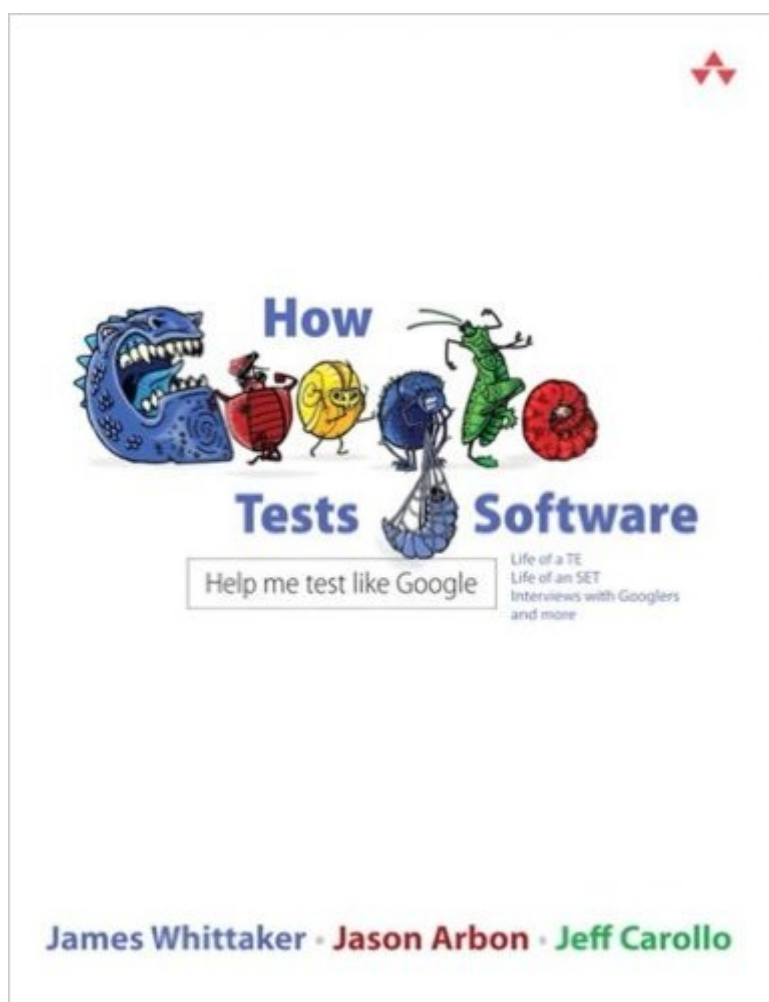


The book was found

How Google Tests Software



Synopsis

2012 Jolt Award finalist! **Pioneering the Future of Software Test** Do you need to get it right, too? Then, learn from Google. Legendary testing expert James Whittaker, until recently a Google testing leader, and two top Google experts reveal exactly how Google tests software, offering brand-new best practices you can use even if you're not quite Google's size yet! **Breakthrough Techniques You Can Actually Use** Discover 100% practical, amazingly scalable techniques for analyzing risk and planning tests thinking like real users implementing exploratory, black box, white box, and acceptance testing getting usable feedback tracking issues choosing and creating tools testing Docs & Mocks, interfaces, classes, modules, libraries, binaries, services, and infrastructure reviewing code and refactoring using test hooks, presubmit scripts, queues, continuous builds, and more. With these techniques, you can transform testing from a bottleneck into an accelerator and make your whole organization more productive!

Book Information

Paperback: 320 pages

Publisher: Addison-Wesley Professional; 1 edition (April 2, 2012)

Language: English

ISBN-10: 0321803027

ISBN-13: 978-0321803023

Product Dimensions: 7 x 0.8 x 9.2 inches

Shipping Weight: 1 pounds (View shipping rates and policies)

Average Customer Review: 4.2 out of 5 stars See all reviews (51 customer reviews)

Best Sellers Rank: #66,444 in Books (See Top 100 in Books) #12 in Books > Computers & Technology > Networking & Cloud Computing > Cloud Computing #31 in Books > Computers & Technology > Programming > Software Design, Testing & Engineering > Testing #82 in Books > Textbooks > Computer Science > Software Design & Engineering

Customer Reviews

When I found out about the book "How Google Tests Software", it didn't take long until I had ordered a copy. I find it quite fascinating to read about how Google does things, whether it is about their development process, their infrastructure, their hiring process, or, in this case, how they test their software. I am a developer at heart, but I have worked for a few years as a tester, so testing is also dear to me. It's quite an interesting book, and it makes some great points about the future of

testing. However, despite the phrase "Help me test like Google" on the cover, it is not as useful as I had hoped when it comes to improving your own testing. The book starts off by describing the key roles at Google: SWE (Software Engineer), SET (Software Engineer in Test) and TE (Test Engineer). Briefly, the SWE builds features for Google's products, the SET develops testing infrastructure and larger-scale automatic tests, and the TE tests the products from a user's perspective. After the introductory chapter, there is a chapter each on the SET and TE roles, and there is also a chapter on the TEM (Test Engineer Manager) role. The final chapter is about the future of testing at Google (and in general). Software Engineer in Test (SET) As the different roles are explained in the respective chapters, there is also quite a bit of detail on how the testing is done at Google. The most interesting part in the chapter on the SET role is the part about the infrastructure. There is (of course) extensive support for running tests automatically. There is common infrastructure for compilation, execution, analysis, storage and results reporting of tests. Tests are categorized as small, medium, large or enormous.

I saw James Whittaker speak at STAR West in 2011, and he gave a keynote titled "Test Is Dead". His talk was essentially a teaser for How Google Tests Software that he co-wrote with Jason Arbon and Jeff Carollo. The premise of the book is that testers need to have engineering skills (sometimes to an equal extent as software engineers) in order for the testing discipline to reach first class citizenship on equal footing with development. The argument aligns well with the movement toward agile software development methods. The book goes on to breakdown testing responsibilities for software engineers (SWEs), software engineers in a test role (SETs), and Test Engineers (TEs). Almost half of the book deals with the roles and responsibilities of the TE, and in the Google model, they do have a higher-level role in testing. In essence, it breaks down like this:*

- SWEs write unit tests for the software they write*
- SETs write tools to enable testing without external dependencies and write automated functional tests*
- TEs coordinate the overall testing activities for a product and focus on the user by doing exploratory testing

In addition, the book also outlines a number of tools (many of which have been open-sourced) that Google uses for testing in the context of these roles. The majority of the content focuses on web applications (it's Google after all), and some of the ideas won't apply if the majority of your development is for internal customers to your company - since you probably have user training and rules about frequency of release. However, I would say that you could apply 80% of the ideas in any context and probably adapt at least 10% (if not more) of the others to your situation.

The main contribution of this book, besides being an excellent read for anyone who considers working at Google, is the proclamation of how seriously software quality should be taken. Paradoxically, the book is technically complex, and yet those who should really read this book are managers - who often have a factory view of software development and fail to understand that high quality costs less. This is such an important lesson that needs to be learned by the software industry, that the effort of any author to demonstrate this point must be lauded. Unfortunately, the book has two main drawbacks: one is that it is so specific, that it is unlikely to be of much help to other companies. The testing framework Google has built is extraordinary, but it is not a framework that can be easily reused in other contexts: it is highly web-oriented, and it leverages Google's distributed infrastructure. The other is that the book is highly romanticized. It almost reads like a romance, and SETs are the heroes. On one page, it is described how a developer can launch hundreds of tests and get coverage reports with a one-line command, a hallmark of efficiency - but on another page, a code sample using the testing framework is presented and it consists of 90% boilerplate code. The book is riddled with confrontations between the idealistic reality the authors describe and how that vision falls short of reality, be it in code samples or interviews with Googlers. Also, SETs are presented as superhumans - in the section where the hiring requirements for a SET are listed, one learns that in order to be a SET at Google, one needs to be a genius. Not a Google-employee level of genius, but an Einstein-who-can-also-read-other-people's-minds level of genius.

[Download to continue reading...](#)

Tame Your Gmail in 5 Easy Steps with David Allen's GTD: 5-Steps to Organize Your Mail, Improve Productivity and Get Things Done Using Gmail, Google Drive, Google Tasks and Google Calendar
How Google Tests Software McGraw-Hill's 500 Physical Chemistry Questions: Ace Your College Exams: 3 Reading Tests + 3 Writing Tests + 3 Mathematics Tests (McGraw-Hill's 500 Questions)
Google AdWords: Earn Better Revenue through Google AdWords Google Drive & Docs in 30 Minutes (2nd Edition): The unofficial guide to the new Google Drive, Docs, Sheets & Slides Google AdSense & SEO Secret \$100/ Day: How I make \$100/ day with Google and my SEO secrets Seo: 2016: Search Engine Optimization, Internet Marketing Strategies & Content Marketing (Google Adwords, Google Analytics, Wordpress, E-Mail ... Marketing, E-Commerce, Inbound Marketing)
SEO: 2016: Search Engine Optimization, Internet Marketing Strategies & Content Marketing (Google Adwords, Google Analytics, Wordpress, E-Mail Marketing, ... Marketing, E-Commerce, Inbound Marketing) SEO for Google Places - The Secret to Crushing Your Competition with Local SEO and Google Places Software Engineering Classics: Software Project Survival Guide/

Debugging the Development Process/ Dynamics of Software Development (Programming/General)
Surreptitious Software: Obfuscation, Watermarking, and Tamperproofing for Software Protection:
Obfuscation, Watermarking, and Tamperproofing for Software Protection Shipping Greatness:
Practical lessons on building and launching outstanding software, learned on the job at Google and
Military Flight Aptitude Tests, 5/e (Peterson's Master the Military Flight Aptitude Tests) McGraw-Hill
Education 5 TEAS Practice Tests, 2nd Edition (Mcgraw Hill's 5 Teas Practice Tests) The Official
SAT Subject Tests in U.S. & World History Study Guide (Official Sat Subject Tests in U.S. History
and World History) Tests & Measurement for People Who (Think They) Hate Tests & Measurement
Scholastic Success With Reading Tests, Grade 6 (Scholastic Success with Workbooks: Tests
Reading) Scholastic Success With Reading Tests, Grade 3 (Scholastic Success with Workbooks:
Tests Reading) Master The Officer Candidate Tests: Targeted Test Prep to Jump-Start Your Career
(Peterson's Master the Officer Candidate Tests) NES Elementary Education Practice Questions:
NES Practice Tests & Review for the National Evaluation Series Tests

[Dmca](#)