A few years ago I co-wrote the book *How We Test Software at Microsoft*, along with Alan Page and BJ Rollison. As the author with the most experience with online services it fell to me to write the chapter on “Software Plus Services.” The book was published in 2009 – just a few years ago – and it may be surprising that many of the techniques in that chapter are already a bit dated. The root of the challenge was that cloud computing wasn’t strongly on my radar at the time.

We had an internal Microsoft memo on cloud computing and a project code named RedDog (now Windows Azure) that leaked to the press as early as 2008, but at the time the cloud wasn’t real for us testers. As test engineers, we always dealt with what needed to be shipped in the next few weeks or months. The techniques we used were the tried and true ones we had at hand to simply get the job done and release the product with good quality. At that time, the approach we used for software testing was very hands on and very lab centric, where you could deploy your service to a machine under your desk and test away or maybe even push the bits down the hall to a test lab. The thought that the service would run as a virtual machine in the cloud or that the storage would be a separate cloud service wasn’t even part of the equation.

In this new book, *Software Testing in the Cloud: Perspectives on an Emerging Discipline*, Scott Tilley and Tauhida Parveen have gathered together some of the top researchers in software testing with an eye to answer the question, “What changes when you test in the cloud?” If you are new to cloud, don’t worry; the book’s chapters cover all the core concepts, from defining what is meant by Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) to many other basics such as virtualization and cloud storage.

Readers will learn from practical examples that are well researched and documented for techniques such as leveraging cloud-based testing services or Testing as a Service (TaaS) along with performance and scale testing in the cloud and even how to improve the testability of your cloud-based application. Some chapters tap into the rise of Big Data and the use of techniques such as MapReduce to analyze and produce test data as well as dealing with the rapid growth of mobile device applications. Another chapter, one of my favorites from a conceptual angle, is about building effective and efficient software testing factories. I love this idea because it embraces agile development and mixes in concepts of organizational and architectural isolation with an eye toward speed of delivery.

In just the past few years I’ve noticed that organizations that move to the cloud either adopt agile development or they build on top of and extend their current use of these development practices. With agile development and the cloud we are seeing projects that used to run for months and years before releasing anything cut down, modularized, and released over weeks up to months. Add to that the shift toward speed as a massively important competitive advantage both for software companies and other
industries that are tapping into technology for a competitive edge and we can now see why the factory metaphor is useful for framing and managing process changes for organizations that are moving to the cloud.

As the wheels of software innovation speed up, we have to ask ourselves, how do we optimize for throughput instead of simply output? How do we maximize value shipped to the customer, at the right quality, that truly delights them? Of course one key answer is the cloud, but the other more germane answer is optimizing our use of the cloud both as a release platform for services and as a tool to help us test those services more effectively.

The last point I want to make is that *Software Testing in the Cloud: Perspectives on an Emerging Discipline* is meant to be a reference and as such you’ll want to rough it up a bit. Tom Peters, author of *In Search of Excellence* and several other great business books, wrote that he loved it when folks would give them their personal copy of one of his books to autograph. He’d flip through the first few pages of the book and note all the dog-eared pages, the yellow highlights and the notes in the margins. He commented though that he would keep flipping all the way to the end of the book and he noticed that the number of underlines and bent page corners would often drop off to nil. To be honest I have a number of books I use as references where my diligent reading and note taking dropped off long before the final chapter.

Each chapter in *Software Testing in the Cloud: Perspectives on an Emerging Discipline* is a great standalone case study but as a collection they become an invaluable reference for anyone goal on ensuring quality of cloud-based solutions. Enjoy the book, mark it up, flag key concepts for quick recall and then when you’ve beaten your copy into a personal reference put it on the shelf about hand high; then, when that young engineer is in your office and not quite understanding your point you can reach over, grab the book, and flip right to that critical illustration.

Please enjoy.

*Ken Johnston*

*Microsoft Corp., USA*

---

*Ken Johnston* is a frequent presenter, blogger, and author on software testing and services. Currently, he is the Principal Group Program Manager for the Bing Big Data Quality and Measurements team at Microsoft Corp. Since joining Microsoft in 1998, Ken has filled many other roles, including test lead on Site Server and MCIS, and test manager on Hosted Exchange, Knowledge Worker Services, Net Docs, MSN, Microsoft Billing and Subscription Platform service, and Bing Infrastructure and Domains. He has also been the Group Manager of the Office Internet Platforms and Operations team (IPO). For two and a half years (2004-2006) he served as the Microsoft Director of Test Excellence. Ken earned his MBA from the University of Washington in 2003. His is a co-author of the book *How We Test Software at Microsoft* and contributing author to *Experiences of Test Automation: Case Studies of Software Test Automation*. 