35

Where to Find More Information

CC2E.COM/3560

Contents

Related Topics

List of Additional Resource sections: page TBD

Web resources: *www.cc2e.com*

IF YOU'VE READ THIS FAR, you already know that a lot has been written about effective software development practices. Much more information is available than most people realize. People have already made all the mistakes that you're making now, and unless you're a glutton for punishment, you'll prefer reading their books and avoiding their mistakes to inventing new versions of old problems.

Because this book describes hundreds of other books and articles that contain articles on software development, it's hard to know what to read first. A software-development library is made up of several kinds of information. A core of programming books explains fundamental concepts of effective programming. Related books explain the larger technical, management, and intellectual context within which programming goes on. And detailed references on the languages, operating systems, environments, and hardware contain information that's useful for specific projects.

CC2E.COM/3581

Books in the last category generally have a life span of about one project; they're more or less temporary and aren't discussed here.

Of the other kinds of books, it's useful to have a core set that discusses each of the major software-development activities in depth—books on requirements, design, construction, management, testing, and so on. The following sections describe construction resources in depth, and then provide an overview of materials available in other software knowledge areas. Section 35.4 wraps these resources into a neat package by defining a software developer's reading program.

35.1 Information About Software Construction

CC2E.COM/3588

I originally wrote this book because I couldn't find a thorough discussion of software construction. In the years since I published the first edition, several good books have appeared.

Pragmatic Programmer (Hunt and Thomas 2000) focuses on the activities most closely associated with coding including testing, debugging, use of assertions, and so on. It does not dive deeply into code itself, but contains numerous principles related to creating good code.

Jon Bentley's *Programming Pearls, 2d Ed* (Bentley 2000) discusses the art and science of software design in the small. The book is organized as a set of essays that are very well written and express a great deal of insight into effective construction techniques as well as genuine enthusiasm for software construction. I use something I learned from Bentley's essays nearly every day that I program.

CROSS-REFERENCE For more in the economics of extreme programming and agile programming, see *cc2e.com/3545*.

Kent Beck's *Extreme Programming Explained: Embrace Change* (Beck 2000) defines a construction-centric approach to software development. As Section 3.1 explained, the book's assertions about the economics of extreme programming are not borne out by industry research, but many of its recommendations are useful during construction regardless of whether a team is using extreme programming or some other approach.

A more specialized book is Steve Maguire's *Writing Solid Code – Microsoft's Techniques for Developing Bug-Free C Software* (Maguire 1993). It focuses on construction practices for commercial-quality software applications, mostly based on the author's experiences working on Microsoft's Office applications. It focuses on techniques applicable in C. It is largely oblivious to object-oriented programming issues, but most of the topics it addresses are relevant in any environment.

Another more specialized book is *The Practice of Programming* by Brian Kernighan and Rob Pike (Kernighan and Pike 1999). This book focuses on nitty gritty, practical aspects of programming, bridging the gap between academic computer science knowledge and hands-on lessons. It includes discussions of programming style, design, debugging, and testing. It assumes familiarity with C/C++.

Although it’s out of print and hard to find, *Programmers at Work* by Susan Lammers (1986) is worth the effort if can find it. It contains interviews with the industry’s high-profile programmers. The interviews explore their personalities, work habits, and programming philosophies. The luminaries interviewed include Bill Gates (founder of Microsoft), John Warnock (founder of Adobe), Andy Hertzfeld (principal developer of the Macintosh operating system), Butler Lampson (a senior engineer at DEC, now at Microsoft), Wayne Ratliff (inventor of dBase), Dan Bricklin (inventor of VisiCalc), and a dozen others.

35.2 Topics Beyond Construction

Beyond the core books described in the last section, here are some books that range further afield from the topic of software construction.

CC2E.COM/3595

Overview Material

Robert L. Glass’s *Facts and Fallacies of Software Engineering* (2003) provides a readable introduction to the conventional wisdom of software development dos and don’ts. The book is well researched and provides numerous pointers to additional resources.

My own *Professional Sofware Development* (2004) surveys the field of software development as it is practiced now and as it could be if it were routinely practiced at its best.

The *Swebok: Guide to the Software Engineering Body of Knowledge* (Abran 2001) provides a detailed decomposition of the software engineering body of knowledge. This book has dived into detail in the software construction area. The Guide to the Swebok shows just how much more knowledge exists in the field.

Gerald Weinberg’s *The Psychology of Computer Programming* (Weinberg 1998) is packed with fascinating anecdotes about programming. It’s far-ranging because it was written at a time when anything related to software was considered to be about programming. The advice in the original review of the book in the *ACM Computing Reviews* is as good today as it was when the review was written:

> *Every manager of programmers should have his own copy. He should read it, take it to heart, act on the precepts, and leave the copy on his desk to be stolen by his programmers. He should continue replacing the stolen copies until equilibrium is established (Weiss 1972).*

If you can't find *The Psychology of Computer Programming*, look for *The Mythical Man-Month* (Brooks 1995) or *PeopleWare* (DeMarco and Lister 1999). They both drive home the theme that programming is first and foremost something done by people and only secondarily something that happens to involve computers.

A final excellent overview of issues in software development is *Software Creativity* (Glass 1995). This book should have been a breakthrough book on software creativity the way that *Peopleware* was on software teams. Glass discusses creativity versus discipline, theory versus practice, heuristics versus methodology, process versus product, and many of the other dichotomies that define the software field. After years of discussing this book with programmers who work for me, I have concluded that the difficulty with the book is that it is a collection of essays edited by Glass, but not entirely written by him. For some readers, this gives the book an unfinished feel. Nonetheless, I still require every developer in my company to read it. The book is out of print and hard to find, but worth the effort if you are able to find it.

Software-Engineering Overviews

Every practicing computer programmer or software engineer should have a high-level reference on software engineering. Such books survey the methodological landscape rather than painting specific features in detail. They provide an overview of effective software-engineering practices and capsule descriptions of specific software-engineering techniques. The capsule descriptions aren't detailed enough to train you in the techniques, but a single book would have to be several thousand pages long to do that. They provide enough information so that you can learn how the techniques fit together and can choose techniques for further investigation.

Roger S. Pressman's *Software Engineering: A Practitioner's Approach, 6th Ed.* (Pressman 2004) is a balanced treatment of requirements, design, quality validation, and management. Its 700 pages pay little attention to programming practices, but that is a minor limitation, especially if you already have a book on construction such as the one you're reading.

The 6th edition of Ian Sommerville's *Software Engineering* (Sommerville 2000) is comparable to Pressman's book, and it also provides a good high-level overview of the software-development process.

CC2E.COM/3502

Other Annotated Bibliographies

Good computing bibliographies are rare. Here are a few that justify the effort it takes to obtain them.

ACM Computing Reviews is a special-interest publication of the ACM that's dedicated to reviewing books about all aspects of computers and computer programming. The reviews are organized according to an extensive classification scheme, making it easy to find books in your area of interest. For information on this publication and on membership in the ACM, write: ACM, PO Box 12114, Church Street Station, New York, NY 10257.

CC2E.COM/3509

Construx Software's Professional Development Ladder (*www.construx.com/ladder/*). This website provides recommended reading programs for software developers, testers, and managers.

35.3 Periodicals

Lowbrow Programmer Magazines

These magazines are often available at local newsstands.

CC2E.COM/3516

Software Development. *www.sdmagazine.com*. This magazine focuses on programming issues—less on tips for specific environments than on the general issues you face as a professional programmer. The quality of the articles is quite good. It also includes product reviews.

CC2E.COM/3523

Dr. Dobb's Journal. *www.ddj.com*. This magazine is oriented toward hard-core programmers. Its articles tend to deal with detailed issues and include lots of code.

If you can't find these magazines at your local newsstand, many publishers will send you a complimentary issue, and many articles are available on line.

Highbrow Programmer Journals

You don't usually buy these magazines at the newsstand. You usually have to go to a major university library or subscribe to them for yourself or your company.

CC2E.COM/3530

IEEE Software. *www.computer.org/software/*. This bimonthly magazine focuses on software construction, management, requirements, design and other leading-edge software topics. Its mission is to "build the community of leading software practitioners." In 1993, I wrote that it's "the most valuable magazine a programmer can subscribe to." Since I wrote that, I've been Editor in Chief of the magazine, and I still believe it's the best periodical available for a serious software practitioner.

CC2E.COM/3537

IEEE Computer. *www.computer.org/computer/*. This monthly magazine is the flagship publication of the IEEE Computer Society. It publishes articles on a wide spectrum of computer topics and has scrupulous review standards to ensure the quality of the articles it publishes. Because of its breadth, you'll probably find fewer articles that interest you than you will in *IEEE Software*.

CC2E.COM/3544

Communications of the ACM. *www.acm.org/cacm/*. This magazine is one of the oldest and most respected computer publications available. It has the broad charter of publishing about the length and breadth of computerology, a subject that's much vaster than it was even a few years ago. As with *IEEE Computer*, because of its breadth, you'll probably find that many of the articles are outside your area of interest. The magazine tends to have an academic flavor, which has both a bad side and a good side. The bad side is that some of the authors write in an obfuscatory academic style. The good side is that it contains leading-edge information that won't filter down to the lowbrow magazines for years.

Special-Interest Publications

Several publications provide in-depth coverage of specialized topics.

Professional Publications

CC2E.COM/3551

The IEEE Computer Society publishes specialized journals on software engineering, security and privacy, computer graphics and animation, internet development, multimedia, intelligent systems, the history of computing, and other topics. See *www.computer.org* for more details.

CC2E.COM/3558

The ACM also publishes special-interest publications in artificial intelligence, computers and human interaction, databases, embedded systems, graphics, programming languages, mathematical software, networking, software engineering, and other topics. See *www.acm.org* for more information.

Popular-Market Publications

These magazines all cover what their names suggest they cover.

CC2E.COM/3565

The C/C++ Users Journal. *www.cuj.com.*

CC2E.COM/3572 *Java Developer's Journal*. *www.sys-con.com/java/*.

CC2E.COM/3579 *Embedded Systems Programming*. *www.embedded.com.*

CC2E.COM/3586 *Linux Journal*. *www.linuxjournal.com*.

CC2E.COM/3593 *Unix Review*. *www.unixreview.com*

CC2E.COM/3500 *Windows Developer's Network*. *www.wd-mag.com*.

35.4 A Software Developer's Reading Plan

CC2E.COM/3507 This section describes the reading program that a software developer needs to work through to achieve full professional standing at my company, Construx Software. The plan described is a generic baseline plan for a software professional who wants to focus on development. Our mentoring program provides for further tailoring of the generic plan to support an individual's interests, and within Construx this reading is also supplemented with training and directed professional experiences.

Introductory Level

To move beyond "introductory" level at Construx, a developer must read the following books.

Adams, James L. *Conceptual Blockbusting: A Guide to Better Ideas*, 4th ed. Cambridge, Mass.: Perseus Publishing.

Bentley, Jon. *Programming Pearls, 2d Ed*. Reading, Mass.: Addison-Wesley, 2000.

Glass, Robert L. *Facts and Fallacies of Software Engineering*, Boston, Mass.: Addison Wesley, 2003.

McConnell, Steve. *Software Project Survival Guide*. Redmond, WA: Microsoft Press, 1998.

McConnell, Steve. *Code Complete, 2d Ed.*. Redmond, WA: Microsoft Press, 2004.

Practitioner Level

To achieve "intermediate" status at Construx, a programmer needs to read the following additional materials.

Berczuk, Stephen P. and Brad Appleton. *Software Configuration Management Patterns: Effective Teamwork, Practical Integration*, Boston, Mass.: Addison Wesley, 2003.

Fowler, Martin. *UML Distilled: A Brief Guide to the Standard Object Modeling Language, 3d Ed*, Boston, Mass.: Addison Wesley, 2003.

Glass, Robert L. *Software Creativity*, Reading, Mass.: Addison Wesley, 1995.

Kaner, Cem, Jack Falk, Hung Q. Nguyen. *Testing Computer Software, 2d Ed.*, New York: John Wiley & Sons, 1999.

Larman, Craig. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process*, 2d Ed., Englewood Cliffs, N.J.: Prentice Hall, 2001.

McConnell, Steve. *Rapid Development*. Redmond, WA: Microsoft Press, 1996.

Wiegers, Karl. *Software Requirements*, 2d Ed. Redmond, WA: Microsoft Press, 2003.

CC2E.COM/3514

“Manager’s Handbook for Software Development”, NASA Goddard Space Flight Center. Downloadable from *sel.gsfc.nasa.gov/website/documents/online-doc.htm*.

Professional Level

A software developer must read the following materials to achieve full professional standing at Construx (“leadership” level). Additional requirements are tailored to each individual developer; this section describes the generic requirements.

Bass, Len, Paul Clements, and Rick Kazman. *Software Architecture in Practice*, Second Edition, Boston, Mass.: Addison Wesley, 2003.

Fowler, Martin. *Refactoring: Improving the Design of Existing Code*, Reading, Mass.: Addison Wesley, 1999.

Gamma, Erich, et al. *Design Patterns*, Reading, Mass.: Addison Wesley, 1995.

Gilb, Tom. *Principles of Software Engineering Management*. Wokingham, England: Addison-Wesley.

Maguire, Steve. *Writing Solid Code*. Redmond, WA: Microsoft Press, 1993.

Meyer, Bertrand. *Object-Oriented Software Construction, 2d Ed*. New York: Prentice Hall PTR, 1997.

CC2E.COM/3521

"Software Measurement Guidebook", NASA Goddard Space Flight Center. Available from *sel.gsfc.nasa.gov/website/documents/online-doc.htm*.

CC2E.COM/3528

For more details on this professional development program, as well as for up-to-date reading lists, see our professional development website at *www.construx.com/professionaldev/*.

35.5 Joining a Professional Organization

CC2E.COM/3535

One of the best ways to learn more about programming is to get in touch with other programmers who are as dedicated to the profession as you are. Local user groups for specific hardware and language products are one kind of group. Other kinds are national and international professional organizations. The most practitioner-oriented organization is the Computer Society of the IEEE (Institute of Electrical and Electronics Engineers). The IEEE Computer Society publishes the *IEEE Computer* and *IEEE Software* magazines. For membership information, see *www.computer.org*.

CC2E.COM/3542

The original professional organization was the Association for Computing Machinery, or ACM. The ACM publishes *Communications of the ACM* and many special-interest magazines. It tends to be somewhat more academically oriented than the IEEE Computer Society. For membership information, see *www.acm.org*.