Conquering the Testing Challenges of Serverless Applications

Presented by:

Peter Varhol & Gerie Owen

Technology Strategy Research

Brought to you by:

888-268-8770 - 904-278-0524 - info@techwell.com - http://www.starwest.techwell.com/
Peter Varhol

Peter Varhol is a software strategist and evangelist who closely observes the testing industry and uses his knowledge and experience to identify new technologies and help companies respond to those trends. His diverse technical background enables him to seamlessly integrate new technologies into his practices and provides him with a unique vision of how to adapt and succeed. His areas of research and practice include team communication, machine learning, DevOps, testing and test automation, and development tools. His efforts aim to use the latest and best technologies to address real-world problems. Peter speaks frequently at conferences, local meetups, and webinars on software development, testing, machine learning, and DevOps topics. Peter blogs at Cutting Edge Computing, and can be found on Twitter. He has master’s degrees in computer science, applied mathematics, and psychology, along with doctoral work in information systems.

Gerie Owen

Gerie Owen is Testing Strategist and Evangelist at Technology Strategy Research. She is a Certified Scrum Master, Conference Presenter and Author on technology and testing topics. She enjoys mentoring new QA Leads and brings a cohesive team approach to testing. Gerie is the author of many articles on technology including Agile and DevOps topics. She recently developed a curriculum for DevOps 101 training. Gerie chooses her presentation topics based on her experiences in technology, what she has learned from them and what she would like to do to improve them. Gerie can be reached through her website, her blog, Testing in the Trenches, @GerieOwen on Twitter, and on LinkedIn.
Testing Serverless Applications

Peter Varhol and Gerie Owen
About me

International speaker and writer
Graduate degrees in Math, CS, Psychology
Technology communicator
Former university professor, tech journalist
Cat owner and distance runner
peter@petervarhol.com
Gerie Owen

• QA Evangelist, test manager
• Subject matter expert on testing for TechTarget’s SearchSoftwareQuality.com
• International and domestic conference presenter
• Marathon runner & running coach

gowen@qualitestgroup.com
Agenda

What the heck is serverless?
Why serverless means DevOps
What do we test?
Shift left and shift right
Testing strategies for a new architecture
Summary
What the Heck is Serverless?

A cloud-computing execution model
You supply the code
  • Generally back-end processing
  • Prompted by an event from the UI or other function
The runtime manages resources
Often event-driven
  • It doesn’t have to be a full application
What the Heck is Serverless?

“Serverless” is a misnomer
• You actually do use servers
• You just don’t provision servers and manage infrastructure

Other names are more appropriate
• Function as a Service (FaaS) is one name
• Event-driven computing is better

What do we mean by runtime?
• JVM and .NET CLR are runtimes
• But serverless runtimes do much less
About Those Runtimes

Most serverless vendors offer compute runtimes
  • Execute application logic but do not store data

Examples of runtimes
  • Amazon Lambda
  • IBM OpenWhisk
  • Azure Functions

Runtimes only execute when an event occurs
  • Only charged when the runtime is executing
Language Support Depends on Runtime

AWS Lambda

• AWS Lambda natively supports Java, Go, PowerShell, Node.js, C#, Python, and Ruby code
• Provides a Runtime API which allows you to use any additional programming languages to author your functions.

Azure Functions

• C#, JavaScript, F#, Java supported
• Bash, PowerShell, Python, PHP experimental
Lambda Architecture
Why Serverless Means DevOps

Cloud-only
- On-prem may be possible depending on the runtime
- But it may not make technical sense

Containerization and workflow
- Functions and runtime can be encapsulated into Docker containers

Microservices
- Sizing functions can be a challenge
- Cost versus service definition
What is Different?

You only pay when your code is executing
• There is an economic model with serverless

Serverless is rarely a complete application
• Back-end only
• No UI, no database

Serverless is stateless
• No state to reset between tests
• Test data confined to sample transactions
• Workflow means all data is elsewhere
Runtimes Can Execute at the Edge

Limited-purpose runtimes can execute small amounts of code on IoT devices

- Real time computing may be possible
- Data collected at the edge
- Then sent to a central store

Good for distributed data systems (DDS)
Testing Strategies for Serverless Applications

Understand the relationship between the code and the runtime

More on the runtime

Shift left

Integration testing

Continuous testing

Performance and load testing

Security testing
Understand the Relationship Between the Code and the Runtime

The cloud vendor provisions the servers and provides the runtime
  • The OS doesn’t matter

The cloud vendor provides security up to the runtime
  • The code is all yours

The runtime kicks off your code
  • And does nothing else
Runtime Implications to Testing

Only code needs to be tested
  • No interactions with runtime
  • Everything else is standard

No calls into runtime or OS

Does the function work with other functions?

And database, UI
Shift Left

Planning and design
  • Understand the application architecture
  • Make suggestions on testability

Unit testing
  • Depending on the function, it may also be a single unit

Map out integration points
Integration Testing

All functions intended to execute under serverless

All services used by the serverless functions
  • Database, UI, Web services

Test API operation
  • Send and receive data through function APIs
    • Compare inputs and outputs for accuracy

Full system test with simulated workflows
Invoke Runtime

Locally
- Prepare sample events
- Invoke the runtime with the events
- Observe outputs

Invoking the runtime
- Run the function inside a custom wrapper.
- Invoke functions locally using tools
  - Serverless framework or AWS SAM local
- Use docker-lambda to simulate an AWS Lambda environment
- Use local-stack to simulate AWS services locally.
invoke Runtime

Local invocation doesn’t replicate the production environment

In the cloud

• upload function
• Prepare sample events
• Invoke the runtime with the events
• Observe outputs
Continuous Testing and Serverless

An approach to managing risk by focusing on not only on improving testing efficiency but more importantly, increasing the effectiveness of your test processes. Assessing and mitigating business risk is the primary goal.

A focus on Quality at every stage of the Continuous Delivery process.
Continuous Testing vs. Test Automation

Continuous testing requires automation, however, it encompasses much more.

**Automated Testing**

1. Write test script
2. Run integration test
3. Wait
4. Write acceptance test script
5. Run acceptance test
6. Wait
7. Write performance test script
8. Run performance test

*Wait time impacts efficiency and delays feedback*

**Continuous Testing**

1. Write all test scripts
2. Integrate code
3. Build
4. Automate integration test
5. Automate acceptance test
6. Automate performance test

*No human intervention*
Why Serverless Needs Continuous Testing

Orchestration between serverless components can be complex
  • Are they calling each other?

Interaction between components and external services can change
  • Different locations
  • Expanded calls

Cloud deployment encourages continuous testing
Performance and Load

Each function requires separate testing
  • Average time to execute
  • Ability to execute under increased load

System performance and load testing

Cloud vendors offer performance and load tools

For local testing, use JMeter or commercial product
Security

Each function is an entry point to the application
• And must be tested for security

But the cloud vendor provides security through the runtime

Your application likely consists of multiple serverless components
• Making security testing a complex endeavor
Shift Right

Monitoring
  • Application performance and availability

Synthetic tests
  • Workflow/features/performance

Health analytics
  • Trend analysis
Testing is Easier . . .

No testing on different OS configurations or browsers
Most serverless functions are relatively simple
  • Usually a single feature of back-end processing
Really only testing APIs and workflow
And Harder . . .

It depends on the application architecture
Programmers aren’t familiar with event-driven systems
Many components make interactions difficult to test
  • This is MVC, but on steroids
Summary

Serverless brings a wrinkle to cloud computing
Testing needs to adapt to serverless realities
  • Testing occurs throughout the lifecycle
  • Testing includes monitoring in production
  • Testing is just the code

Our skills still apply, as long as we know the differences
Questions?

Thank you!