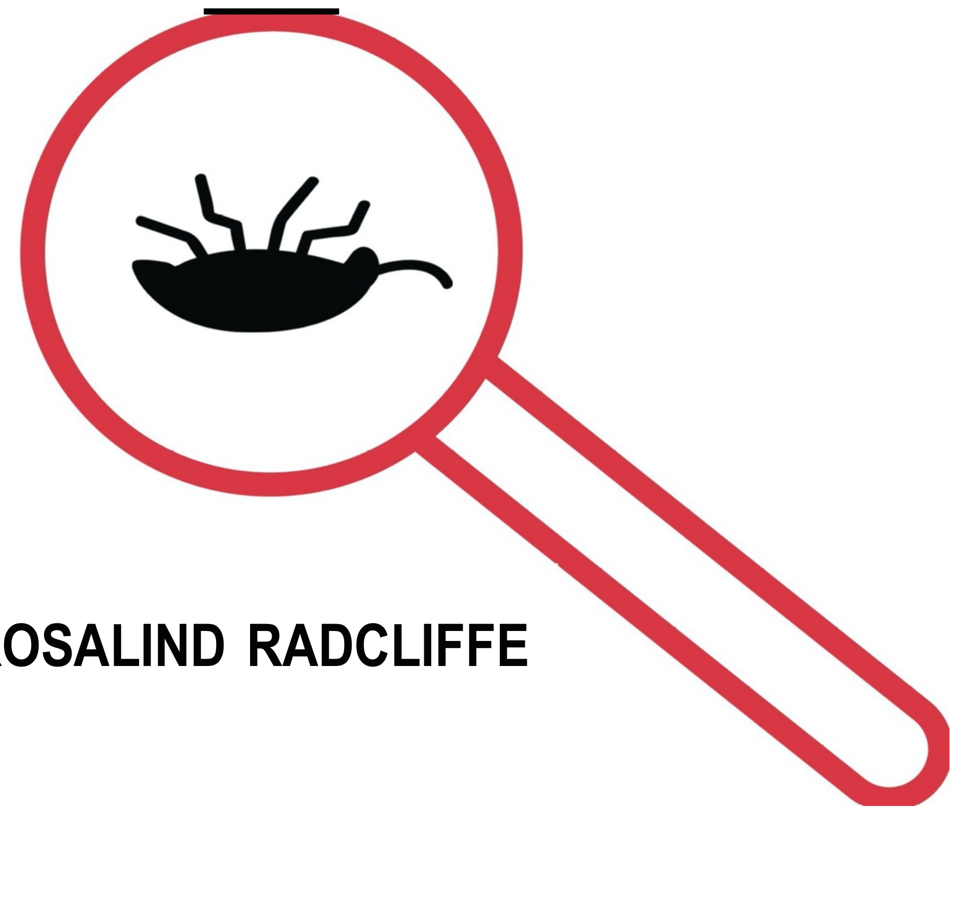
Enterprise Bug Busting

FROM TESTING THROUGH CI/CD TO DELIVER BUSINESS RESULTS



ROSALIND RADCLIFFE

Enterprise Bug Busting
From Testing through CI/CD to Deliver Business Results
©2021, Rosalind Radcliffe

Published by Accelerated Strategies Press
acceleratedstrategiespress.com

ISBN: 978-1-09838-149-3
ISBN eBook: 978-1-09838-150-9

"There is no longer any question whether modern DevOps practices can be used to accelerate and improve quality for legacy systems. Rosalind describes DevOps testing concepts in language that makes sense for large enterprise systems and describes large system challenges in a way that makes sense for DevOps testing. If you were looking for a primer of what you need to know and understand before you apply continuous testing in a complex system, you found it. There are two modes for using this book: 1) With a highlighter in hand to mark the ideas you want to apply as you bring enterprise DevOps testing to life; and 2) Hand this book to a peer or leader in your organization and tell them to read it so that you can talk about what to do next. I recommend both modes."

Mark Moncelle
Enterprise DevOps Practitioner

"Enterprise Bug Busting hits home on the all important topic of testing in the enterprise. It defines a roadmap for implementing a continuous integration practice in your organization, and Rosalind Radcliffe, drawing on a lifetime of DevOps experience, demonstrates how application development leaders can improve software quality whilebuilding hybrid cloud applications that are delivered continuously."

Sherri Hanna
Senior Offering Manager, DevOps for IBM Z Hybrid Cloud

"Mainframes are still a critical part of IT infrastructures, and Rosalind Radcliffe is an expert in bringing modern development technologies to trans- form these systems beyond their traditional role as systems of record."

Charlene O’Hanlon
Chief Content Officer, MediaOps

“In my years in the DevOps community, Rosalind Radcliffe has been a shining, true North Star pointing the way for enterprises to adopt the latest technologies, while leveraging their existing investments to do more, better and faster. This book is a continuation of this common sense, best practices approach to success.”

Alan Shimel
CEO and Founder, MediaOps

“Software quality is an immutable theme throughout Rosalind’s remarkable career. She artfully imparts her masters-level knowledge. Enterprise software teams will immediately benefit from this book and alter course towards improved software quality in this age of digital transformation.”

Mitch Ashley
CEO and Managing Analyst, Accelerated Strategies Group

“This book showcases Rosalind’s unique ability at providing common sense transformational guidance to enterprises with mainframe and complex environments.”

Jayne Groll
CEO, DevOps Institute

To my husband Bob who has always been there to support me, and for my children so they can understand more of what I do.

FOREWORD

Do you work in a large organization that has a variety of applications including mainframe systems running your core business? Do you struggle to deliver improvements with the speed the business requires and the quality the customer expects? When working with themainframe teams, are you confused as to why everything tends to work differently than the rest of the applications? If so, then this book is for you, and Rosalind is the best guide to help you navigate all of these challenges and understand how to create quality systems for complex environments.

I met Rosalind when she presented at the 2015 DevOps Enterprise Summit. It was in the early days at the conference when most of what was being presented was about improvements on small applications that were not core to the business. Everyone was recommending small- isolated teams working with front-end code. When Rosalind steppedonto the stage to talk about mainframes I thought, “Well, this is going to be interesting.”

She started clearly describing the differences between the stability provided by the redundancy of distributed systems vs. the stability provided by mainframes that support the core business of most large enterprises. Rosalind stepped through each and every DevOps practice being discussed at the conference and showed how each and every one of the practices can be implemented on mainframes. Rosalind closed by stating that there isn’t anything fundamentally wrong with mainframes, and in fact they run most of the core businesses in the industry. We just shouldn’t be developing on them the same way we did 30 years ago. It was one of the best presentations I have ever seen, and I decided I had to meet her.

As I have gotten to know her better over the years, I have learned she has the unique ability and skill to speak strategically with executives and technically with senior engineers. She can go into an organization and help the executives understand the changes required. When the senior technical leads that have been doing the same thing the same way for 30 years say it won’t work, she can dive into the technical details to address their concerns and help them understand how it actually can and should work. Rosalind is one of the most impressive people I have met in the industry.

She has a passion for quality and 30 years of experience driving improvements both inside IBM and with its customers. Through this experience she has realized quality is not something that can be delegated to the quality assurance organization. It needs to be everyone’sjob. It is not just a function of testing. It involves everything from the idea for a new capability to ensuring it meets the needs of the business and the expectations of the customer. An effective quality system involves all the steps in between.

Rosalind starts with this broader view of quality and shows how to provide an end-to-end system. She demonstrates her knowledge of how to apply these concepts to a broad range of applications. More importantly she shows how to apply these concepts to mainframes,which she is uniquely qualified to do. She helps people understand how mainframes are different from the other applications.

She deep dives on how to create effective quality systems for the mainframe with customer examples and key learnings. Rosalind also covers the latest capabilities of the IBM Z that can support these more modern development approaches that are valuable but not broadly used. She shows how and why organizations that haven't modified their development approaches as much as they should have over the last 30 years should be leveraging these new capabilities.

If you work in a large enterprise with lots of different applications, including the mainframe, this is a must read. It will help you understand how to create a truly effective quality system and explain how and why the mainframe is different. It will also help you understand the more modern approaches to development that your mainframe teams need to start adopting.

Gary Gruver
President of Gruver Consulting

TABLE OF CONTENTS

WHO SHOULD READ THIS BOOK

This book is targeted at large enterprise organizations that have systems including IBM Z (the mainframe as many may consider it), or anyone interested in learning more about software quality including IBM Z. This book is targeted to executives who are in an organization with these large complex systems, as well as engineers working to improve the overall process. This book does not assume any specific background knowledge, and therefore, includes definitions and terms so those with multiple/various backgrounds can gain an understanding of software quality for the large enterprise.

INTRODUCTION

No matter where you look today, you will find technology everywhere. This includes software, which plays a significant part in our lives in some way every day. I've witnessed this evolution to software everywhere over the course of my lifetime, going from a day when the only software directly affecting my life was in the telephone system, though not in the telephone attached to the wall. A time when cars were mechanical machines without software guiding, helping and entertaining, and when watches were about fine Swiss movement. Today, we can't get away from software. Software is in hospitals, cars, itcontrols the electric grid, it's used daily in our phones and our homes. Any financial transaction you make involves software as well, even if cash is used, the processing in the cash register is all software.

With software everywhere, the quality of that software and supporting hardware can affect our lives in many varying ways. If an internet search renders a 'page not found' message it's a minor inconvenience. But when software fails in a car or in an airplane, it's life threatening. The combination of software and hardware controls so many facets of our lives in ways many people don't even recognize. The importance of this must be recognized and acknowledged by those of us in the IT industry.

As noted, there are instances where the quality of software can determine life and death. While most of the time it's not that critical, it isimportant to understand that

software in any form does impact our lives. This understanding is why it is important to improve the overall quality of the systems being built.

The goal of this book is to provide insights that will help enterprises improve the overall quality of software being created today. This overall quality is achieved throughout the entire development lifecycle based on the inclusion or omission of activities, automation and organizational strategy. I will provide definitions for areas that drive software quality, key insights from years working with various organizations, and stories to explain what others have done. Some stories will be positive and some not so positive. Sharing both the good and the bad is intended to help others learn from and steer clear of the mistakes made by those before them. We all learn through experimentation, I hope to help readers avoid some trouble, and offer insights into new options to improve the overall process. I will share stories from my 30+ years of experience working with various companies, across the globe, large, small and everywhere in-between, as well as my work within IBM.

One important note is that this book is primarily focused on the existing enterprises with existing applications and code bases. Organizations that have been around for a while vs. the general startup organization who's entire code base may be less than a few years old. Though there are lessons for everyone, the examples provided and discussions will be based on large long-term enterprises. Startup environments will have a different view as they will generally have started more with modern DevOps practices, or cloud native development practices. Common attributes of large enterprises are:

- Widely diverse languages, coding styles and application architectures.
- Testing and deployment practices are often well established and

require a culture change.

- Up or downstream systems that “just work” and are seen to be too expensive or risky to change.
- Complexity of the system is too large for any one team to understand, develop, or manage. This requires multiple layers, which adds additional complexity when testing the entire system.
- The scale, complexity and reliability requirements, drive the requirements that automation has to be designed from the start for production standards.
- Return on investment needs to be considered in all aspects of change, change does not happen just because something new comes along, but it has to provide business value. An example would be the focus on automated testing for a part of the solution that does not change often.

Why is it so important to consider the overall software quality of large complex systems? Large complex systems including IBM Z, host more than 70% of the total structured data. IBM Z is used by most of the world’s top banks, insurance companies and retailers, it is heavily usedin travel and transportation, not to mention the various governmentsystems. These systems provide the business value to many organizations, yet these systems are seen as legacy, hard to maintain andhard to work with. However, they are at the core of the business process,by bringing the full complex system including the IBM Z into the overall software quality process, using automation for the repeatable tasks, and bringing modern processes to the development and operations environment, not only does the quality improve but the business value can be unlocked for greater use by other parts of the organization as part of the digital transformation.

When I started at IBM, I was just out of college and my first job was working on the then current version of IBM Z, which was the IBM S/370. My original job was to simply learn the language used for the system, and learn how the system was developed and tested. First assignments were focused on testing, or small coding changes to getused to how the system worked. We did have automation, and I learned early on, to automate tasks that had to be repeated. The parts I workedon were critical to the system, so I had to be very careful with changes tomake sure nothing that used to work was broken. This took time and conscious effort. To test the various terminal types, I had to go to a terminal room and test the functions on each of the various terminals available. When we got our first PCs, the team had the ability to use the terminal emulator to test all the various options instead of having to spend time in the terminal room, this helped speed up the process and allow for more automation to be used for the testing.

These early days taught me a lot about the importance of quality. One other early opportunity I had was to work with clients directly through our major User Groups. This early feedback directly from the client regarding how they actually used the product, what they liked and what they did not like, helped drive changes for improved user experience. I also spent a number of years in IBMs User Centered Design organization, the precursor to what we have now, Design Thinking, witha focus on users. Human factors testing allowed us to see how actual users would use the capabilities. With this information we could better design the functions. This focus on what is best for the end-user led to the work we did with Common User Access (CUA), which helpeddefine the industry standard user interface. This work also led to the IEEE standard for user interface.

Anyone using a computer today still sees the impact of that work, the menu bar across the top, with File, Edit, View and Help, in standard locations, and the standard cut, copy and paste keyboard shortcuts – ctrl-c, ctrl-v, ctrl-x, ctrl-z, (or cmd on Mac).

Having worked to help drive this standard across all of the major IT vendors at the time was a challenge, but the fact that it lives 30 years later shows the value of this effort to focus on the end user. This was obviously a team effort, but I was glad to be part of it so early in my career.

Throughout my career I have worked in various roles, but this beginning drive for quality for our end users has stuck with me. Over the 30 years since, development practices have gone through many changes and evolutions, however, when looking at many IBM Z shops thedevelopment and operations practices have not evolved in the same way.Even today when I talk with some Z developers, I see their development practices and tools, look exactly the same as when I started. Those 30 years of evolution that helped drive the change for digital transformation, for software everywhere, seems to have left out this very critical part of the organization. The IBM Z system itself has not stood still, it has continued to evolve as all other systems have. Now is the time for the people working with IBM Z software, to also take advantage of all the new possibilities available.

This book leverages those changes in the system and takes those experiences and brings them together to help organizations around the world learn from the lessons of others.

Development practices have evolved over the years, but fundamentally it's always the same. Software development requires developers to use their knowledge and skill to create a set of capabilities. These capabilities must be verified to ensure they do what is intended and they must interact with the rest of the system in the way intended. This is a very simple view of the process, but fundamentally, at its core that's what software development is.

Software development is, at its core, the innovation of individuals creating new functions based on a set of user requests. It is not the same as a manufacturing process where you work to remove variability to help ensure every object comes out the same with the same level of quality. Creating a quality system requires a combination of activities, automation and measurements to ensure that creation satisfies the nonfunctional, as well as, functional requirements within the right risk profile. The acceptable level of risk will vary based on the type of software being developed, the ease of deploying a fix and the impact ofa problem.

The reason to create the software is to provide business value. Building software is done by a team of responsible people, and these teams do not work in isolation from each other. These teams include the customers, internal or external, and representatives of the external users. The requirements for the software form a hypothesis for what can provide that value. That then needs to be verified by the user, once created, to allow for the adjustment as necessary. The value is useless without quality. Nobody wants a product that does not function to meet their needs.

Since the actual creation of the software cannot be tightly controlled, the activities related to the creation should be as automated as possible to remove the opportunity for error from all the surrounding activities. Simply put, the creative activity of writing the code needs all the flexibility to allow for innovation. The process of building, and deploying the software should be completely automated as there is no need for creativity in this process, but there is a high requirement that it be highly repeatable.

What is built as part of this creative process is based on a set of requirements, based on market, business and user input. These requirements will also include a set of nonfunctional requirements, or at least nonfunctional expectations for the function. This includes characteristics such as availability, response time and security, among others. The requirements driving the software creation, the software creation itself, and all the processes related to it, are often referred to as the software development lifecycle (SDLC).

The creative process of designing software requires methods to test that software to ensure it satisfies the requirements as specified. This testing process is what many people include in a quality assurance program, but in reality the software quality is determined by the entire SDLC.

Over the years various transitions have happened with the SDLC, generally to add additional steps in the process, approvals, and additional checks to help improve the perceived quality of the solution.

However, as I will describe in the book, some of the changes added, alleviated a problem but brought more significant problems into the process. We, therefore, need to address the entire lifecycle in context of the quality we are working toward.

Another important note in driving software quality is that since it is a creative process, you can't guarantee there are no problems. The goal should be to deliver the appropriate level of risk for a problem. Determining the level of risk acceptable to each solution, anddetermining appropriate ways to measure that risk are key factors.

Measurements are an important part of driving software quality, but determining the right metrics to drive the right behaviors is the challenge. Providing the right focus at the right time in the process to drive the highest quality possible with the least effort.

I will detail throughout the remaining sections of the book how each aspect of the software development lifecycle contributes in its own way to the final solution running in production. I will discuss the implications for measurements and how they drive behaviors in ways that may not lead to improved quality.

In many ways this book is written for executives and engineers alike who work in organizations that have IBM Z, but are not familiar with it, this book provides the explanations for why things have come to be done a certain way, as well as suggestions for how to help change those practices. However, for those of you working with IBM Z, you can skip those definitions of terms you already know, but the suggestions for new ways of working apply to you as well.

I have designed this book so that it can be read front to back, however, if only a particular topic is of interest, each section can stand alone. Each section will contain the definition and foundation information, as well as key insights and stories from various customers environments. In addition, background information is included for those less familiar with particular areas such as the IBM Z environment.

- **Section 1** provides general background information, definitions for key areas such as software quality, metrics, continuous integration and continuous delivery (CI/CD) and background information. It will include a description of the types of customers that I will use as examples in the book as well as abstract descriptions of those customers. In addition it provides high-level overview of IBM Z and definitions of key termsrelated to IBM Z and an introduction to traditional z/OS development processes. The goal of this section is to lay the foundation for an understanding of IBM Z as well as areas related to software quality.
- **Section 2** delves into essential components related to the process of software quality including the overall pipeline, environments for use during the process, fundamentals related to data for testing and overall high-level practices that should be followed. This section describes how modern development practices can now be used including IBM Z, and describes the importance of each aspect as it relates to the process.
- **Section 3** provides a view of the various types of testing, the users who perform the tests, and additional considerations for each type of testing. It describes how the IBM Z system can be

included and additional considerations for inclusion in each type of testing.

- **Section 4** offers a summary, options for roadmaps for transformation and additional stories to bring the entire process together.
