animals, people, microbes, and so on) with the natural world. These include biology (medicine, epidemiology, genetics, and others), military strategy and war games, social and political behavior, ecology, economics, finance, business planning, and many other complex systems. Sports teams are even constructing digital twins for their players, to track their physical condition and predict their performance (Siegele 2020).

1.3 The Computational Engineering and Science Ecosystem

The virtual prototyping/digital surrogates/digital paradigm requires a computational ecosystem (see Table 1.2).

Table 1.2 *Elements of the Computational Ecosystem*

1. Computers (or at least computer time)
2. Computer networks (access to computers)
3. Data storage
4. Experienced and skilled users (customers)
5. Testing and test data
6. Application software (including developers and user support staff, if software is to be developed)

Figure 1.5 presents an example of such an ecosystem for the DoD's High-Performance Computer Modernization Program (HPCMP). The next chapter discusses these elements in more detail.

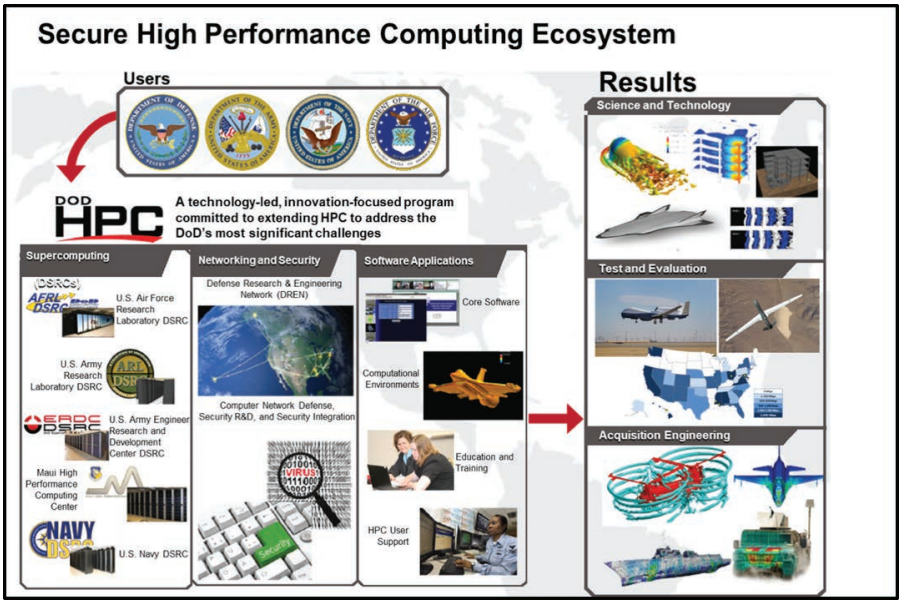

Figure 1.5 *Illustration of essential elements of a successful customer-focused, secure ecosystem required to support virtual prototyping from HPCMP*

(Courtesy DoD HPC Modernization Program)

The HPCMP ecosystem supports both product development and scientific research. It includes secure, cloudlike, high-performance computing and networking resources, along with various software applications, including the CREATE family of virtual prototyping software applications. The outputs of this ecosystem are virtually tested air vehicle, ground vehicle, ship and antenna design, and scientific research results for DoD problems.

Although computers, computer networks, and data storage capabilities are at the forefront of advanced technologies, they are primarily general-purpose, commodity technologies that are available from commercial vendors along with supporting services (for example, the cloud and commercial communication networks). The need for a workforce with skills in the relevant engineering and scientific disciplines is obvious. Organizations already have much of the workforce they need to use computers for design or research and understand how to recruit to augment their workforce.

Acquiring the needed software applications presents a fundamentally different challenge than obtaining computers, communication networks, data handling, and even a capable engineering workforce. It is important to recognize that technical software is not like hardware. It is not a commodity and is not general purpose. Almost always, every technical software application is developed for a specific set of technical problems. Software used to calculate the effects of airflow across an airplane wing cannot be used to determine the performance of an antenna. Engineering and scientific software is based on the laws of science (physics, chemistry, and so on). Software that can address a complex technical problem is at least as complex as the problem. Case studies indicate that developing a good multiphysics engineering or scientific software application generally takes 5 to 10 years (Post 2004). At the top of the list of important requirements for an engineering and scientific software application is accuracy. By this, we mean that the software must be accurate enough to support its potential uses. Validation testing, including comparison of the software predictions with experimental test data, is a key requirement for demonstrating the needed accuracy.

For many design and analysis tasks, product development organizations can obtain application software from independent software vendors (ISVs). The competitive advantage shifts from the software to the effective use of the software. If commercial software products can meet an organization's needs, licensing can be attractive. However, commercially available software may not meet the needs of a specific product development effort. Crucial features may be missing. The algorithms for important effects may not be accurate enough. In that case, it may be necessary to develop the needed software applications. Following the 5 to 10 years needed to develop the multiphysics simulation, the software will require continued support for the whole time it is needed. This book focuses on lessons learned from examples of real programs that have developed and deployed such software.

Scientific research organizations face similar challenges to develop and deploy scientific research software. Unless other research organizations have developed software that meets a research organization's needs and are willing to share, each organization will need to develop its own software. Even if externally developed software is available, considerable modification and future development may be required to address evolving needs.

For both product development and research, a dearth of useful, actionable information exists about successful software engineering and project management principles and practices for developing large-scale scientific software applications. The evidence indicates that many, if not most, of these projects either fail to meet their

goals or fail totally (Ewusi-Mensah 2003, Glass 1998, and Gorman 2006). The purpose of this book is to address this need.

1.4 High-Performance Computers: The Enablers

The principal driver and enabler of computational engineering and science is the exponential growth of computer performance since the end of World War II: 17 orders of magnitude. We are not aware of any other technical advance of this magnitude.

Two Historic Increases in Mankind's Technological Capabilities

- Explosive power from a few pounds of black powder in the Civil War, to the 50-megaton hydrogen bomb tested in the USSR in 1961—a net increase of ~$2x10^{13}$ in explosive power
- Continuous human-led travel speed increase of ~$3x10^{4}$ spread over millions of years
  - Human walking: 2 to 4 mph
  - Horse travel: 30 mph
  - Railroad travel: 50 to 400 mph
  - Civilian jet aircraft travel: Up to 2200 mph
  - New Horizon mission to Pluto: 50,000 mph

There are only several dozen supercomputers in the world with near-Exascale performance (10^{18} FLOPS), and only a handful of engineers and scientists at the world's largest research and engineering facilities have access to these computers. However, many, if not most, scientists and engineers in major industries, universities, and government laboratories in the U.S., Europe, and Asia, now have access to supercomputers with processing powers in the PetaFLOP range (1 PetaFLOP=10^{15} FLOPS) (see Figure 1.6) (Strohmaier 2015). This enables them to tackle and solve problems at the leading edge of scientific research and engineering design. Also, there are thousands of powerful servers, and millions of desktop and laptop personal computers with processing power in the GFLOPS to TeraFLOPS range. It's a "Brave New World," and the industries, universities, and government laboratories not using these technologies are beginning to fall behind.

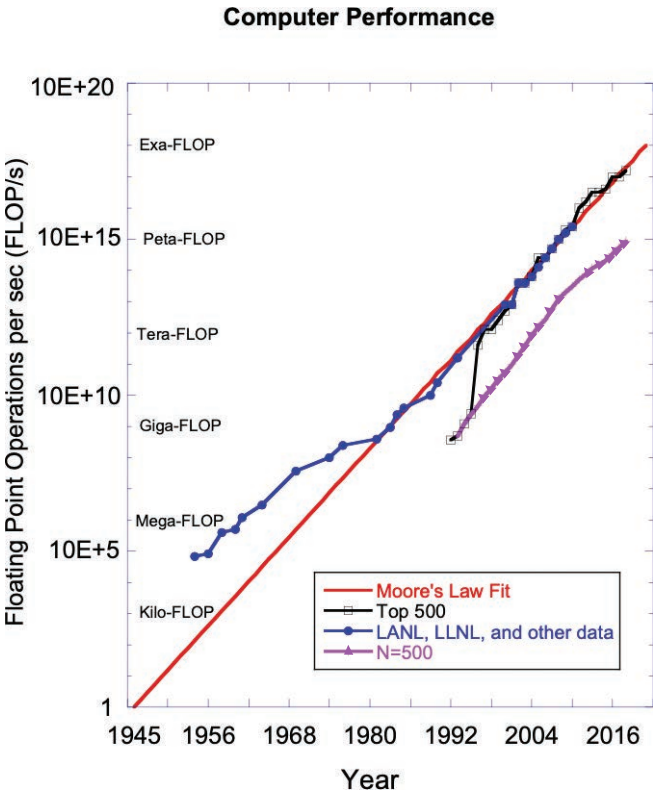

Figure 1.6 *History of supercomputer performance growth, 1945 to 2019*

(*Data is taken from the top500.org website,* http://www.top500.org/; (Erich Strohmaier 2015); and historical data from LLNL and LANL, analyzed by Douglass Post.)

1.5 Full-Featured Virtual Prototypes

Only in the last decade or so has the full promise of virtual prototypes been realizable. By 2010, the processing power of the fastest high-performance computers had reached a level to support accurate predictions of the future performance and behavior of full-scale, full-featured products and complex natural systems. The previous generation of digital surrogates often suffered from a lack of adequate resolution and the inability to include all the important physics needed for accurate predictions. The capability to deploy *high-resolution*, *multiphysics*, *systems-of-systems* software makes full-featured (see Table 1.3) product design now possible.

Table 1.3 *Attributes of Full-Featured, Physics-Based, High-Performance Computing Software Applications*

1. The applications include all the major physical effects that determine the performance and behavior of the system.
2. They enable the design and generation of complex, multidimensional (3-D) digital product models of full-scale systems, not just components.
3. They support the high-resolution calculations needed to capture fine detail.
4. They use highly accurate mathematical and computational solution algorithms.
5. They can demonstrate the validity of the predictive capability of the software applications through comparison with detailed, accurate experimental test data.
6. They can predict the performance and behavior of a complex full-scale system (for example, an entire ship, airplane, or planetary weather system).
7. They are able to quantify the uncertainties of the predictions.
8. They can complete a high-fidelity, time-dependent, multidimensional, multiphysics calculation in minutes to hours, to provide timely results (compared to months to years in 1998).

A tremendous advantage of these new capabilities is that they enable industrial, academic, and governmental organizations to collect and consolidate, in a software application, their most important and insightful past corporate knowledge and experience, along with their new research knowledge. Experienced engineers can then immediately use that software application to develop designs that are informed by this knowledge.

Very often in the past, many of the most promising results of government, academic, and industrial research programs have not been successfully transitioned into real products or applications, due to these limitations:

- The lack of transition paths
- The difficulties of gaining the interest, acceptance, and endorsement of the design community
- Funding obstacles for developing and testing prototypes
- Competition from other promising research results

In the U.S. Department of Defense, concepts that fail to transition from research into real products or operational concepts fall into what is referred to as the "Valley of Death" (Montgomery 2010). Computational engineering can help bridge this chasm by providing a path to real application of the research (see Figure 1.7).

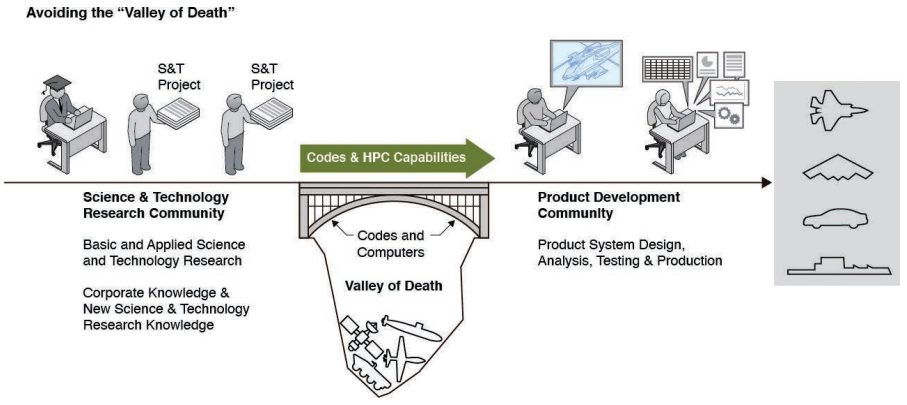

Figure 1.7 *Physics-based software with high-performance computing can provide the means to transition from science and technology discoveries to the design and production of real systems to leap over the "Valley of Death"*

(Courtesy CMU Software Engineering Institute)

1.6 The Advantages of Virtual Prototyping for Systems of Systems

The construction and analysis of virtual prototypes is a natural and powerful method for analyzing the behavior of systems of systems. A major advantage is that it forces the engineer or scientist to address all the important effects that determine the behavior of the system of systems, not just the ones that are the easiest to address. We illustrate that with the example of air vehicles.

1.6.1 Systems of Systems: Aircraft

Air vehicles are an example of systems of systems that provide natural tests of our depth of understanding of their operation. To accurately model and predict their behavior, it is necessary to successfully integrate all the crucial physics elements that determine the system performance. If the software cannot successfully do that, it is missing at least one key part of the required physics capability. In practical terms, this means that the software developer has to build and implement solution algorithms that can smoothly integrate all the important individual physics effects and operational controls (see Figure 1.8). One of the CREATE software applications, Kestrel, illustrates this (McDaniel 2016 and McDaniel 2020).

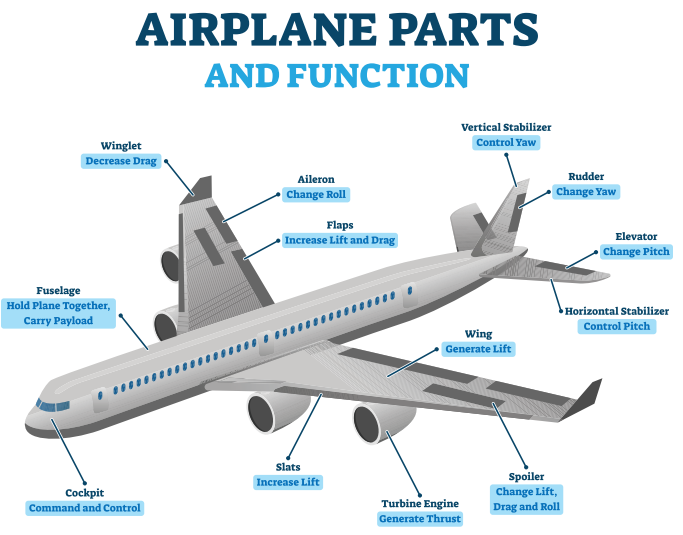

Figure 1.8 *Major aircraft systems and functions for a generic commercial airliner (VectorMine/Shutterstock)*

The major physics effects include the following:

- Time-dependent motion with six degrees of freedom (pitch, yaw, roll, surge, heave, and sway—6DoF)
- Gravity and Newton's laws of motion
- The forces on the aircraft structure due to the flow of air across the outer surfaces
- The reaction of aircraft structure to those forces
- The forces due to propulsion systems (jet engines or propellers)
- The forces from the air flow on the passive and active control systems
- The forces on the landing gear

Kestrel was developed to smoothly integrate all these effects to be able to accurately calculate aircraft flight for subsonic and supersonic aircraft. The predicted flight path and other measures of performance compare well with measured flight data and wind tunnel tests. Kestrel is being used extensively by the U.S. Naval

Aviation and Air Force communities (Shafer 2014), as well as U.S. industry (Stookesberry 2015). Kestrel includes:

- A six degrees of freedom (6DoF) algorithm for computing the aircraft motion in response to the forces acting on it
- A computational fluid dynamics solver for computing the air flow and resultant forces on the structure, including the passive and active control systems
- A computational structural dynamics solver for computing the response of the structure to the airflow and other loads
- A hierarchy of models for computing the effects of the propulsion systems, from one-dimensional empirical models to turbo-machinery models
- The capability to dynamically move the active control surfaces and accurately calculate their effect on the flight path and the loads placed on them

Described in very simple terms, aircraft are solid objects moving rapidly through a fluid (air). Their forward motion is driven by a propulsion system (propeller or jet). As the aircraft moves, it pushes air out of the way. The air flowing around the aircraft (particularly the wings) produces forces (lift and drag) on its structure that enable it to overcome gravity and fly (lift). Displacing the air and friction with the airplane exterior skin takes energy that generates resistance to the forward motion of the aircraft (drag). An aircraft moves through time and space with six degrees of freedom, based on the forces from the airflow, the propulsion system, the passive and active control surfaces, and gravity. The airflow can exhibit turbulence, depending on local conditions. The software must be capable of computing this complex motion.

As evident from Figure 1.8, aircraft are highly complex systems. The major external components are the fuselage, wings, engines, control surfaces, and landing gear (not shown). Interior struts, ribs, and so on embedded in the fuselage, wings, static control surfaces (vertical and horizontal stabilizers), and the landing gear provide most of the aircraft's structural strength.

Much of the complexity of an aircraft system is due to the need for flight control. This representation of commercial aircraft has 12 active control systems (6 on the right and 6 on the left) and 7 passive control systems. A real commercial airliner likely has more. Flight control is essential. The Wright brothers were among the first to appreciate this and the first humans to successfully achieve powered, controlled flight. This insight made it possible for humans to be able to fly.

Below their exterior surface, modern aircraft are highly complex as well. In addition to structural supports for the aircraft exterior, there are fuel tanks, propeller and jet engines, fuel lines, avionics, the flight cabin, the cockpit, cargo spaces, hydraulic systems, steel cables, power lines, electronic cables, sanitary systems and plumbing, exterior and interior sensors, communication systems, radar systems, weather systems, flight recorders, navigation systems, ventilation, emergency oxygen, cabin pressure systems, cabin doors and windows, life rafts, life preservers, and myriad other systems. The aircraft doesn't need all these systems to fly, but they are all essential for the aircraft to meet all its operational goals for a pleasant safe flight. The only safer way to travel today is in elevators.

1.7 Virtual Prototyping: A Successful Product Development and Scientific Research Paradigm

As mentioned earlier, virtual prototyping has a long track record of solid achievements. Virtual prototyping for the Manhattan Project was one of the major drivers of the original development of computers during World War II until the mid-1950s (Ford 2015 and Atomic 2014). Manhattan Project scientists developed 1-D virtual prototypes to predict the criticality of nuclear reactors and nuclear explosives and the implosion compression of fissile materials. Following the end of World War II, the development of electronic computers continued at a number of institutions, including the Institute of Advanced Studies at Princeton with John von Neumann and others, where the first "modern" computers were developed (Dyson 2012). The U.S. nuclear weapons programs at the Los Alamos, Livermore, and Sandia National Laboratories have continued the work begun during the Manhattan Project to pioneer the use of supercomputers to successfully design and sustain the U.S. nuclear stockpile. Nuclear tests are expensive and unpopular, so simulating and optimizing nuclear explosions with a computer offers tremendous advantages. Instead of conducting hundreds of real nuclear tests, only a few are needed to calibrate and confirm the computer predictions. These U.S. Department of Energy (DOE) laboratories continue to be among the leading users of high-performance computing hardware and software, especially since the U.S. stopped conducting nuclear tests in September 1992 (Energy 2019).

Motivated by the problems the DoD was having in designing and developing new military platforms, the Office of the Secretary of Defense launched a small experimental program in 2006 to develop and deploy a set of physics-based, high-performance computing software applications. The program goal was to determine whether virtual prototypes could help the department meet the challenges of

designing and producing major military systems more rapidly and less expensively than in using physical prototypes. In addition to the main question about the viability of virtual prototypes, other concerns included these:

1. Could government and contractor teams develop the needed software?
2. Would government and industry groups be able to use the software to design and guide the production of successful complex military systems?

The Computational Research and Engineering Acquisition Tools and Environments (CREATE) program proposal was approved in late 2006 to address these questions. It was executed by the DoD High Performance Computing Modernization Program (HPCMP), beginning on October 1, 2007. The HPCMP rapidly organized and sponsored software application development teams to build and deploy 12 software applications for the design and development of naval ships, military aircraft, ground vehicles, and radio frequency antennas, including developing three-dimensional digital product models. The development teams were located at major DoD laboratories and warfare centers. During the following 12 years, the CREATE program was able to provide examples of the high value of virtual prototypes for the development of complex products (Post et al. 2016). The CREATE software tools are now being used by more than 2,000 acquisition engineers (40% government, 50% industry, and 10% other sectors), contributing to more than 180 different DoD programs.

Many other recent commercial examples of the value of virtual prototyping exist. The benefit to Goodyear Tire of adopting the virtual product development paradigm has already been discussed. The process of adoption at Goodyear was evolutionary. During its transition to use of the tire design software, Goodyear continued to build a physical prototype of the final design produced by the physics-based tire design software and continued to test the physical prototype tire before starting to manufacture it. Goodyear found that the virtual tire designs were so successful that it could start manufacturing immediately after the final design and virtual tests were completed. Goodyear then tested the first tires from the initial production run. Design flaws appeared so infrequently that the advantages of the virtual tire design process (faster and cheaper, with greater product innovation and fewer design flaws) outweighed the disadvantages of an occasional flawed design. Goodyear engineers used the lessons learned from those few flaws to improve the tire design software and reduce the rate of flaws in future designs (Council 2009).

Whirlpool is another example of a company that successfully adopted the virtual prototyping paradigm. Whirlpool is one of the world's leading manufacturers and marketers of major home appliances (including the Whirlpool, Maytag, KitchenAid,

Jenn-Air, Amana, and Brastemp brands, among others). Today's home appliances are highly complex products. Their design and manufacture involve trade-offs of cost, safety, reliability, performance, maintainability, efficiency, and convenience. Safety is a major issue; for example, a washing machine must not tip over if a child sits on the door when it is open. The appliance market is international, and the products must be designed to fit into the homes in dozens of countries on almost every continent. A large American refrigerator or stove will not fit into a smaller European or Japanese apartment, for example. The appliance market is also highly competitive. Appliances can be assembled anywhere, with components from a global supply chain. Whirlpool models the appliances as systems of systems, including the packaging for shipment. A refrigerator that arrives at the appliance store scratched or dented during shipment has lost most of its sale value. Whirlpool models the fluid flow (water, refrigerant, and others), airflow, heat flow transport, mechanical strength of the appliance frame and components, electrical layout and switching, motor and pump performance, level of vibration, noise levels, computerized control systems, and mechanical balance. The company has substantially replaced much of its physical prototypes with virtual prototypes. "Testing using virtual prototypes has, for the most part, replaced physical testing—we're no longer using the old 'heat and beat' approach," says Tom Gielda (Gielda, 2009), Whirlpool's engineering director for global mechanical structures and systems.

Automobile companies increasingly supplement crash tests with virtual tests, which are faster and less expensive. Virtual tests are easier to diagnose and analyze than real crash tests. According to a 2014 press release, Ford Motor Company increased its computing power by 50% to maximize the speed and number of virtual crash tests it can perform (Ford_Media 2014). The automaker performed more than two million crash test simulations from 2004 to 2014. By comparison, Ford performed its 20,000th full vehicle crash test at its Dearborn, Michigan, testing facility around 2014. The new computing power allows Ford to include up to two million finite elements in its virtual crash test simulations, a significant increase from half a million elements a few years before 2014. Ford's virtual tests include front impact, side impact, rear impact, roof strength, and safety system checks.

Virtual prototypes with high-performance computers played a pivotal role in the development of Ford's EcoBoost engine technology that was introduced in late 2010 (Kochhar 2010). Derrick Kuzak, group vice president of global produce development at Ford, stated, "EcoBoost is truly a smart solution for consumers because it provides both improved fuel economy and superior driving performance. The combination of turbocharging and direct injection allows smaller engines to perform like larger ones while still delivering the fuel economy of the smaller powerplant." Nand Kochhar, the chief engineer at Ford for global materials and standards engineering, said, "A lot of HPC-based computational analysis is involved in simulating the

trade-offs between performance, shift quality, and fuel economy. In the case of the engine, we conduct combustion analysis of turbocharging—optimizing a fuel–air mix, for example. To develop overall vehicle fuel efficiency, we use Computational Fluid Dynamics calculations to compute the optimal aerodynamics of the proposed vehicle" (Kochhar 2010).

Procter & Gamble (P&G) has used virtual prototyping in many parts of its product development process (Lange 2009). P&G uses computational tools to design and test its product packaging for plastic bottles intended to hold bleach and other liquids. It modeled the transport of potato chips (for example, Pringles) through baking ovens to maximize the throughput of the chips. The detailed properties of surfactants (the major ingredients in soaps, detergents, lotions, and shampoos) determine how well P&G products meet its customers' needs. In addition, environmental concerns, both with production and with consumer use and product disposal, have become highly important. Initially, the company's knowledge and research on surfactants were experimental, but the research demands began to overwhelm what was possible with that approach. P&G turned to modeling the behavior of the atoms and molecules in the surfactants using computational chemistry and molecular dynamics calculations. With this approach, P&G was able to identify ways to produce better products. Kelly Anderson, a senior scientist specializing in molecular dynamics at P&G, said, "Molecular dynamics allows us to make approximations about the interactions—the chemical potentials that are occurring. By investigating the molecular composition of these materials, we are better able to predict what properties a formulation will exhibit—not only its immediate characteristics, but what will happen to the mix 6 months from now. By mixing and matching different molecules containing different configurations of atoms, we can create the most desirable characteristics for our consumer products, such as detergents and shampoos, and at the same time, ensure they are safe and environmentally friendly. That's really the magic of what we are trying to do" (Lange 2009).

1.8 Historical Perspective

The computer-driven leap in our ability to predict the future has its roots in a realization by Greek natural philosophers in the late 7th and early 6th century BCE: The physical world might not be controlled by the capricious whims of gods and spirits (see Figure 1.9). Greek natural philosophers began searching for an underlying order in nature—for natural and universal laws that relate cause to effect, to explain the physical universe instead of relying on mythology and religion (Barnes 1965, Parkes 1959, and Pollitt 1972). The next era of advances in science and engineering began

during the Italian Renaissance in Italy (15th and 16th centuries), partly through the rediscovery of ancient Greek science and mathematics via the Islamic states in Spain and the Near East. The Age of Discovery, which began in the 16th century and lasted into the 19th century, saw further advances, such as the invention of the calculus to explain motion. The Age of Discovery was followed by the scientific and mathematical advances of the 17th and 18th centuries in Northern Europe, and the scientific revolution in the 19th and 20th centuries in Europe and North America.

Figure 1.9 *Greek Gods and mythological beasts populated the heavens in 700 BCE (matrioshka/Shutterstock)*

With modern supercomputers and the appropriate software, humankind now has the ability to design and accurately predict the future performance of new products, as well as the future behavior of natural systems. Like those just cited, starting with the Greeks, this is a historic advance in our ability to develop new and exciting technologies and to better understand our world and the universe quantitatively. Some technology leaders place the advent of advanced computing as part of the Fourth Industrial Revolution (Schwab 2016). It is our view that what has happened is far more significant. Computational engineering now allows accurate predictions of the future performance and behavior of some of the most complex products ever conceived—*before* they are built. Computational science gives us the capability to ask and answer fundamental questions about natural phenomena such as the weather, the climate, and the structure, history, and future of the universe that we could not address before now.

Index

D

E

F

G

H

I

J-K

L-M

N

O

P

Q-R

S

T

U

V

W-X-Y-Z