Greg Andrews and *The (Book) Review*: A Cautionary Tale

David E. Bakken, Ph.D. (UA 1994),

Washington State University,

January 21, 2010

Abstract

This document describes the hoax pulled on April 1, 1992, on Prof. Greg Andrews by the author of this short note, who was then a graduate student in Greg's department. Namely, it was a fake review in *IEEE Computer* of his (actually quite good) book on concurrent programming, *Concurrent Programming: Principles and Practice*. The review was written with the goals of being rather off-the-wall, by someone obviously unqualified to pass technical judgment on Greg's book, and by a Canadian version of Archie Bunker. Nevertheless, the review somehow faked out Greg and went down in departmental folklore (at least among the graduate students). This document is being written as input to a retirement roast of Greg on January 21, and will serve to preserve this successful hoax for posterity (at least if posterity is defined as the lifetime of Google's archives).

Contents

BACKGROUND **2**

CREATION OF *THE REVIEW* **2**

OMINOUS FORESHADOWING 2
COLOPHONY 3
THE WORD GETS OUT: ADVANCED CAUTIONING BY PROFESSOR RICK SCHLICHTING 3

DESIGN OF *THE REVIEW* **4**

OFF-THE-WALL 4
INCOMPETENT REVIEWER 5
CANADIAN ARCHIE BUNKER 5

APRIL 1, 1992: D-DAY FOR *THE REVIEW* **5**

AFTERMATH OF *THE REVIEW* **6**

LESSONS LEARNED FROM *THE REVIEW* 7
A GENEROUS YET MODEST OFFER 7

REFERENCES **7**

APPENDIX **8**

ANOTHER JOKE ON GREG (AND ALMOST RICK) 8
SCANNED VERSION OF *THE REVIEW* 8

Background

(This document is written in the first person …. Apologies for this scandalous informality for an academic, but it is just way too much indirect grammar otherwise. And even the "royal we" seems silly here, ergo I am sticking with first person singular.)

This document is organized as follows. First, I provide pertinent background information for understanding the context of The Review. Next, I describe its creation: how I foreshadowed it, how it was produced, and goals in writing it. Next I describe what transpired on the day of the review. After this, I discuss the aftermath of the review, and lessons learned. An appendix then briefly describes another joke I successfully pulled on Greg, and gives the 3 pages (two TOC) constituting The Review that I put in his physical mailbox on that glorious morning of April 1, 1992. But now for "the rest of the story"…

I became a computer science graduate student at The University of Arizona in the fall of1988, and immediately gravitated to concurrent programming and Greg Andrews, and was planning on doing a Ph.D. under his supervision, looking at "portable parallelism" or similar issues. I had a programming job with Greg working for about a year parallelizing the runtime system [TB94]. of Greg's concurrent programming language, SR (which at the time was threaded—co-routines more precisely—but did not exploit multiple physical CPUs). In this context I had weekly meetings with Greg and Gregg Townsend.

Probably in my third or fourth semester (AY89-90), I took a course on fault-tolerant computing from Rick Schlichting, and I decided that my heart was really into distributed fault tolerance, not concurrent programming. So I was secluded by the dark side of the force (w.r.t concurrent programming), and switched advisors and topics (though Greg and I remained on very good terms and he on my committee).

Fortune smiled on this hoax in a number of ways. First, it turned out to be very fortuitous that I had not been an advisee of his for a year or two when I pulled off the hoax. Even after Greg realized that the review was fake, as I describe below, it took him some time (a few hours I believe) to realize that I did it, because I was not meeting with him regularly at that point in my graduate studies. Also, his textbook [And91] ended up taking a few more years for him to finish than he anticipated when I took the concurrent programming class from him (I think in Fall 1988). This made him even more vulnerable to a hoax (IMO; YMMV) and certainly made him more irritated when some incompetent reviewer savaged his book in a negative review.

Creation of *The Review*

Ominous Foreshadowing

I had known for a few years that I wanted to do a fake review on him (um, maybe those few years were the ones his book was over schedule!). So I provided some ominous foreshadowing to the review. I knew that Greg was a member of the ACM, but (at least at the time) not of the IEEE. This was fortuitous in two ways. First, I knew that if he got a fake review from an IEEE magazine in his mailbox he would not have the original handy to compare it to. Second, a few times I admonished him that some day he was "going to get burned" by not being an IEEE member. Boy, did that irritate him—here was a graduate student lecturing a professor on professional society membership—but as a matter of

principle (not just for mischief!) I had to say that to give him a small clue (and, OK, have fun in doing so; that falls under the category "nice work if you can get it").

That should have been a clue for Greg, but, then, I would also (as a matter of principle) razz him any time my beloved WSU would play either Washington or Stanford in football[1]—even if I knew were probably going to get creamed—so maybe that diminished the value of the clue for him. Or maybe he was just plain gullible, at least that day[2].

Ironically, way before The Review he had some clue that I was mischievous. In the welcoming picnic the first week or so of classes in Fall 1988, we were playing volleyball (of course!), and Greg and I were on opposite teams. When I served, Greg happened to be in the back row on the other side, I did the first one underhanded (I had never really played the game before). For the second serve, I decided to try an overhand serve, hitting it has hard as I could, and then seeing what happened. I was not aiming for anything (not that I had the accuracy had I been aiming!), but was very hard and flat, barely above the net and then hitting squarely in Greg's groin. He apparently thought it was on purpose: after the game he made it a point to come up to me and (somewhat ruefully) say "nice serve". Rats, I must have forgotten to tell him it was actually an accident, but that should have given him a small clue.

Colophony

When I did the review, WYSYWIG editors were not very common, and I was quite happy using LaTeX anyway. So what I did was to typeset the verbiage of my review in LaTeX, cut it out, and then tape it over a photocopy of the actual first page of book reviews in that issue. Because[3] *The Review* did not take up a full page, I typeset the first part of the review that was actually there starting on the bottom of that page.

What I planned to deliver to his box, then, was this review page plus two other pages with the table of contents for that issue, giving proof to Greg (remember he was not an IEEE member….) that it was the Real McCoy.

Interestingly, if one takes the time to look very closely at the actual review (in the Appendix), one can see that the columns are not perfectly lined up (I had to tape the columns in individually, for some reason I cannot remember now 18 years later). But, again, fortune smiled on this hoax: *ipso facto* Greg did not have the time to notice that stuff closely, as described below. Some things are just meant to be….

The Word Gets Out: Advanced Cautioning by Professor Rick Schlichting

I worked on the review verbiage on the afternoon of March 31, and into the early evening. A few fellow graduate students knew I was doing this, and word got to Prof. Rick Schlichting (who worked closely

[1] Greg is a native of Washington state (Olympia), and actually has ties to WSU through his father. Greg did his undergraduate studies at Stanford. He did his graduate studies at the University of Washington, whose Husky football team is the rival, *bête noire*, and arrogant big brother to my beloved WSU Cougar football team.

[2] Breaking news: Oxford has announced that the word "gullible" will be removed from the next version of the OED.

[3] Before working with Greg, I would have started this sentence with "Since". However, he drilled into my head that "since" means after some point in time, while "because" is more appropriate due to its connotation of cause and effect. To this day, I am grateful for that, and for him drilling into me the evils of split infinitives (of which I still sin via, and probably there are a few in this document).

with Greg). Rick warned me "*You'd better make sure that he knows it's a fake, Bakken!*". So I went overboard and made it even crazier than I was originally intending to.

As a sanity check, I showed my final verbiage to Peter Druschel, who was working that evening in the cubicle farm, to make sure that Greg would realize it's a fake. Peter said the review was "crazy" and no way would Greg be fooled by it.

Design of *The Review*

My strategies in writing the review were threefold: make it very off the wall, make it clear that the reviewer was technically incompetent to pass judgment on the book, and to make the reviewer out to be a Canadian version of Archie Bunker. In case you have to be told, I am a smart aleck, so I would have done much of this anyway, but I went overboard after Rick's cautionary advice. It was a great excuse, not that I needed one.

We now examine these areas of going overboard in turn.

Off-The-Wall

The review started out quite reasonably for the first two paragraphs (indeed, the first paragraph calls the book a *tour de force* and the second paragraph, which outlines the book's major parts, ripped phrases and I think sentences directly from the inside cover of the book). Most of the first two paragraphs were taken verbatim from the book's jacket cover, which should have tipped Greg off, but did not.

And the first sentence of the third paragraph is quite reasonable. But, then, the review quickly goes off the deep end, including:

- It has very harsh language that even the most severe of reviews would never use (even if the reviewer was tactless enough to write it, no way would the editor of the book reviews allow it to be published).
    - It says that the book "fails miserably", which is harsh language. *The Review* even rants about a secretary or even WordPerfect having to do concurrent programming yet not being helped by the book.
    - It gets very personal against Greg (post-childhood trauma, saying Ada is way better for concurrent programming than SR, talking about his son's education, etc).
    - It ends by saying that Greg's book is "utterly without any redeeming values".
- The review says that the sales of this book won't meet Greg's goals about paying for his sons' education, but it may pay for a few weeks of college food.
- The name of the reviewer is the name of the Canadian Prime Minister from 1957–1963. Greg is old enough that he should have recognized this name, even had he not been from a border state. I mean, Diefenbaker died in 1979, the year I was graduated from high school, and Greg is probably 10 years older than me.
- The reviewer's ostensible affiliation was with "Simulé (fake) University".
- That university was in "Moose Jaw", a town I made up (and whose province was not given, which should have been yet another mini-clue). I mean, how stereotypical of a name can you get?

- Update: a few years after Greg's 2010 retirement roast, to my horror I found out that there actually is a town in Canada named Moose Jaw. I alerted Greg immediately. I was concerned that there was some cosmic consequence to my using that name as such, perhaps a rip in the space-time continuum. Which, actually, as long as it happened in Montlake, might not be so bad. Indeed, many times when I fly from Pullman into Seattle—usually with an approach from the north—I try to convince the Horizon Air crew to dump the bilge onto Husky Stadium. Think about it: it would improve both the airplane and the field! But so far I have not convinced them to do it.

Incompetent Reviewer

The Review also tried to make it clear the reviewer was incompetent. Ways this were done included:

- *The Review* whines about a COBOL or BASIC or FORTRAN programmer with zero or one programming class not being able to do concurrent programming with this book.
- *The Review* talks about computer historians.
- The reviewer is an MIS guy not a CS guy.
- The reviewer talks about "us normal folks" who can't use the book.

Canadian Archie Bunker

The reviewer was finally made out to be a Canadian version of Archie Bunker (of *All In The Family* fame). In particular, in one rant about the book not being able to pay for Greg's sons' college or even two weeks of college food, the reviewer speculates that it may be able to pay for:

- A bag of pork rinds (President (#41) George H. W. Bush's favorite food; Bush was in office in April 1992)
- Some jelly beans (President (#40) Ronald Regan's favorite food)
- Some grits (President (#39) Jimmy Carter's favorite food, or at least one stereotyped to him in this way)

April 1, 1992: D-Day for *The Review*

On that fateful morning, I got in early, slipped *The Review* (and two TOC pages stapled to it) in Greg's mailbox, then avoided being in the department. That was the plan, even though I didn't think it would fool him.

However, Greg was department chair back then. So he got in to work that day probably with hundreds of things to do and a big inbox. And he probably even read it before his first cup of coffee. Or, maybe his is just gullible (c.f. the other joke I played on him, described in the Appendix; that lends credence to this theory).

But, whatever the reason, I am told that he fell for it hook, line, and sinker. It supposedly took him a few hours to realize it was a fake. During that time I am told Rick consoled him about the review and gently suggested that he look at the calendar (i.e., to notice that the day was April Fools).

During those hours of despair for Greg—which must have felt like years or even decades to him—I had to duck back into the department; I think to get a book from my locker. I happened to walk by Greg in

the hallway. Boy, he seemed to be in the foulest mood that I have ever seen anyone in, short of after a death in the family (which this potentially was, of course, for his book). Actually, come to think of it, his mood was really fouler than most I've seen due to a death in the family!

I am told that Greg wrote off the review as a crazy fluke, but after some hours he realized it was a fake. But, then, for some hours he did not know who wrote it. Certainly he interrogated (and maybe even tortured/waterboarded) his current advisees, but recall that by then I was not his advisee for the last year or two and thus off of his immediate radar screen.

But, finally, that afternoon I had to come back into the department to get more things and study. By then he had figured out that I had done it. I was studying on the big table by the restrooms when Greg walked by to use them. He just shook his head with a rueful smile and said that I had missed my calling in life: creative writing.

Aftermath of *The Review*

A day or two later I posted the review outside one of our grad student cubicle farms (which was the big room on the east side of the 7th floor of Gould-Simpson building). I wrote clearly in large red letters that it was not a real review. So then everyone else in the department who heard about it now could read it. I am told that this episode has gone down in departmental folklore.

A put a vague allusion to *The Review* in the Acknowledgements page of the penultimate draft of my dissertation. However, for some inexplicable reason Greg (who was on my PhD committee) didn't like that too much, so I removed it. ☹ But later I wrote this document. ☺

In the year or so after, I sent a copy to a few colleagues who where in the department but had moved on. One of these was Professor Norm Hutchinson. His email tersely said "This is a masterpiece!" and asked permission to send a copy of *The Review* to Professor Stella Atkins (I think a former student of Greg's; of course I said yes!).

Professor Curtis Dyreson (then a graduate student like me) told me circa 2005 that he and Dave Lowenthal (a PhD student of Greg's on D-Day) and Greg were at some conference or meeting, and they were still chuckling about this incident. When I visited Greg briefly when he worked at the National Science Foundation while I was in Washington DC, he also was still shaking his head over the whole thing (fortunately with a rueful smile on his face—do you see a pattern here: Greg saying something to me with a rueful smile on his face?).

I ran into Larry Peterson few years ago at a DARPA or similar event. He was there with a UA CS professor, I forget whose name. Right after Larry introduced him with my name, he said that this was the guy who did *The Review*. ☺ I sent him the attached writeup, and his comment the next day was that my fictional reviewer was not just incompetent (per my design) but also insane! I consider this to be not only true, but also a fairly significant result in anthropology, though YMMV. And, if not, pretty fun.

I sent this document to Rick Snodgrass and a few others in 2013. Rick said "I don't think I've laughed as much reading any other email from the last 35 years, that is, since I started using email. … I especially liked the officious tone of your essay, including the typography and section titles. And many aspects of The Review were just brilliant, such as the references to presidents and the affiliation of the reviewer."

I hope to send this into the Museum of Hoaxes. I'll certainly have to modify it a bit, making it more stand-alone for non-programmer types and also perhaps eventually removing the widespread usage of the first person (which may make it a lot less fun—really more stuffy—to read).

Lessons Learned from *The Review*

The subtitle of this document is that it is "A Cautionary Tale". Why is it worded in this way? There is some debate about this (at least there was between my imaginary friends[4]), but speculation includes the following:

- It sounded very sexy to me so I shamelessly used it. Hey, it got your attention, right?
- There are lessons here to be learned about
    - Greg,
    - Me,
    - Trust (not unlike Ken Thompson's Turing Award speech "Reflection on Trusting Trust"), or
    - Life in general.

Speculation, anyone? Inquiring minds want to know….

A Generous yet Modest Offer

I offer that I am hereby quite happy to write a book review for anyone in his area of computer science who needs one. To be a good sport, I will even throw in a set of Ginzu knives. And, if I end up writing a textbook like he might in a few years, as a matter of principle I will insist that Greg do a review of it for IEEE Computer. Because this review would be reviewed, Greg wouldn't be able to get away with anything nearly as wild as I did. But maybe he can slip in a few Carrolian-like allusions or slip in other double (or even triple) entendres, perhaps involving a husky or a cougar.

Dave Bakken (bakken@wsu.edu & dave.bakken@gmail.com)
January 2010 (updated slightly in June 2013)
Pullman, Washington, USA

References

[And91] Andrews, Greg. Concurrent Programming: Principles and Practice. Benjamin/Cummings, 1991, ISBN 0-8053-0086-4

[TB94] Townsend, Gregg and Bakken, Dave. "Porting MultiSR". In *Porting the SR Programming Language*. Department of Computer Science, The University of Arizona, 1994. From the SR distribution http://www.cs.arizona.edu/sr/.

[4] Like a great tee shirt says: "I used to have imaginary friends, but my therapist took them away". ☺

Appendix

Another Joke on Greg (and Almost Rick)

I had previously pulled a joke on Greg, that arguably should have given him some clue. It was based on a vulnerability: Greg (probably by oversight) had his permissions set on his X-Windows display device such that anyone in the department could write to it. And, hey, X was designed for a remote application to display….

I had weekly meetings with him (and Gregg and I think sometimes others, including Dave Lowenthal). So I timed how long it took to get from my office to his (on the same floor), then added in a fudge factor of 10 seconds. I then took an open source X-Windows program that made the contents of a screen appear to be melting (I think the program was called *decay* or *ferck* or something like that), and added in that delay for the melting to start. Also, in the main melting loop, I put a five second delay for the first few iterations. That way, the rendering of his windows would "melt" only by a few pixels, then wait five seconds. But after maybe 15-20 seconds the delay was no longer there, so the melting happened fast.

To pull it off, I planned to come one minute late to the meeting (in Greg's office). I started the hacked program then walked down the hall. When I got to his office, I looked over his monitor and touched it with a concerned look on my face. I said with fake concern that his monitor seemed to get hot. He looked at it with moderate alarm, and saw it "melt" a few pixels at first then more. After he was faked out for a bit I told him.

For posterity, I note that I had a big practical joke planned on my advisor, Rick Schlichting in early 1994. However, it never got past the initial planning stages before my defense in July 1994, because I was just too busy. I thought about doing it in the year after, but decided that, as a matter of principle, I should try to not do a major practical joke on someone unless they have signature power over my dissertation or similar means to mess with me in a big way.

Scanned Version of *The Review*

On the following 3 pages are the contents of *The Review*. These are a scan of an identical photocopy of what I put in his box on that fateful day.

Following that is a scan of the actual review page in that issue. When I first got the idea to include it in this document for fun, I was aghast to find that the IEEE only has part of that issue available online, and not the review page. However, I noticed in my office that the one hardcopy magazine I have from years ago is *IEEE Computer*, and I had that issue. I really only kept them so I would have some hardcopy magazines to go with my books for a very "professorial" look in my office. But, apparently, not only was this hoax meant to be, but this accounting of it for posterity was meant to be, too, and to be complete at that!

BOOK REVIEWS

Editor: Alan Kaminsky, Rochester Institute of Technology, PO Box 9887, Rochester, NY 14623-0887
(716) 475-5255; Internet ark@cs.rit.edu;Bitnet arkics@ritvax.

Concurrent Programming: Principles and Practice

Gregory R. Andrews (Benjamin/Cummings,Redwood City, Calif., ISBN 0-8053-0086-4,637 pp., $47.75)

As sequential computers are rapidly approaching the limits imposed on them by the speed of light, parallel computers are becoming increasingly important. Indeed, many (including this reviewer) believe that by the close of this decade the term "computing" will be synonymous with "parallel computing." It is in this context that Andrews brings us this *tour de force.*

The book is organized into four main sections. Part I covers basic concepts including useful programming notations and logics for sequential and concurrent programming. Here the reader is immersed into the world of formal program verification; more on this later. Part II presents a systematic method for solving synchronization problems and shows how to apply the method to realworld problems. Here the reader learns how to implement communication and synchronization through shared variables, semaphores, and monitors. Part III explains how to apply the method introduced in Part II to distributed programs in which communication and synchronization are based on message passing. It covers asynchronous and synchronous message passing, remote procedure call, and rendezvous. Finally, Part IV surveys a variety of concurrent programming languages and models, including Turing Plus, Occam, Ada, SR (the author's language), and Linda.

The basic goal of this textbook cum historical reference is to help programmers of all stripes better utilize parallelism. In this vein it fails miserably, however. I see no way that it will empower COBOL or BASIC programmers to achive their parallel programming potential. Likewise, it offers no hope for the BSEE with one FORTRAN course (the median software engineer) to become an expert in parallelism. Ditto for the secretary programming in WordPerfect (or, for that matter, the WordPerfect program programming in PostScript). If inflicted on them, Andrews' book is even likely to constitute a post-childhood trauma for all graduate students not specializing in parallel research. It is thus bound to be an utter commercial failure. And I find it fascinating that Andrews chose to describe both his SR and Ada in the same book, since Ada is clearly more powerful, expressive, and easier to master. This editorial mistake will undoubtedly sabatoge sales of his promised book on SR.

The book does have a well-defined niche that it will fill quite richly, however. Each chapter has exhaustive historical notes and references. These will undoubtedly be viewed as an invaluable source by computer historians centuries from now. Unfortunately for Andrews, there aren't that many computer historians alive today.

Indeed, in the preface Andrews expresses his hope that royalties from this book will help pay for his two sons' educations. Unfortunately for him, this is not likely to be so, although it may pay for a few weeks of college food. Or a bag of pork rinds. Or maybe some jelly beans and grits.

In summary, this is a very useful and important book, but only to a handful of people. For them this book represents a rich mother lode of knowledge. For the rest of us normal folks it is utterly without any redeeming values.

John "Chief" Diefenbaker, Dean
College of Information Science
Simulé University
Moose Jaw, Canada

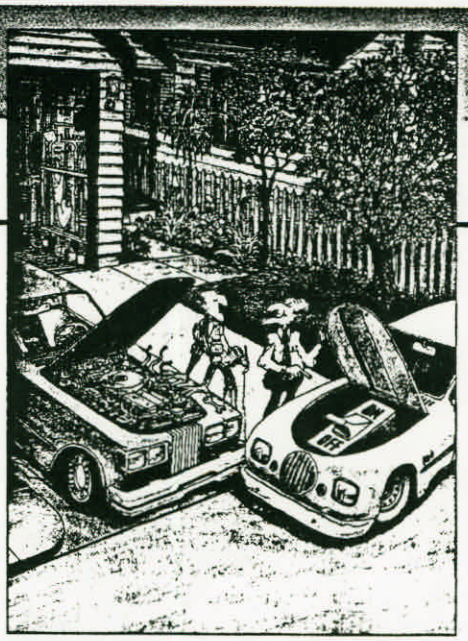
Software Engineering with Abstractions

Valdis Berzins and Luqi (Addison-Wesley, Reading, Mass., ISBN 0-201-08004-4,624 pp., $35)

In their preface, Berzins and Luqi note the difficulty of convincing new computer science students of the importance of systematic methods and formal specifications because of their experience in programming on a relatively small scale. Certainly, the widespread and cheap compilers gives many new students the sense that they already know what software development is, sometimes even better than their professors. They think of it as programming or, even worse, coding.

The basic goal of this textbook for an advanced one-year course in software engineering is to demonstrate the use of modern formal methods. The authors emphasize approaches amenable to computer assistance.

They concentrate on "a consistent treatment of the entire software development process and strive to present a single approach with sufficient depth to let the readers carry out the process, with concern on



Cover: Jay Simpson, Design & Direction

DEPARTMENTS

4 Computer Society Message
Technical activities within the society

10 Authentication Revisited

81 Update

83 Computer Society News
Nominations sought for IEEE fellowship

86 Product Reviews
Word processors

94 New Products

98 IC Announcements

100 Microsystem Announcements

103 Conferences

106 Call for Papers

108 Calendar

125 Book Reviews

128 Open Channel

Career Opportunities, 117; Computer Society Information, 127; Membership Application, 99; Advertiser/Product Index, 80; Reader Service Card, 80A

In the next issue

Wafer-scale integration

COMPUTER

10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA 90720-1264

Submissions

Send 12 double-spaced copies of articles and special-issue proposals to Jon T. Butler, Dept. of Electrical and Computer Engineering, Code EC/Bu, Naval Postgraduate School, Monterey, CA 93943-5004, phone (408) 646-3299. For electronic submission, Butler's addresses are j.butler on Compmail II and butler@ece.nps.navy.mil on Internet.

COMPUTER

March 1992 • Vol. 25, No. 3

ARTICLES

12 The Usability Engineering Life Cycle
Jakob Nielsen
A usability engineering process to ensure good user interfaces includes elements to be considered before the design, during the design, and after field installation of a software product.

25 SoundWorks: An Object-Oriented Distributed System for Digital Sound
Jonathan D. Reichbach and Richard A. Kemmerer
SoundWorks lets users interactively manipulate sound through a graphical interface. The system handles digitally sampled sounds as well as those generated by software and digital signal processing hardware.

38 Mediators in the Architecture of Future Information Systems
Gio Wiederhold
Mediators embody the administrative and technical knowledge to create information needed for user decision-making modules. The goal is to exploit the data technology puts within our reach.

50 A Taxonomy and Current Issues in Multidatabase Systems
M.W. Bright, A.R. Hurson, and Simin H. Pakzad
Global access and local autonomy are central to multidatabase system design. This article draws on other information-sharing systems research to define key issues.

63 The Stanford Dash Multiprocessor
Daniel Lenoski, James Laudon, Kourosh Gharachorloo, Wolf-Dietrich Weber, Anoop Gupta, John Hennessy, Mark Horowitz, and Monica S. Lam
Directory-based cache coherence gives Dash the ease-of-use of shared-memory architectures while maintaining the scalability of a message-passing machine.

Published by the IEEE Computer Society

Circulation: *Computer* (ISSN 0018-9162) is published monthly by the IEEE Computer Society, 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA 90720-1264; phone (714) 821-8380. IEEE Computer Society Headquarters, 1730 Massachusetts Ave. NW, Washington, DC 20036-1903; IEEE Headquarters, 345 East 47th St., New York, NY 10017. Annual subscription included in society member dues. Nonmember subscription rates: available upon request. Single copy prices: members $10.00; nonmembers $20.00. This magazine is also available in microfiche form.

Postmaster: Send undelivered copies and address changes to *Computer*, IEEE Service Center, 445 Hoes Lane, Piscataway, NJ 08855. Second class postage is paid at New York, New York, and at additional mailing offices. Canadian GST #125634188.

BOOK REVIEWS

Editor: Alan Kaminsky, Rochester Institute of Technology, PO Box 9887, Rochester, NY 14623-0887
(716) 475-5255; Internet ark@cs.rit.edu;Bitnet arkics@ritvax.

Object-Oriented Design

Peter Coad and Edward Yourdon (Yourdon Press, Englewood Cliffs, N.J., 1991, ISBN 0-13-630070-7, 197 pp., $33)

This slim companion to the authors' previous collaboration, *Object-Oriented Analysis* (Yourdon Press, 1990), is a most welcome addition to the object-oriented literature. It will appeal to students, practitioners, and managers struggling to enter the world of object-oriented programming and to those already in it who want a treasure trove of useful techniques.

The heart of the book is a very simple and elegant insight about the architecture of object-oriented systems. The authors identify four components of an architectural model from their study of object-oriented designs, namely, the problem domain, human interaction, task management, and data management components. They address the design of each component type in separate chapters. Then they introduce the structured-design criteria of coupling and cohesion, and extend them — for the first time as far as I know — to object-oriented designs.

Obviously, coupling criteria can be applied to messages in the same way that they are traditionally applied to subroutines. Less obviously, the criteria can be extended to classes. Low coupling between objects implies that the number and kind of messages sent or received from other objects should also be kept simple. Even more subtly, the criteria can be applied to evaluate the inheritance structure. In this case, Coad and Yourdon suggest high, not low, coupling between classes as a design goal.

This three-step application of coupling criteria can also be applied to coherence criteria by looking at the coherence of each service in a class, each class as a whole, and the inheritance structure.

The book uses graphical class and object diagrams that are a seamless outgrowth of the diagrams presented in *Object-Oriented Analysis*. Object International Inc. offers automated support for drawing and checking the notation. In fact, the authors use the tool's design to illustrate the application of their ideas.

It may be impossible to gather all the useful techniques and design criteria for object-oriented systems into a single book, particularly when they are still being proposed and hotly debated; but some omissions here are worth pointing out. There are no detailed design guidelines. Such paradigms as the Smalltalk community's MVC model can help tie the design together by explicitly describing the constraints and relationships between, say, the problem-domain and human-interaction components. The book also omits the pitfalls of designing classes for reuse — for example, conflicting styles of validating arguments or handling errors.

The authors' recommendation of strong inheritance coupling seems to me to fly in the face of the so-called Law of Demeter, which argues for the need to hide implementation secrets even from a class's heirs. I feel the jury is still out on this issue, and I would have liked Coad and Yourdon to tackle it explicitly.

I found the structure of the bibliography flawed. It is divided into primary bibliography, secondary bibliography, reference publications, and related publications. Some of these sections also separate books from articles. Since all references in the text are cited in exactly the same way, I sometimes had to look in as many as five sections before finding a reference in a sixth.

None of these points diminish my strong recommendation of this thought-provoking gem of a book. Some readers may find the four basic design components suspiciously similar to current object-oriented technology, with its awkward gaps between the rich, commercially mature frameworks for graphical user interfaces and the poorer, more experimental handlers for tasks or persistent objects. But these components are also a boon to organizing quite different design concerns and techniques. I look forward to testing these wonderful new ideas on my next object-oriented design project.

Alejandro Teruel
Universidad Simon Bolivar
Caracas, Venezuela

Software Engineering with Abstractions

Valdis Berzins and Luqi (Addison-Wesley, Reading, Mass., ISBN 0-201-08004-4, 624 pp., $35)

In their preface, Berzins and Luqi note the difficulty of convincing new computer science students of the importance of systematic methods and formal specifications because of their experience in programming on a relatively small scale. Certainly, the widespread availability of personal computers and cheap compilers gives many new students the sense that they already know what software development is, sometimes even better than their professors. They think of it as programming or, even worse, coding.

The basic goal of this textbook for an advanced one-year course in software engineering is to demonstrate the use of modern formal methods. The authors emphasize approaches amenable to computer assistance. They concentrate on "a consistent treatment of the entire software development process and strive to present a single approach with sufficient depth to let the readers carry out the process, with concern on compatibility and integration, rather than a broad survey of popular techniques."

To support this approach, they use Spec, a formal specification language that they introduced in earlier works. "Spec can specify the behavior of three different types of software modules: functions, machines, and types. These modules can interact via three types of messages: normal messages, exceptions, and generators." Modules and messages form a set of primitives that can express all common software-component properties.

The semantic basis for Spec is the event model with its primitives of