Research Project Exhibition 2022

M.Sc in Software Solutions Architecture

M.Sc in DevOps

FOREWORD

Today's Research Project Exhibition is a day marking great achievement by our Master's students in our DevOps and Software Architecture Programmes. January 2018 when the students now presenting their research projects started their Masters programme journeys seems like only yesterday, yet so much has happened since then. For the quality of work being presented by our Master's students we salute you. We know how much has been learned over the two years just as we know how much inspiring talent and industry you have brought to your efforts.

The Projects being presented today come from those completing the M.Sc. in Computing with DevOps and the M.Sc. in Software and Solutions Architecture. As well as congratulating the students in the programme we should also congratulate your lecturers and project supervisors.

These programmes are being taught here in TU Dublin but they are collaborative programmes where we have joined with industry in defining and scoping these programmes. Industry in this case was personified in Tony Devlin, Programme Consultant for Technology Ireland Skillnet. Technology Ireland Skillnet and their ICT employer partners identified the market need for these programmes, TU Dublin, Computing responded, co-designing an industry list of requirements into an academic Masters programme.

For the Software Architecture programme we are proud to partner with the International Association of Software Architects (IASA) and their Irish partners the Irish Computer Society (ICS) in offering this programme. It should be noted that this programme is recognised by the IASA for membership of the association.

For today's presentations we are delighted to have eminent speakers Ronan Dalton, CTO IBM Ireland, Chris Johnson, Senior Solutions Architect, AWS, Dr. Paul Clarke DCU and Brian Loomis from Princeton Digital Advisors. We really appreciate your contribution to this event.

Today we have a new cohort starting their journey as well as a cohort moving to the final part of their journey. They have plenty to learn but they too will soon reach this point. Creating and passing on knowledge is the duty of a University and doing this in fast moving Technological Fields relevant to industry is the particular mission of a Technological University.

We hope you enjoy the project symposium.

Finbarr Feeney PhD

Head of Department of Computing

TU Dublin

MSc Software & Solutions Architecture
MSc Computing with DevOps

RESEARCH PROJECT SYMPOSIUM AGENDA

21st January 2022

15:00 - 15:15 Opening Addresses

Dr. Pramod Pathak, Dean of the Faculty of Digital and Data, TU Dublin

Dr. Barry Feeney, Head of the Department of Computing, TU Dublin, Tallaght Campus

Dr. John Burns, TU Dublin, Tallaght Campus - Master of Ceremonies

Gary Clynch, TU Dublin, Tallaght Campus

15:15 - 16:00 Keynote Address

Mark Greville, VP of Architecture at WorkHuman

16:00 - 17:45 Poster Presentations Track 1 – Cloud and Infrastructure

16:00 - 17:45 Poster Presentations Track 2 – Software and Data

Technology Ireland ICT Skillnet

TU DUBLIN
OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH
TECHNOLOGICAL UNIVERSITY DUBLIN

Online MSc in DevOps

With most technology organisations moving their delivery platforms to a DevOps approach the shortage of people with cross sectional skills in DevOps is now acute. Developed by industry as a direct response to this need this first-ever Master's degree in DevOps aims to fill these important talent gaps and give credit, recognition and credibility to technologists working in this field.

The advantages of Development Teams and Operations Teams collaborating to improve the delivery of technology solutions has meant a rapid adoption of DevOps approaches to the Software Development Lifecycle. Closely associated with Lean and Agile concepts in enhancing the delivery of technology solutions, the DevOps approach has impacted very rapidly on the Technology industry.

Most existing DevOps 'specialists' grow or develop into their role with no formal standards or certification, and a modicum of training in the actual practice of cross functional DevOps practices. They may already be experienced, highly skilled, competent and high performers in their own field of Software Development, Computing, IT Management, or Quality Assurance but they can lack the knowledge and understanding of the other cross functional disciplines they now find themselves working with daily. Understanding not only the technical, but also the business and human factors at play during the high pressure demands of modern software delivery processes, is essential in the modern discipline of DevOps.

Award Level

There are two phases to the award. Candidates are registered for the full Masters of Science in DevOps Level 9 degree (90 credits) however candidates may opt to exit the programme on successful completion of the first three semesters with 60 credits and receive a Level 9 Postgraduate Diploma in DevOps (60 credits). Please note exit awards are at the discretion of the college and no refund of fees will be due.
The award structure will place greater emphasis on continuous assessment, practical and project work rather than on formal examinations. In fact there are only 2 modules that carry an actual exam.
The aim is that participants will gain a deep understanding of the topics and content covered, and be able to demonstrate this acquired knowledge as proven competence in tests and exercises drawn from practical "real life" DevOps scenarios.

Programme Delivery

The programme will start with a 3 day workshop which will involve all participants being physically present. This is seen as important to facilitate networking, experience sharing and group learning.

It is expected that lectures will be delivered one evening per week in term time and every 3-4 weeks there may be a requirement to hold lectures twice in that week. There will also be a requirement to attend one on-campus day at the end of each semester.

Lectures will be streamed live from TU Dublin (Tallaght Campus) i and will be available for download and offline viewing.

Semester 1: Introduction to DevOps

Human and Organisational Issues	Software Development Methodologies
• Lean and Agile movements and methods • Assess and evaluate organisational design and culture to facilitate DevOps style development, deployment and support • Develop and manage global multi-disciplinary teams including an understanding of the cultural and practical issues which arise • Be able to form, lead and develop teams • Assess competence, accountability, responsibility, norms and operational management • Collaboration, negotiation and partnering • Managing the Future - Creating a readiness for organisational change, organisational development and change management	• Technical implications of DevOps – the philosophy, the history, the SDLC, Lean, Agile Manifesto, continuous feedback and learning • Change, Source, Defect Control Systems, Examination of major industry implementations (e.g. Atlassian, VSTS) • Code Promotion • Code Synchronization • System Debugging • Software QA • Automated Testing • Software Security Vulnerability Management • Software Telemetry and Monitoring • Feedback and Learning

Semester 2: DevOps Fundamentals

Business Technology Strategy	IT Infrastructure Fundamentals for DevOps
• The Business Case for Agility and DevOps • Lean/Agile management/methods/frameworks (SAFE) • Product road maps, pipelines, backlogs, valuing new features and technical debt • Business case development and risk assessment • Creation/management of multi-annual business plans • Financial Management of Product and Technology life-cycles • Project Management and Methodologies • The end of the monolithic project • Designing for agility and value • Challenges for DevOps • Regulated Software • Impact for Customers of DevOps approach	• Automation of Infrastructure • Task and Process automation languages • Advanced System Administration • Software Security • System Hardening • Policies and implementation • Virtualisation • Containerisation • IT Network and Infrastructure Protocols • IT Network Monitoring • Continuous Deployment • Cloud Computing Concepts • Infrastructure as Code

> *Understanding not only the technical, but also the business and human factors at play during the high pressure demands of modern software delivery processes, is essential in the modern discipline of DevOps.*

Semester 3: Advanced DevOps

Advanced IT Infrastructure for DevOps	DevOps in Practice
• Architectural Design to support DevOps • The DevOps supply-chain and PLM relationship • DevOps in the Public Cloud • Comparative Analysis of Cloud Offerings • Cloud Scalability and Elasticity3 • Load Balancing • Virtualisation Automation • Provisioning and Orchestration • Software Configuration Management • Software Provisioning Management • Security in the Public Cloud • Degrading systems gracefully • Chaos Monkey • Server-less Compute in the Cloud	• The DevOps paradigm/pipeline in practice requirements • Develop Continuous Integration/Test/Deployment Release management • Monitor and Learn • Feedback and Iteration • Detailed DevOps Case Study of the technical and human experiences of typical practitioners, e.g. - Google SRE (Site Reliability Engineering) - Intercom (Customer Messaging Platform)

Semester 4: DevOps Research

Research Methods	Research Project
• Academic Writing • Qualitative and Quantitative research • Surveys • Statistics	• Applied piece of Research in DevOps area • Encompasses a Proof of Concept/Prototype • Supplements DevOps Theory knowledge *This is an opportunity for students to carry out a piece of work which is at the cutting edge of the field and explores in depth a feature or element of that field. It is perfectly feasible, and there are many examples of this, for students to carry out their research project on a piece of work of direct relevance to their company or organisation. The academic team in TU Dublin (Tallaght Campus) have deep industry experience and have supervised and developed MSc. projects which explore business values, infrastructure automation and DevOps projects with real industry relevance.*

Technology Ireland ICT Skillnet

M. Sc. Applied IT Architecture (online)

In conjunction with Irish Computer Society and accredited by International Association of Software Architects. A Technology Ireland Skillnet funded programme. This programme is 80% online with two days per semester attendance required.

M. Sc. Computing with DevOps (online)

This programme was designed in conjunction with leading ICT companies such as Microsoft, Fidelity, IBM, Ericsson who form the Technology Ireland Skillnet. This programme is 80% online with two days per semester attendance required.

Non-Standard Applicants

Note for interested applicants: Next intakes for these Skillnet programmes set for January 2019. Standard admissions requirements include a relevant bachelor's degree at honours level. It is recognised that there are experienced and skilled potential participants for the programme who may not fit the standard entry profile. A non-standard admission process is available here which can be based on prior experiential learning and/or qualifier modules. These qualifier modules can be taken from September 2019 for admission in January 2019. Contact bfeeney@it-tallaght.ie or mhendrick@it-tallaght.ie for more information.

Accreditation of Master of Science in Applied IT Architecture by IASA

Technology Ireland ICT Skillnet

The TUDublin (Tallaght Campus) M.Sc. in Applied IT Architecture is the first of its kind in the world to be developed based on the IASA Five Pillars.

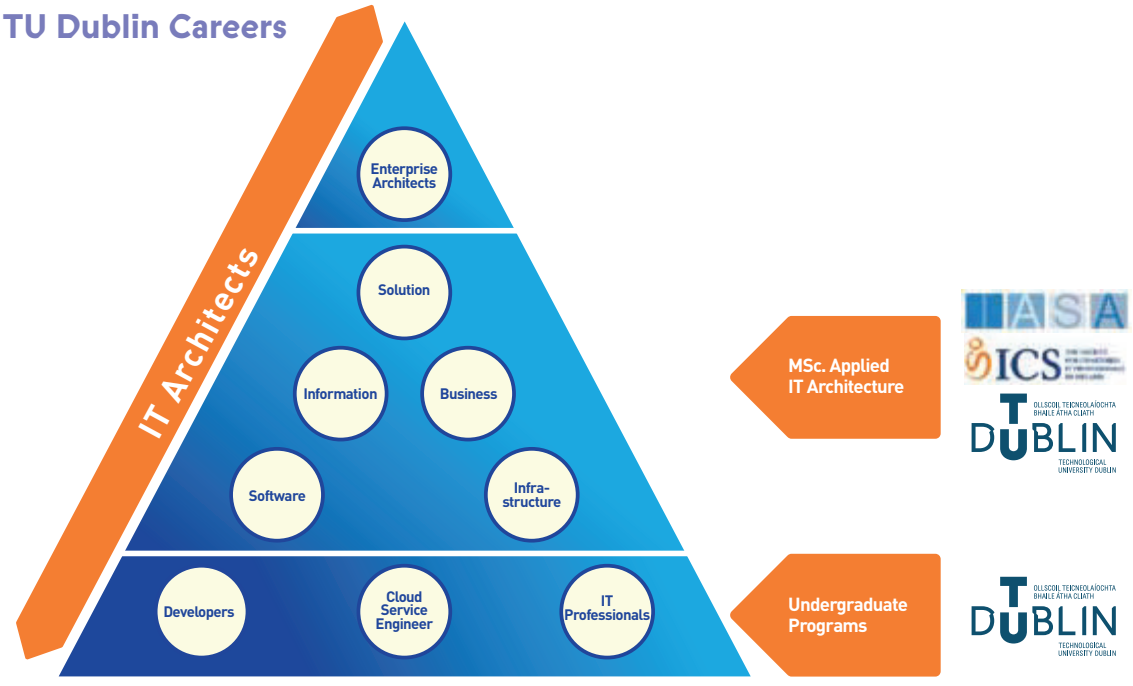
For the first time, candidates can gain a full Masters of Science degree in this specialist area through a mixed learning process with an emphasis on practical application in the workplace.

What is IT Architecture? (From IASA Global)

Architecture at IASA is the practice of business, organization or client gain through the application of technology strategy. It is the art and science of designing and delivering valuable technology strategy. At its core, the ITABoK describes how to create a professional person or group of professionals who can consistently find new applications of technology to generate positive outcomes for their client or employer. IT Architects:

- Retain depth in technical skill as well as business skill
- Able to successfully work with both business and technical staff
- Develop their own or others business cases based on technology driven innovation
- Retain the ability to deliver projects on those business cases
- Deliver business projects more successfully based on outcomes than others

TU Dublin Careers



PROJECTS

Case Studies of Deploying and Enabling Low-Latency Microservices in a Kubernetes Multi-Cluster Environment

Emerson Gomes
Department of Computing, TU Dublin, Tallaght, Ireland
X00169710@myTUDublin.ie

Introduction

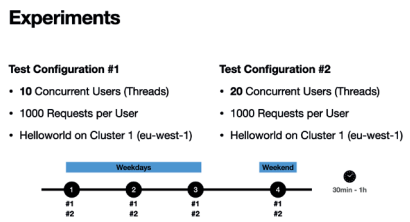
The adoption of Kubernetes as the orchestration engine for container-like applications has been considered by many companies who are looking to implement an infrastructure that provides high-availability and horizontal scaling features for its applications hence most of the cloud providers nowadays offer a managed service for Kubernetes. Moreover, companies are now managing a fleet of Kubernetes clusters while they are expanding their services globally. This research paper presented an architecture and common requirements to launch a multi-cluster environment using Service Mesh as the service-to-service communication layer, and carried out a series of experiments in a form of case studies to analyse the performance of two main Service Meshes - Istio and Linkerd, extracting data such as CPU, Memory, Throughput and Latency figures under different load configurations on AWS (eu-west-1 and us-east-1) regions. As a result of these case studies, both Service Meshes have enabled a microservice to run in two interconnected clusters, having Linkerd shown a great advantage in terms of CPU and Memory performance which can also enable a cost-efficient infrastructure while delivering a good Request per Second throughput.

Research Questions

- **RQ1**: What are the best practices and common requirements for deploying Service Mesh technologies?
- **RQ2**: How to provide and enable a lower-latency application using Service Mesh in a multi-cluster environment?
- **RQ3**: Can a multi-cluster strategy really help companies to provide better user experience at a lower cost?

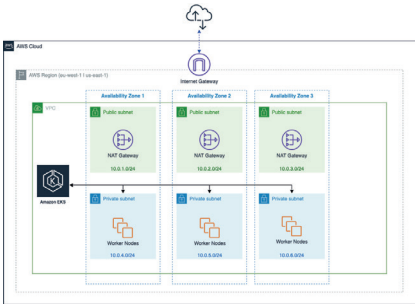
The objective of this experiment is to assess the behaviour of different Service Mesh components and configurations aimed to address the listed Research Questions in this paper.

Testing Strategy



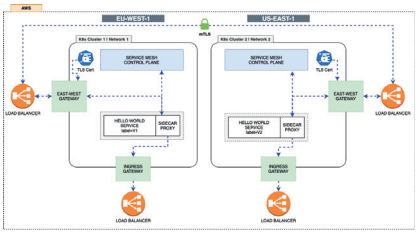
Architecture

1. Kubernetes Architecture



To achieve the desired multi-cluster state, a minimum of two clusters were required, having each one of them running in separated regions to promote a higher availability at a global scale. Terraform - version 1.0.5 - is the automation solution chosen to help with the provision of the infrastructure in the cloud environments. Amazon Web Services (AWS) is the cloud platform selected due to its relevance in the market and by the number of services out-of-the-box offered. including EKS - Elastic Kubernetes Service, a managed Kubernetes platform. Europe (Ireland) and the United States (North Virginia) were the two selected AWS regions for hosting the EKS clusters.

2. Multi-cluster Architecture:



Istio is an open-source service mesh created in collaboration between Google, IBM and Lyft to address challenges around microservices communication and security in a distributed environment.
Linkerd is another open-source created by Buoyant and donated to the Cloud Native and Computing Foundation (CNCF) initiative. It is comprised of a UI (dashboard), control and data planes similar to Istio.

Research Findings

Test Configuration #1				
	CPU (mi)	Memory (MB)	Latency (s)	RPS
Istio	23.60	281	2.33	3.75
Linkerd	0.99	36.49	1.42	4.63
	96% less CPU than Istio	7.7x lower than Istio	0.91s faster than Istio	23% more requests than Istio

Test Configuration #2				
	CPU (mi)	Memory (MB)	Latency (s)	RPS
Istio	39.63	286	4.66	3.92
Linkerd	1.13	38.77	1.41	4.78
	97%	7.4x higher than Linkerd	3.25s faster than Istio	22% more requests than Istio

Conclusions and Future Work

Given the results presented and discussed in this research paper, it can be concluded that Linkerd is the service mesh type that enabled a lower latency for the tested microservice in both load scenarios, providing reliability over the expected performance.
Linkerd Service Mesh was also the option that has the potential to help companies to provide a better user experience at a global scale with lower cost, due to the low CPU and Memory usage shown in detail by this research paper. It is also understood that the cost factor is a broader and open question, but demanding such little resource, Linkerd test results proved to be a cost-efficient choice.
Finally, this research paper also presented some interesting findings with a potential for further case studies and benchmarking analysis, including the understanding of memory performance dump and the grained details of Istio's Envoy and Linkerd-proxy code implementation to handle threads for the CPU consumption.

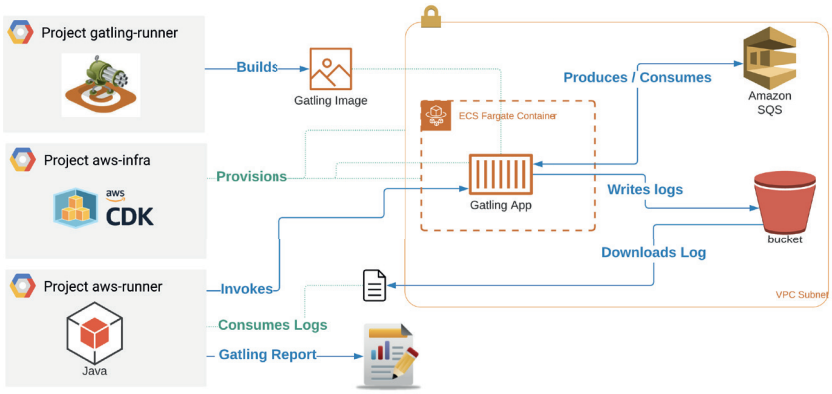
Evaluation of Amazon SQS in the face of Producer / Consumer Reliability Patterns

Martin Devaney
Department of Computing, TU Dublin, Tallaght, Ireland
X00169687@myTUDublin.ie

Introduction

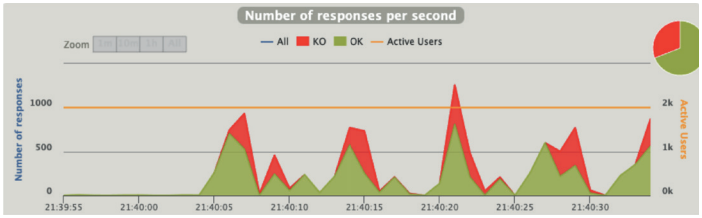
Understanding how different configuration settings affect the behaviour of SQS queues when messages are produced and consumed from it, is the core aim of this research. AWS provide two types of queue - Standard queue and FIFO queue, both of which have configuration that can be set to alter the queue behaviour. The research investigates how the queue performs when the configuration is changed against different sizes of loads.

Test Harness

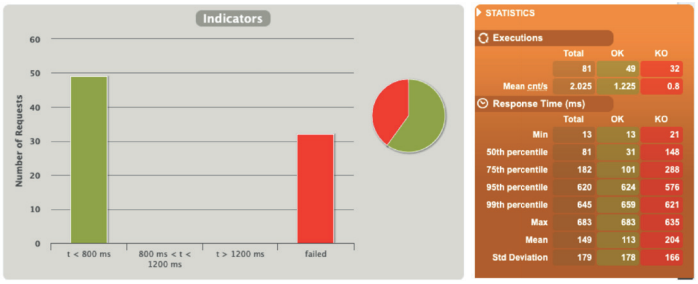


Performance tests were created using Gatling Load Testing and were executed inside an AWS context (using Fargate) to minimise the connectivity overhead.
The solution is comprised of three projects: a Gatling application to load test, a CDK application which provisions the AWS resources and a Java application which co-ordinates the tests by invoking the container with different parameters at Runtime.
The Gatling application writes and reads messages into the SQS queue and can simulate multiple message producers or consumers running simultaneously. The scenarios tested for this research executed tests for 10, 100 and 1000 simultaneous producers / consumers.
The Java application invoked individual scenarios and subsequently downloaded the generated Gatling log to generate a Gatling report locally.
NOTE: The Test Harness is based on an Open Source project created by Richard Hendricksen.

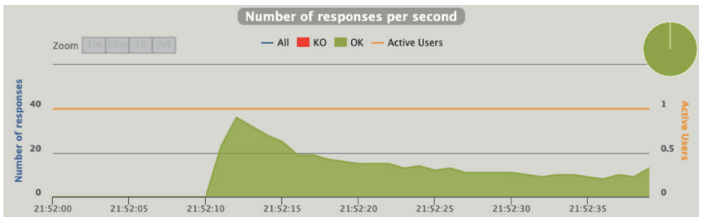
Key Observations



1000 Producers and Consumers simultaneously using the same FIFO SQS queue results in erratic behaviour with the average throughput per second falling from 303 requests for 10 Producers scenario to 182 in this scenario. The queue struggles to process requests with extremely heavy traffic.The green areas are successful requests while the red are unsuccessful requests (they exceed the maximum allowed throughput of 300 requests per second).



This scenario measures the FIFO queue performance where a Producer is randomly sending a deliberately oversized message payload that cannot be processed by the queue (max payload size is 256kb).
The throughput in this scenario for 10 Producers is just 2 requests per second - in the baseline scenario the average throughput for 10 producers was 303 requests.
The failures here should not be confused with requests that were rejected for exceeding the queue throughput limit (in other scenarios) which did not adversely affect the queue performance on its own.



This scenario demonstrates the impact of an increasing (but allowable) payload size when producing into the queue for a single Producer.
It can be observed that as the payload increases, the throughput per second decreases from 38 requests per second to approximately 11 requests per second.
This demonstrates that the queue throughput is impacted by the size of a valid payload.

Conclusions and Future Work

The findings arising out of this research are that the configuration settings can be used to apply a different behaviour and in certain scenarios can perform better than the default configuration. Interesting observations were noted about the impact of extremely high volumes of requests that seem to cause the SQS queue to dramatically decelerate the rate of requests being processed. The impact of the payload size when producing into the queue was also surprising with the queue struggling to process requests where the message payload was over the allowed size.
Taking the time to consider the rate at which messages are expected to be sent or received from the queue, the length of time to process those requests, the size of the requests, if message ordering or guaranteed delivery are required were identified as the key considerations when making design decisions.
Future work would be to perform similar analysis for SQS queues with encryption enabled as well as investigating how Batching could improve on the average throughput speeds attained in these tests.

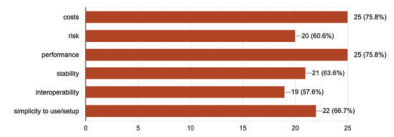
Comparing AWS Native Lambda to AWS Container Image Support Lambda

Eric Strong
Department of Computing, TU Dublin, Tallaght, Ireland
X00169645@myTUDublin.ie

Introduction

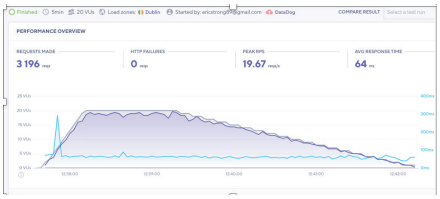
This research project compares native AWS Lambda with Container image Support Lambda and provides an analysis into findings to answer the question "is it feasible to package up a deployment and use these within the Lambda service?". In the past Lambda deployments were limited to 250MB zip uploads. Now with the introduction of Container Image Support, Lambda can have up to a 10GB image along with dependencies. This is a game changer for software solutions that require large deployments or data and will allow the free up the larger deployments to allow adaptation in the cloud more easily. The Lambda Service now offers the ability to now obtain containerized software functionalities and integrate easily with a Container registry provider and be scaled in the cloud without a concern for infrastructure management. In addition to better dependency management, isolation, security, and more invocation methods such as Synchronous, Asynchronous, Streaming, Batch and Event driven mechanisms. This research dives into a comparison of both and will give insight for this new Lambda offering for enterprises to make use of. Use cases such as DevOps, Machine Learning, Artificial Intelligence, and Data Science will now become even easier to integrate with Serverless technologies due to a container-based deployment and an improvement on image to package up dependencies. The project makes a fair comparison of Lambda variants between Performance, responsiveness, Cost and Integration with other AWS services.

Technology Adoption Survey



Lambda Responsiveness

Container Image Lambda:



Native Lambda:

Research Outcomes

1. Container Image Lambda Demonstrated Control Over Codebase:

- Calling the functions showed no errors: Throughout the experiment and test scenarios. Given running a container behaves the same as that of the local environment, zero failure cases of Codebase were found in the metrics.
- No unexpected behavior was evident: Memory and CPU findings revealed container image support was better in some cases. Where CPU may be somewhat slower but only by 20ms of CPU time. This shows that packaging code up and running in Lambda in comparison with zip deployments will give a sense of control and predictability. Cold starts were less frequent in Container Image Lambda compared to Native Lambda.

2. Container Image Lambda Performed better than Native Lambda:

- CPU metrics, responsiveness, duration of the Container Image Lambda versus the native image Lambda under the same test criteria Container Image Lambda proved to be similar if not better in most cases. Given the added advantage of Container dependencies
- Memory management for Container Image Lambda had better memory utilization than that of Native Lambda. However, CPU utilization proved to be slightly slower

3. Container Image Lambda Integrates Easily with other AWS Services:

- Integration API Gateway and EventBridge showed to be the same and seamless

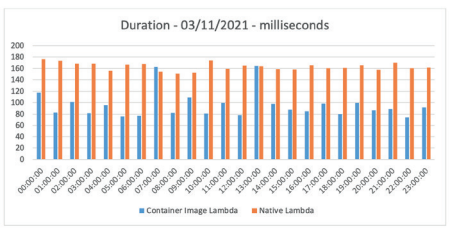
4. Container Image Costs were Comparable with Native Lambda with the minor exception:

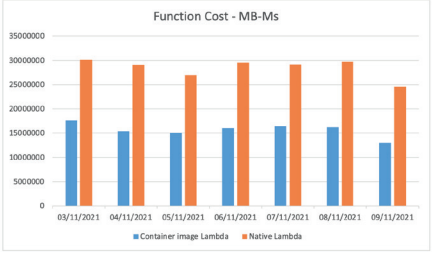
- The costs gathered from the Lambda executions showed to be lower in comparison of Native Lambda with Container Image having fewer cold starts - Container Image Lambda charges for init duration

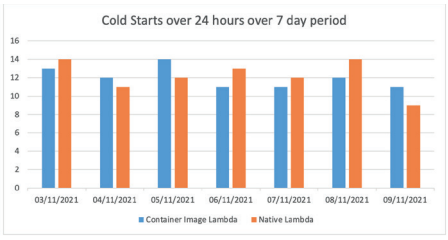
5. Container Image is a good competitor to Native Lambda:

- Offering good performance, memory management, costs, integration with existing services. The added deployment size benefits are huge and a game changer

Research Findings







Conclusions and Future Work

This research project provided a comparison experiment between Native Lambda and Container Image Lambda where steps were outlined to create the experiment and monitoring was hooked in and test scenarios and criteria were outlined. Tests were conducted to capture results and provided metrics to compare both FaaS offerings. The test result findings proved that it was feasible to package up code and use the Container Image support offering based on performance, responsiveness, costs and interoperability. Results proved Container Image Lambda was a contender compared to Native Lambda and had offered a better result in all of the test criteria. Only Python 3.8 was used but AWS offer other runtimes such as Java, C, Rust, Go and NodeJS. For future work a custom runtime with Container Image Support would be considered a great experiment to then make a comparison between runtimes using the same Container Image Support feature. In addition a suggestion for future work is to investigate the creation of a custom runtime and uploading to Container registry along with codebase. This would make for an interest and complex technical project. Explore the use of container image FaaS with other providers such as GCP,Azure and compare against AWS offering.

Comparison of Multi-Cloud Infrastructure as Code Tools

Diarmaid O'Keeffe
Department of Computing, TU Dublin, Tallaght, Ireland
X00169640@myTUDublin.ie

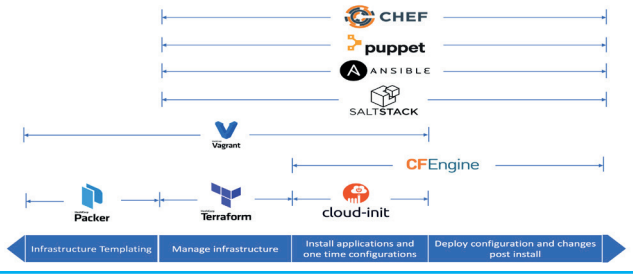
Introduction

For many companies today, having reproducible and transferable infrastructure resources during, or after migration to cloud platforms is extremely important. Each cloud platform provider has unique offerings in terms of cost, security, features etc. that their customers may seek to avail of. Defining 'Infrastructure as Code' (IaC) can assist with this in a major way. While the major cloud providers have their own native IaC tools, the multi-cloud IaC tools which are available in today's market can help with avoiding vendor 'lock-in'. Workloads must be transferable without significant upheaval costs and changes. This research sought to identify the optimal multi-cloud IaC tool available in the market today. Initial information about each tool (such as the ease of setting up each development environment) was recorded and a pre-defined testbed was implemented on AWS and Azure.

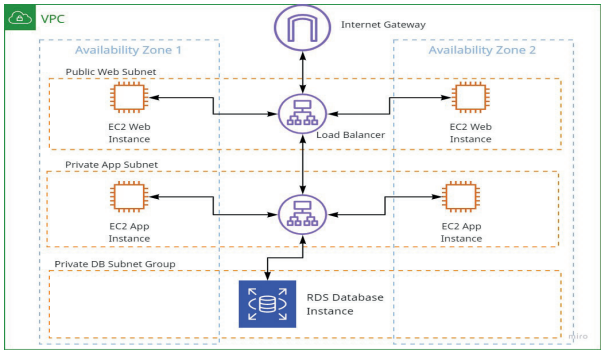
IaC & Orchestration Tools

Iac vs Orchestration Tools

The graphic on the right shows the typical (separate) responsibilities of IaC tools and orchestration tools in the lifecycle of application infrastructure. Traditional IaC tools like Terraform are used only until the application infrastructure is built. From there, tools like Ansible and Puppet configure application data. This research sought to understand the broader capabilities of tools like Ansible and Terraform across each field, from infrastructure provisioning to configuration management of application data. Pulumi, Ansible and Terraform were the tools examined within these criteria .



AWS Testbed Diagram



Testbed Definition

The diagram on the left shows the final testbed implementation (AWS specific naming convention) for this research. This infrastructure was replicated using each of the three tools across AWS and Azure platforms. Based on the results of this implementation, metrics were gathered and used in analysis when performing a comparison of the IaC tools. Web, application and database subnets were created, along with load balancers, security groups, access rules and virtual machine instances. Apache Web server was installed and minimally configured on the public internet-facing web instances. Terraform and Pulumi (with their declarative programming), had the added benefit of managing resource dependencies. This enabled efficient tear-down of the infrastructure with a single command. Ansible (with imperative programming) did not have the same capability, therefore, it's managed resources had to be sequentially removed in order of resource dependencies.

Analysis

The tools were compared using qualitative and quantitative methods of analysis, where each point of comparison was assigned a resulting score out of ten. As can be seen in the final total scores (on the right), Terraform narrowly outperformed Pulumi in the overall comparison, with Ansible receiving the lowest score of all three,

Ansible, with it's simplistic YAML syntax, took less time to implement the testbed than the other tools and also performed well in configuration management. However, it was by far the slowest in terms of execution speed and was poor at state change detection.

Pulumi (implemented with Python), outperformed the other tools in relation to the ease of code migration across cloud platforms. The use of Python libraries and features (like debugging tools) made troubleshooting issues easier to address than with Ansible or Terraform.

Terraform performed consistently well across many of the areas of comparison. Managing state change between executions was easiest using Terraform and it's documentation base was more centralized and accessible to new users than Pulumi or Ansible's documentation.

Comparison Table & Total Scores

Comparison of Multi-Cloud IaC tools			
	Pulumi	Ansible	Terraform
Execution Speed	7	4	7
Scaling of managed resources	7	5	8
Development time required	6	8	5
State Change Detection	6	4	8
Ease of use of definition language	7	7	6
Platform Migration Capability	7	4	6
Quality of documentation	5	6	9
Configuration Management Capability	4	8	4
Total	51	46	53

Conclusions and Future Work

It is clear from this research that a tool like Ansible (which is still seeking to gain a footprint in the area of Infrastructure as Code on cloud platforms), is not yet mature enough to compete with the declarative model offered by native IaC tools. However, as an increasing number of modules related to management of cloud resources are published, it is possible that they will seek to implement some means of mimicking the ease of management of the other two tools. Terraform (as of the time of this research) is the tool of choice when it comes to provisioning and maintaining infrastructure across multiple cloud platforms. Pulumi, with further improvement in key areas, can match (if not supersede) the position of Terraform as the IaC tool of choice. For future research in this area, it may be useful to examine the effectiveness of pairing a traditional IaC tool with a traditional orchestration tool. This scenario would involve implementing a solution using combinations such as Saltstack and Terraform and comparing them to Ansible and Pulumi.

A Comparison of Asynchronous Middleware as a Managed Service

Colm Leheny
Department of Computing, TU Dublin, Tallaght, Ireland
X00051965@myTUDublin.ie

Introduction

Asynchronous middleware platforms are increasingly being used by distributed systems to manage the flow of data between their components allowing them to interoperate. This research project provides a comparison of two managed asynchronous middleware services provided by GCP and AWS. The comparison focuses on a number of typical enterprise use cases and three metrics: Configuration, Performance and Cost.
The motivation for this research is to provide technology architects with a methodology for comparing the asynchronous middleware solutions provided by both GCP and AWS. Although these services are implemented differently, by providing a limited number of use cases and a simple architecture for capturing a number of metrics we can make a comparison that will provide technology architects with some insights, that may help decide on the correct technology to use and even influence the design of a system.

Performance

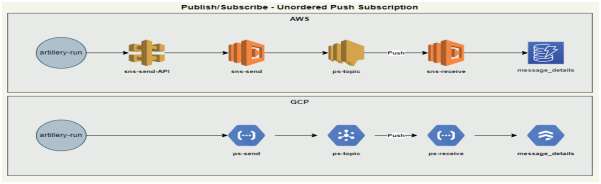
Two performance metrics were selected, namely Throughput and Latency.
Latency: this is the time it takes for a message to be published to the queue or topic.
Throughput: this is the time it takes for a message to be received by the subscriber. To capture the data required to compare these two metrics, the configuration for each use case was developed on both AWS and GCP. This architecture provided the ability to run tests and capture data to provide data that could be analysed further.

Cost

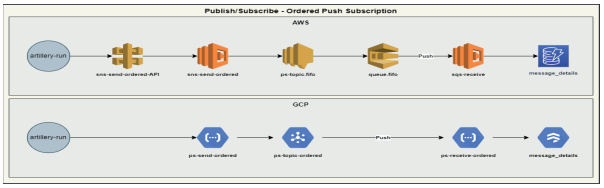
To estimate the cost of running the above architecture on both GCP and AWS, detailed pricing calculators are provided by both cloud service providers to estimate costs against all their services, including: SNS, SQS, Lambda, Pub/Sub and Cloud Functions. To implement a practical architecture for this research Lambda and Cloud functions are used both to send and receive data from the topics/queues used. The cost of running these serverless functions will be included to provide information on how much it would cost to implement this architecture and to determine are there any prohibitive costs associated with this configuration.

Configuration

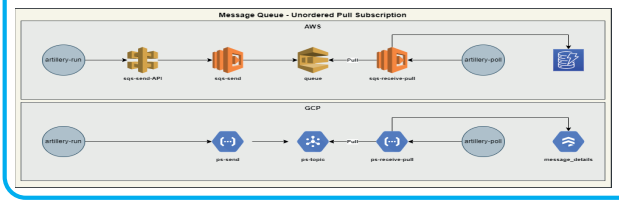
1. Publish/Subscribe Unordered Push:



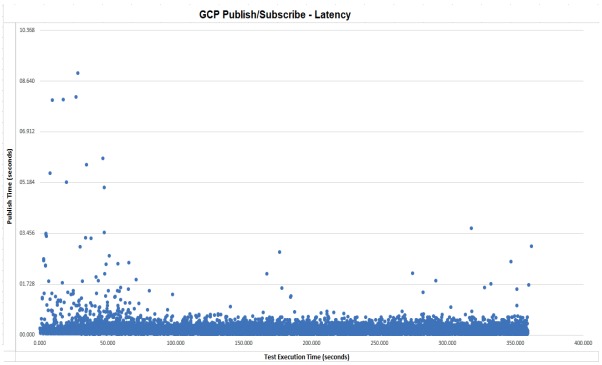
2. Publish/Subscribe Ordered Push:

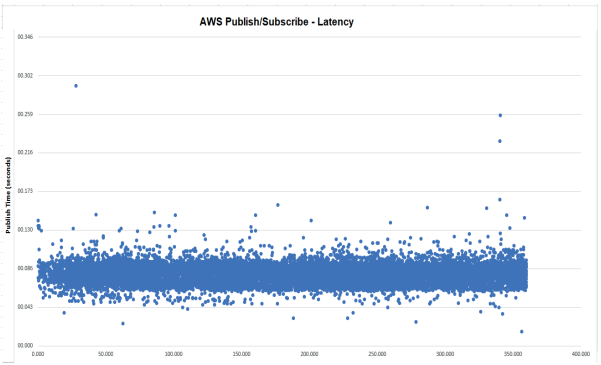


3. Message Queue Unordered Pull:



Performance Overview





Conclusions and Future Work

Both GCP and AWS provide extremely performant solutions that allow for asynchronous communication to be implemented easily. A number of differences exist between how GCP and AWS implement each solution - including: Message Ordering, Message Delivery and Cost. For example, in relation to message delivery, AWS provides both at-least-once and exactly-once delivery and GCP only supports at-least-once delivery.
There are a number of areas that could provide further research opportunities, for example, Python was the language used in all cases whereas other languages such as Node may provide better latency and therefore the ability to publish more messages per second to a topic or queue. Also, focusing on other subscribers available on AWS - other than Lambda functions may give an opportunity for some interesting research.

Container Security Analysis

Anthony Staunton
Department of Computing, TU Dublin, Tallaght, Ireland
X00004139@myTUDublin.ie

Introduction

With DevOps being adopted by organisations on a constant basis, the migration from monolithic applications stacks to micro-service based architectures is becoming a normal task. With the move to these micro-service based applications more organisations are reliant on containerisation to provide their end solution / to supplement their existing deployment or delivery mechanisms to their customers. While it is often the case that these organisations would have need to align to certain standards (often to be certified as compliant to regulations in certain sectors) and they would understand any tasks they would have traditionally performed to align to these criteria, there would often appear to be a lack of procedures in place to ensure their micro-services (and by extension the containers they use) are compliant with any security regulations.
In this work the author will look at some previous research on how containers may be secured based on static analysis, and will provide some data related to the dynamic analysis of the containers while they are running on a system. The author then hopes to convince the reader that a combination of static and dynamic analysis of container images, and their resultant containers, is required in order to provide a level of confidence to any organisation, as to the level of the container security (or the lack there of).
The author also wishes to provide a test environment that would facilitate not only the replication of the results shown in this work, but to also provide a basis on which organisations or other research projects may extend upon. The creation of the environment has been automated where possible, with an aim to provide ease of use from within the environment but to also provide a mechanism to easily extend upon the functionality if required in the future.

Is Static Analysis Enough?

It was shown that static analysis is not enough to provide confidence in running containers in an organisation, as a system administrator the data provided would not show enough detail to determine if the containers will not introduce security issues.
Previous works have suggested monitoring containers at run-time and creating SELinux or AppArmor profiles to secure the system. In the authors experience a lot of system administrators disable AppArmour or SELinux as they find it interferes in their daily tasks (not all applications ship with security profiles and administrators do not want to spend time creating them). Therefore in this work other methods of dynamic analysis were investigated, namely antivirus scans and network monitoring

Rootless Containerisation

Rootless Containers have been suggested as a best practice. In this work it rootless containerisation was investigated and all issues encountered were discussed.

Topic Overview

In previous works it was suggested that static analysis be enhanced by dynamically generating security profiles based running containers. It is the authors experience that administrators do not use these profiles so alternative dynamic analysis solutions were investigated.

Host UID	UserNS UID
1000	0
100000	1
100001	2
...	...
165535	65536

This works also looked at running containers in rootless mode. Although this is suggested as best practice it was found to cause numerous issues for analysis tasks and also in the choice of tools.

Conclusions and Future Work

It was suggested, as a consequence of this work, that static analysis could not be relied upon as the sole determination of a containers insecurity. Therefore, with static analysis alone, it would be difficult for an organisation to determine if it is safe to run specific containers within their environments (be they development or even test environments). It was also intended to demonstrate that even with some simple dynamic analysis (e.g. an antivirus test and some network scanning) more confidence can be attained in the security of container images and the detection of issues during run-time. The authors suggest that RQ1. has been answered, some level of dynamic analysis should be performed.

Although running docker in user space caused issues it is the opinion of the author that it is not good practice to give elevated privileges to all users in order for them to run containerised workloads. Therefore rootless container functionality was retained knowing it would have an impact on the required effort in testing this work. The author also suggests that the fault does not lie in running the containers in user space but is due to the tool-sets having little or no interaction with each other and thus causing duplication of effort and/or inefficient use of resources.

The author suggests that as a mechanism for securing a production environment "rootless" containerisation would be of benefit. However if implementing in a test or development environment using containers in user space could cause issues for testing and validation. It is also possible that the use of other user space containerisation tools, such as Red Hat's Buildah, Podman and Skopeo may provide a solution that could answer this question more definitively and may be the basis of future work.

All container images used in this testing were exported to tar (while conducting the test runs). These archive files, the log data and all data generated from testing has been uploaded and shared in the following locations

https://drive.google.com/drive/folders/155Udb_U_OSRoU65OYdjEpqGq-J6veW6M?usp=sharing

The code used to create the Virtual Machine for the test environment and all scripts etc can be found in the following git repositories. The README.adoc lists the script locations within the repository and lists the commands / step required to create the VM, add the test data etc.

https://gitlab.com/nerdyninja/mscdevops

Comparing Microsoft Azure Web Apps and AWS Elastic Beanstalk Hosting Solutions for Business Adoption.

Gary O'Hara
Department of Computing, TU Dublin, Tallaght, Ireland.
X00169654@myTUDublin.ie

Introduction

The public cloud can be a great solution for businesses to adopt & allow them to meet with demanding customer needs, market dynamics, scalability & performance requirements. There are several services to choose from & this research specifically looks at the Microsoft Azure Web Apps & AWS Elastic Beanstalk solutions in the Platform-as-a-Service (PaaS) model. The research not only looks at the comparison between these selected services, but also the business decision factors driving this environmental modernisation approach. What factor does a business focus on as they investigate these solutions? Why might they choose one solution over the other, or potentially use both for a multi-cloud strategy? Which solution is easier to use? To produce useful findings, our research methodology used a number qualitative & quantitative methods: 1: Researching current literature & publications on these solutions. 2: Conducting interviews & surveys with Business Technology Leads & Operational Employees to capture the wider view, the thought process, & the migration blockers when it comes to choosing a PaaS solution. 3: Platform testing with custom user actions scripts (users: 50/100/500/1000) conducting registration and login tasks to produce successful or failed response time results. 4: Capturing the user experience to identify which platform is easier to use.

Researching

Our research began with investigation into existing literature to identify & build upon the explored work & projects in this area. Publications & media were found & used as comparison material to allow us to answer the research questions.

Business Adoption Decisions

From the qualitative research results, the conducted interviews & survey answers from Business Executives & Operational Employees shared many similarities. The top survey areas to drive businesses to these solutions are: the availability of cloud skills, the capabilities & supported frameworks & languages, the ability to achieve data residency, the support solutions offered, the learning solutions/documentation available, funding opportunities for customers to drive the migration, & assessments to drive business cases for stakeholder approval. The interviews produced interesting remarks, "we're a Microsoft house", indicating that their preference is for Microsoft Azure Web Apps, & another phrase: "Our developers are already AWS certified & familiar with the AWS PaaS service", siding towards AWS Elastic Beanstalk. Even though these phrases were expressed during the interviews, the representatives said they were open to learning about what each platform offers & they can then decide with supporting information & testing as to which platform they would go with. In some cases a multi-cloud strategy was specified.

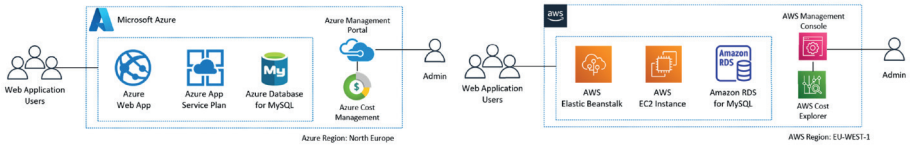
User Experience

Which solution is easier to use? We have identified that both Microsoft Azure Web Apps & AWS Elastic Beanstalk have learning curves attached. Microsoft have taken the more user-friendly approach for customers, with more guided service creation wizards, less compute configuration options to choose from in the low to mid-range tiers & more structured service plans to make for an easier deployment to the Azure Web App service. AWS offers more compute configuration choices, less structured service plans but a very capable solution for the advanced user who understands the options. The documentation from both providers is good, but again Microsoft offering that more user-friendly structure.

Platform Testing

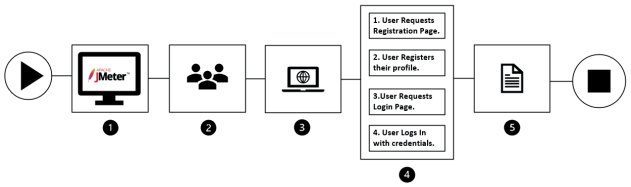
1. Solution Environments Setup to receive user actions:

The solution environments were created with cost as the constant. $100 per month per environment. Scope: Identical application: PHP frontend/MySQL backend, single instance, single region to remain within budget.



2. Performance testing setup and User Testing Flow to send user actions:

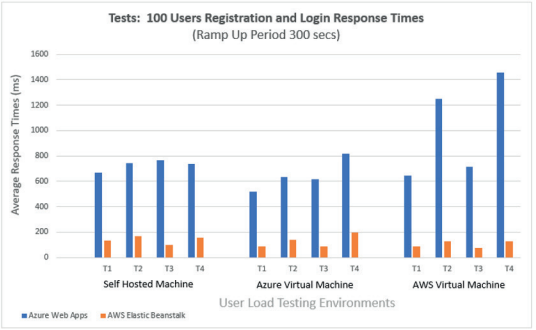
Three separate environments were created to send user actions to the individual web applications (Self Hosted Machine/Azure Virtual Machine/AWS Virtual Machine). Each environment sent users (50/100/500/100) to action tasks using custom scripts over a ramp up period of 300 secs to mimic real life scenarios.



Steps. 1: Apache Jmeter test plan. 2. Jmeter custom scripts sends user load to Azure & AWS. 3: Azure & AWS receive user load. 4: Custom script user tasks. 5: Response Times logs collected for analysing.

3. Sample Response Time Results from 100 users completing actions on Azure & AWS:

The chart shows the average response times across the results, Web Apps is resulting in a 797.5 ms average response time compared to Elastic Beanstalk's 123.4 ms average response time. Both platforms have a 100% pass rate on the tests. The AWS platform allowed for better performance due to the more powerful compute choice at the tested price point. The Web Apps S1 Plan began to fail in the 500/1000 user tests.



User actions:

T1: User requests registration page.

T2: User registers their profile.

T3: User requests login page.

T4: User logs in with credentials.

Conclusions and Future Work

Can a business simply just pick one of these solutions, and with the click of a few buttons, upload & deploy a web application to a live environment, in theory & practice: Yes as there is enough guidance in the provided docs & intuitiveness in the platform & UX design. But should it be done this way? No, like most things, preparation is key, landing zones, (CAF) Cloud Adoption Frameworks, (WAF) Well Architected Frameworks, policies & governance of the cloud environment are fundamental to get correct as a business migrates to the public cloud model for their future strategy. Are these PaaS solutions comparable? From the research & findings we can say yes from a high-level point of view, Microsoft Azure Web Apps & AWS Elastic Beanstalk are comparable, but as you investigate & develop each service, the differences begin to appear in the configuration options, the level of knowledge required to use these services, the compute power selection, the structured & unstructured tiering & pricing options, & some UX design differences. Future work. To build onto these findings, the premium tiers of Azure Web Apps can be investigated to compete with AWS. In testing, we found the web application tier was the bottleneck with failed requests before reaching the database tier.

Auto-scaling containers: Analysing and optimising container-based workloads in OpenShift.

John Murphy
Department of Computing, TU Dublin, Tallaght, Ireland
X00169711@myTUDublin.ie

Introduction

In this paper, we will examine the auto-scaling feature of containerisation in relation to how it performs on Red Hat OpenShift. Auto scaling is a mechanism that allows for more than one running instance of an application to be created when demand for that application increases. Conversely, when demand decreases the superfluous running instances are gracefully shut down. We will examine how the platform performs when asked to auto-scale various types of applications using its initial baseline thresholds for instantiating a new instance of the application. We will see if this initial threshold is optimally set, if not then they will be adjusted to a more or less tolerant value to better handle specific workload types. We will examine containers running on a cloud-based OpenShift clusters. From the results of the tests we will make a series of general recommendations for setting auto-scaling triggers based upon application types. If applicable these recommendations will be compared against default auto-scaling triggers that may exist; this is with a view to determining approximate increases in efficiencies on the OpenShift platform as well as any subsequent financial saving that could be made over a period. We used two values to measure the quality of service of our tests. These were response time and error rate. using these two measure we could determine whether the test passed or failed.

Traffic using Bell Curve model

Our first set of tests simulated user traffic modelled on a Bell Curve. It simulated linear increasing growth of traffic arriving at the containerised application. This type of growth modelled standard increasing traffic based upon high user demand for a product or service, or as a result of activity on social media. We used two type of containerised applications, the first a microservice based on SpringBoot. The second was a containerised middleware application that authenticated and authorises the user. As the traffic increased our thresholds were hit and the container autoscaled upwards in response to user traffic. If the containerised services passed our quality metrics then we increased the threshold at which they scaled thus decreasing the likelihood of a container scaling and restricting resources on the container.

Traffic using DDoS model

The second set of tests were run using a traffic model that represent a sudden surge in demand for the service. This could be as a result of a Denial of Service attack, or perhaps a regional node failure with traffic redirecting suddenly to the next available operating node. As in the first set of tests we measured against response times and error rates against our quality of service metrics and adjusted our thresholds accordingly.

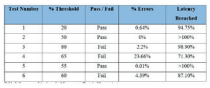
Results

1. Bell Curve Microservices

In this set of tests we found that microservices performed exceptionally well in the Bell Curve traffic tests. The service remained operational and passed all our quality of service tests up to 95 percent threshold (resources consumed). It was noted that at this 95 percent level there were a high number of errors. We found the area between 85 and 90 percent to offer the best level of response time and have the fewest processing errors.

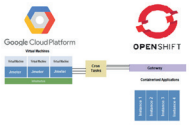
2. DDoS Microservices:

As in our first test we found the micorservices to remain functional even when placed under the stress of a DDoS attack. We had increased the tolerances of our quality of service metrics due to the nature of this traffic. The microservice remained acceptably operation up to approximately a 60 percent threshold. Once that was exceed we started to see increased errors and far longer response times.



3. Middleware container under Dell Curve and DDoS traffic

Our final set of tests were made against the middleware application. In these tests we found that regardless of threshold the middleware application cound not scale to demand quickly enough. This resulted in high number of errors being returned to requests, and a very high response time. The application often became unstable, only returning to normal after the traffic had abated.



Conclusions

There are several conclusions that can be derived from the testing that was undertaken. We established that microservices scale well to demand as expected. If the service is internal and behind a gateway then the scaling threshold can be set to between 85 and 90 percent of resoruces consumed before a new instance is created. If the microservice is external and subject to unregulated traffic then the threshold should be lower at around 55 to 60 percent. Containerised middleware applications do not scale well. In some confirmatory testing we 'pre-scaled' our middleware containers before running the traffic loads and they peformed far better. In the case of middleware applications especially those exposed to external traffic and providing core services. these should have more than one running container instance in order to better cope with surges.

Future Work

The area of scaling and containerisation is fertile ground. The first of which was what are the recognised surge models and how often are the likely to occur. There is research available in the trapezoidal and triangular models. Do other linear models exist and what is the frequency on their occurrence? Detailing these models, their likelihood of occurrence, and the context they are likely to occur would be valuable to understand. The human element of adoptions of new technology is also fertile ground. How well understood are the platforms by the operators? How likely are these operators to change settings away from pre-configured values? Would they be compelled to do so? What is the risk versus the reward? Is there are reluctance to take steps to optimise and adjust platforms due to an inadequate reward or a sense of 'why take the risk'? As with any new technology there is a shortage of expertise, how likely is the system administrator to "play it safe"? Finally, we used SpringBoot in our as our microservice. This is a widespread and established mechanism for delivering microservices. A newer framework for delivering these microservices does exist in the form of Quarkus. This framework offers far faster instantiation times than offered by the SpringBoot microservices. Would Quarkus microservices offer improvements particularly in instances of DDoS / regional failures?

An Investigation of the impact a CDN has on the performance of an Azure App Service

David Griffin
Department of Computing, TU Dublin, Tallaght, Ireland
X00169652@myTUDublin.ie

Introduction

Over the last 2 years demand for Internet services have never been higher. Cloud computing along with CDN offer some unique benefits to help with this demand such as content delivery networks delivering content closer to the end user and cloud computing being able to scale easily to meet growing demands. This research is aiming to test multiple ways of analysing CDN performance in order to help make an educated decision of which CDN option would be most suitable to use across the Azure cloud platform. The primary focus of testing is looking at the default web delivery setting and testing these under heavy traffic and analysing the results from both the user and server side. Secondary testing looked at making web page changes and cache control changing removing the general web delivery and analysing any performance changes.

Main Testing

General web delivery A total of five identical web applications were tested using JMeter measuring the response times, number of samples, transactions and network bandwidth. Tests were performed for 28 days, twice per day and overall results compared across each offering

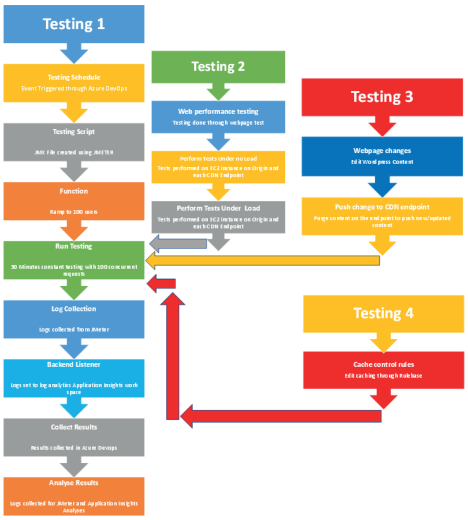
Web page performance Tests were performed using webpagetest.org to analyse time from first byte, rendering time, first contentful paint,speed index, webpage vitals, document completed and fully loaded times

Secondary Testing

Web page changes were made and then testing looked at how long it took to purge content on the CDN endpoints and for the changes to be re-cached on the CDN endpoints and display to the end user.

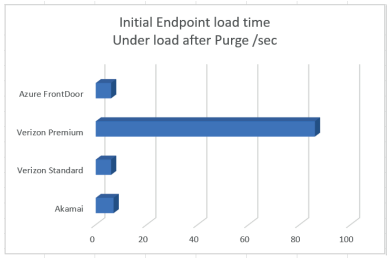
Caching control rule changes were made by changing caching rules for all images,then specific images and finally video content to analyse any changes in performance.

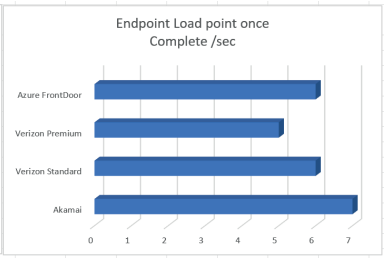
Testing Framework

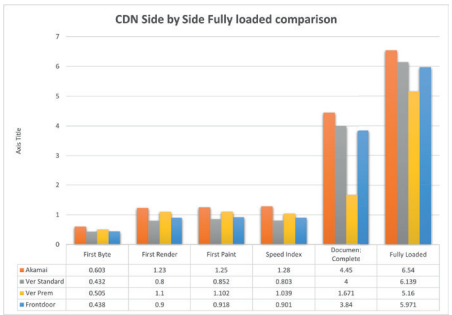


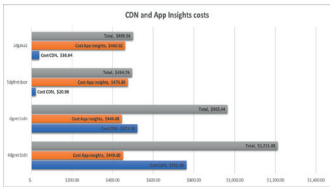
There are currently 4 different CDN options at the time of this paper, Akamai, Verizon standard, Verizon Premium and Azure Frontdoor. Tests were carried out across all offerings using various testing methods. Each test was carried out when there was zero traffic on the web application and then again when there were 100 concurrent requests over 30 minute period and repeated twice per day for 28 days.
The testing framework used a combination of JMeter load testing along with Web page performance tests from webpagetest.org in order to analyse both user and server side performance. Azure DevOps was used as the testing schedule and to execute the testing scripts.
Once all performance tests were ran all costs were analysed looking at web service, CDN Endpoint costs and log analytics.

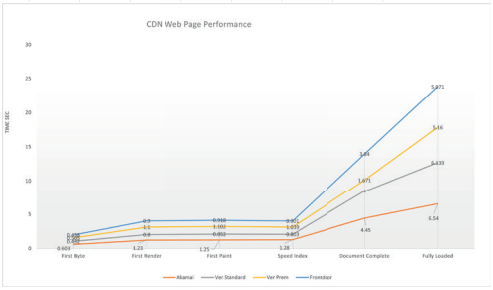
Testing Results

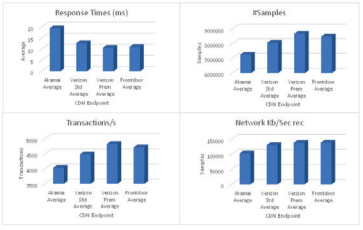












Conclusions and Future Work

Final conclusions Verizon Premium has the best overall performance across all tests being 15.71% faster in overall web performance and 4% faster in response times to its closest comparison being Azure Frontdoor. Taking in cost considerations, Verizon Premium is 189.29% more expensive than Azure Frontdoor standard so justifying the performance benefit needs to be considered depending on the type of application running.
Final conclusions Investigation comparing Azure Front door premium and Verizon premium could be investigated further. Azure CDN options were investigated in this paper which allows research across AWS, GCP or native CDN solutions.

Using Rust for Azure Functions

Rahul Ayyampully
Department of Computing, TU Dublin, Tallaght, Ireland
X00169650@myTUDublin.ie

Introduction

Number of customers using serverless functions in AWS, Azure, Google Cloud, and other cloud platforms increased in recent years and most customers started treating serverless functions as a critical part of their software stack. Since performance of a function affects both user experience and cost, it is key to find the right language, configuration, and platform combination to get the best performance and pricing from serverless functions.Rust is a system level language, proven to be offering significant performance and cost benefit in AWS Lambda. This research attempts to find whether similar advantage could be seen in Azure Functions by comparing cold and warm execution metrics of Rust functions with that of other languages.

Languages in Scope

Rust has no native support in Azure Functions and require a custom handler. Go Lang is the same situation and could be a great comparison. C# and NodeJs have native support in Azure Functions and were included in scope due to their popularity in the platform and programming community.

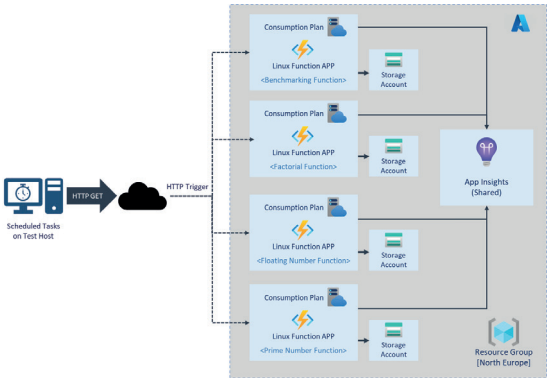
Test Functions

All functions used for the tests use HTTP trigger and accepts parameters through query string. For each language in scope, a lightweight function, accepting a string parameter and returning a 'hello' response was used for benchmarking. Apart from this, factorial, floating number and prime number calculation functions were used to collect cold and warm execution times and resource utilisation. This means that a total of 16 functions were created and deployed for the experiment.

Test Setup

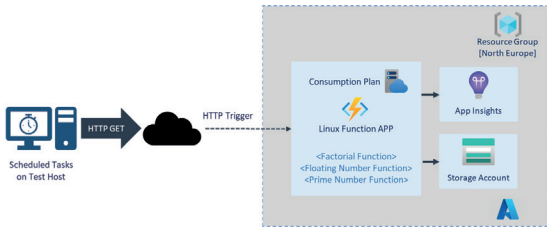
1. Single Function Instances

In this configuration, only one function was created per function app, resulting in 16 function apps (12 test functions + 4 benchmarking functions). Each Function App had its own storage account and all apps were configured to emit telemetry to a shared Application Insights instance. All the functions were triggered from a PC connected to internet on an hourly basis using windows scheduled tasks and batch files. Batch files contained curl statements to trigger the functions and pass necessary parameters through a query string. Each execution triggered a number of cold and warm starts of the functions and this was monitored using the telemetry emitted to App Insights.

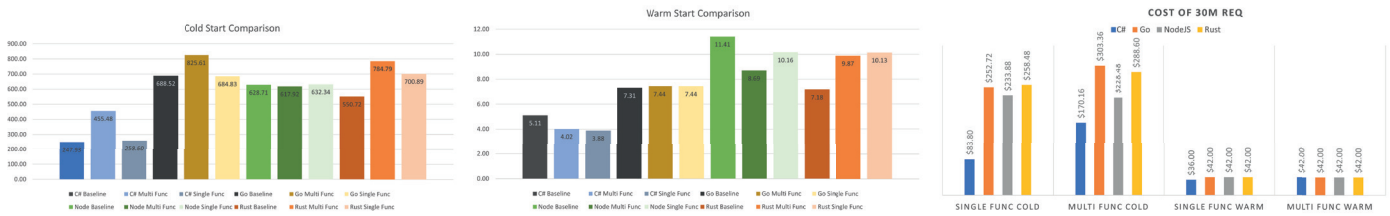


2. Multi Function Instances

This set up used only one function app per language resulting in just four function apps, each containing 3 test functions. A shared App Insights instance was used for this test as well allowing consolidated telemetry. Since the multi-function app creates and scales all the function instances in the app at the same time, only the first execution of the first function triggered on an app per batch experienced cold start.



Results



Even though Rust benchmarking function finished second for cold starts, the other functions fell behind both C# and Node in all tests and finished only fourth in the case of multi function instances. For warm starts, Rust once again finished second for benchmarking but fourth in single function and third in multi function instance tests. C# clearly outperformed other languages and Node was second best for cold starts while Go was second for warm starts. Performance clearly impacted pricing too and the languages with native support showed an auto warm up pattern reducing the number of cold starts. Languages needing a custom handler took longer to boot up for cold starts due to higher host initialisation time.

Conclusions and Future Work

C# outperformed all other languages in scope including Rust. The need to use a custom handler significantly increased the host initialisation times and degraded the cold start performance of Rust. Languages with native support were easier to create and configure and experienced significantly low number of cold starts due to an unexplained auto warm up anomaly called out in the thesis. Running more than one function on a single app can reduce the number of cold starts and bring advantages as all functions are warmed and scaled up together. Future research could be conducted to find the advantage of Rust, when used for more complex and intensive functions with external bindings and dependencies. Functions running on windows instances, Azure Durable Functions and functions on dedicated or premium hosting plan could also be analysed in future.

Predicting user frustration using virtual reality telemetry data

Phillip Ryan
Department of Computing, TU Dublin, Tallaght, Ireland
X00169673@myTUDublin.ie

Introduction

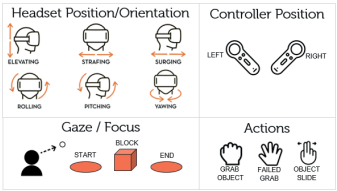
The interest and investment in virtual reality (VR) is increasing. Big technology companies are competing to build the metaverse, a digital corollary to our current physical world interactions. Virtual reality offers an immersive experience which heightens the users emotional responses. However if the user experience frustration whilst immersed in a virtual environment, the user has fewer reliefs then traditional software applications. In VR, this frustration can lead to the user exiting the immersion and future avoidance. Traditional measurement of the user experience can be costly and time-consuming with results available only after the experience has occurred.
Virtual reality headsets and hand controllers capture a large amount of telemetry data relating to user's position and interaction within the environment.
Is it possible that VR telemetry data could be useful in predicting user frustration?

Experimental Design

A VR application was created in which a participant performed a series of tasks which were designed to frustrate. After each task the participant answered a two item questionnaire on the learnability and ease of use, scored using a 5-point Likert scale. This self-reported measure of frustration was used as the dependent class variable. The telemetry data collected during the task was analysed using classification algorithms to determine if they could accurately predict the self-reported user frustration. In addition, participants were asked to provide demographic information to determine if any gender or age bias could be detected.

Data Collection

The telemetry data included a time series of:



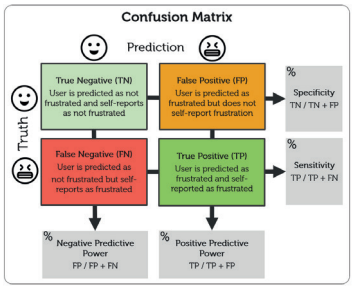
Only the first 10 seconds of telemetry data was analysed. The usefulness of any prediction is in allowing future software to potentially adapt or intervene to avoid the eventual user frustration.

Classifiers

Telemetry time series data was processed to extract 25 independent variables. Analysis via Pearson, PCA and RFE showed 11 had moderate or strong correlation with the Frustrated class variable. A check for multicollinearity reduced this to five independent variables to be used to attempt classification. Sample size is $(n = 25)$.
The machine learning algorithms selected to predict the binary classification were **Logistic Regression, Naive Bayes, Decision Tree and k-Nearest Neighbours**. To reduce bias k-Fold cross-validation was used. Due to the small sample size no deep learning algorithms were used.

Analysis

The mean accuracy of the classifiers were all >= 80% except for decision tree with an accuracy of just 61.67%. For this research question the underlying metrics needed to be considered using the confusion matrix which tabulates the prediction versus the truth. The True Positive (TP) results are of course very important, indicating the users with rising frustration for which an intervention could potentially be made. The False Positives (FP) are users who are miss-classified as frustrated but are in fact not, meaning we may waste resources intervening for a user who is not seeking relief for potential frustration.
Although False Positives are undesirable, worse are the False Negative (FN) miss-classifications. A False Negative is where the model predicts no frustration, but the user is self-reporting frustration. This is detrimental in VR and any predictive model needs to avoid or reduce this type of miscalculation. The best performance metric therefore for this research question is Sensitivity, the proportion of identified frustrated users over the actual frustrated users. The classifier which can deliver a high sensitivity is one which can detect almost all frustrated users and miss-classify only a small proportion of frustrated users as not frustrated.
The results for the classifiers shows that the Naïve Bayes delivers the best performance on sensitivity at 90.91%. The Naïve Bayes model was able to correctly predict 10 of the 11 frustrated samples as well as miss-classifying just one, the best performance across True Positives and False Negatives. To ensure statistical significance, two-tailed t-tests were performed on the Naïve Bayes models versus the other classifiers and all had p-values well below 0.05.



Metric	Logistic Regression	Naive Bayes	Decision Trees	k-NN
True Positive (TP)	8	10	7	8
False Positive (FP)	1	3	4	2
True Negative (TN)	13	11	10	12
False Negative (FN)	3	1	4	3
Sensitivity TP / (TP+FN)	72.73%	90.91%	63.64%	72.73%
Specificity TN / (TN+FP)	92.86%	78.57%	71.43%	85.71%

Evaluation

In evaluating the results it was necessary review the experiment and any threats to its validity. To ensure fairness, the model results were analysed to identify any bias particularly in terms of the participant gender or age and no bias was found. To ensure the explainability of model, it was be able to understand how a prediction was calculated. Naïve bayes is a probabilistic model, so can be easily inspected by reviewing the probability tables for each of the five variables. To ensure the reliability of the class variable, as a measure of frustration, the questionnaire scores were assessed using Cronbach's Alpha and with a result of 0.79, this is deemed acceptable for this type of research.
But some threats exist in this pilot research, chief among them is the small sample size of 25, which could affect both the attribute and model selection. Due to Covid, all the participants in this pilot were known to the researcher, which again could introduce some selection bias. The Naïve Bayes model may also have over-fitted the data and with more data perhaps its performance may differ. Finally, although the results are promising, the tasks required the participant to interact with the in a particular way. The results therefore may not be generally applicability to other VR scenarios.

Conclusions and Future Work

The telemetry data produced by the virtual reality headset and hand controllers, shows promise in predicting (90.91%) whether the user will become frustrated. This research identified that user actions like grab, failed grab, object slide, head swivel and hand jerk are the predictors for a Naïve Bayes classifier. Conversely, the users gaze was not found to be a useful predictor.
This research contributes to both the user experience and virtual reality fields, showing the possibility of extending the current laboratory-based user research into the wild. It also creates the possibility of future software that can predict user frustration and dynamically adjust or intervenes before the user becomes frustrated and abandons the application and exits the immersion.
Future work should aim to increase the sample size ideally by embedding within actual production VR software. The scenarios should also be broadened to understand how applicable the model is as well as establishing the ground truth for each user and scenario as different attributes and models may be required depending on the circumstance.

Rust WASM vs JavaScript

Everton Santos
Department of Computing, TU Dublin, Tallaght, Ireland
X00169671@myTUDublin.ie

Introduction

The Web platform's evolution has given rise to sophisticated and demanding Web applications such as numerical calculations, audio/video processing and games. Currently, JavaScript (JS) is a dominant technology solution to handle all site interactivity. Consequently, JavaScript is under enormous pressure to provide the demanded web site low response times. To address the response time demand, an association of the major browsers created WebAssembly (WASM or WA). A standardised and platform-independent bytecode designed for the web. The research objective is to compare and contrast the performance of multiple algorithms written in JavaScript and Rust WebAssembly versions in terms of Execution Time and Memory Consumption. The experiment takes place inside the browsers Chrome, Edge, and Firefox. Furthermore, the Research Methodology section covers test design, test procedure and data collection. Finally, the Discussion section summarises the results of the experiment. Rust WebAssembly is faster in terms of execution time. However, it requires more memory than JavaScript.

Tools

The JavaScript algorithm versions utilises ECMAScript 6 as the standard, while Rust is the selected language to generate the WebAssembly algorithm versions.

Selected Algorithms

The study selected four algorithms which were divided into two categories: searching/sorting and computation algorithms. The searching/sorting category includes the algorithms: Binary Search and Merge Sort, while the computation category contains the algorithms: Sieve of Eratosthenes and Fibonacci. Low performance in these algorithm categories can affect website user's experience.

Results

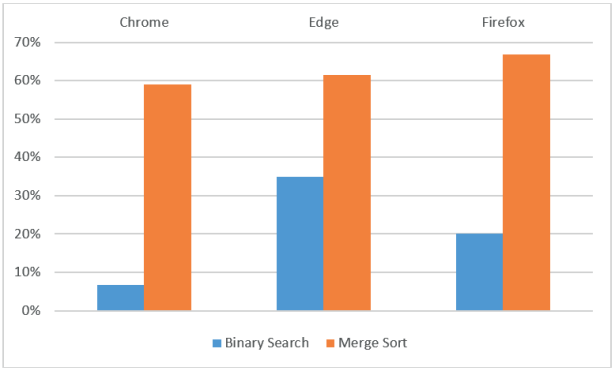
1. Mean Memory Consumption of WebAssembly over JavaScript:

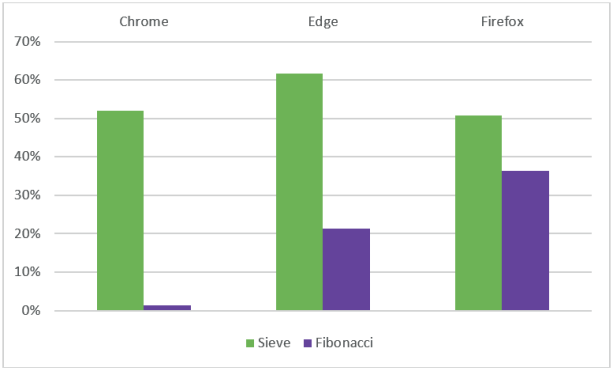
	Searching/Sorting Cat.		Computation Category	
	Binary Search	Merge Sort	Sieve	Fibonacci
Chrome	39.11%	51.47%	37.87%	-3.39%
Edge	30.19%	52.01%	35.89%	-7.45%

2. Mean Execution Time of WebAssembly over JavaScript:

	Searching/Sorting Cat.		Computation Category	
	Binary Search	Merge Sort	Sieve	Fibonacci
Chrome	6.67%	58.95%	52.03%	1.28%
Edge	35.00%	61.50%	61.60%	21.33%
Firefox	20.00%	66.77%	50.68%	36.32%

Mean Execution Time of WebAssembly over JavaScript





Conclusions and Future Work

The Rust WASM algorithm versions presented a better performance in terms of Execution Time than the JS versions. However, the WASM versions required more memory. Therefore, choosing what language to use will depend on the balance between performance gain and memory use for the algorithm in question.

In this study, multiple paths were identified for possible future work. Some paths were identified by the results of this work, while others by time or resources constraints. The test execution and data collection were applied over a unique computer architecture (AMD laptop with Windows 10 OS). Other computer architectures could be used. A Mac computer is a good candidate that would enable test execution on the Safari browser. Finally, other algorithm versions could be instrumented as well as other languages that compile to WebAssembly, such as C/C++, GO and Swift.

An investigation into the factors affecting software defects and security vulnerabilities through the code review process

Brendan Lally, supervised by Dr John Burns
Department of Computing, TU Dublin, Tallaght, Ireland
X00169686@myTUDublin.ie

Introduction

Code reviews from the chromium family of projects were studied. Previous work in this field had identified a higher rate of defects in code committed on Friday than on other days of the week, but this was with a project that wasn't using a modern code review process. Other research looking at the chromium projects had identified a higher rate of missed defects in reviews by experienced reviewers; this raises the question of whether fatigue may explain that effect. Furthermore, it was identified that substantially more defects were found in bug-fixes. Was that because the review process was better, or the changes were of a lower quality?

Research questions

RQ1: Is there sufficient evidence to show that the likelihood that a pull request is approved which introduces a defect or which introduces a security-related defect is greater on any particular day of the week?
RQ2: For those reviewers with a sufficiently large number of total reviews, is there a correlation between the proportion of reviews which are approved and which turn out to be buggy, and the number of other reviews which have been conducted by that same individual in the previous 12 hours?
RQ3: Of those software defects which were introduced to the system, is there evidence to show that bug-fix pull requests are disproportionately responsible for the introduction of those defects.

Methodology

Defect reports were categorised and the changes which fixed those defects were identified. Using SZZ Unleashed, the changes which introduced the code that needed to be fixed was identified. These changes were then referenced to the code reviews to identify those reviews which had allowed defects to be introduced and those which had not.

Results

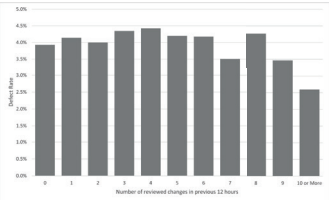
1. Day of week:

It was identified that the defect rate for changes changed over the lifetime of the project, to account for this and to consider the within-week variation by day, an ANOVA test of the defect rate on each day was used. The result of the test was not significant

$F_{Crit} = 2.3805$
F = 0.9109
p = 0.4568

2. Recent Reviews:

The correlation between the number of reviews conducted in the last 12 hours and the defect rate is -0.590. p=0.056 which is not statistically significant. Furthermore the correlation is actually in the opposite direction to that which would be expected by the hypothesis, since it was anticipated that more recent reviews would imply more defects passing through, not fewer.



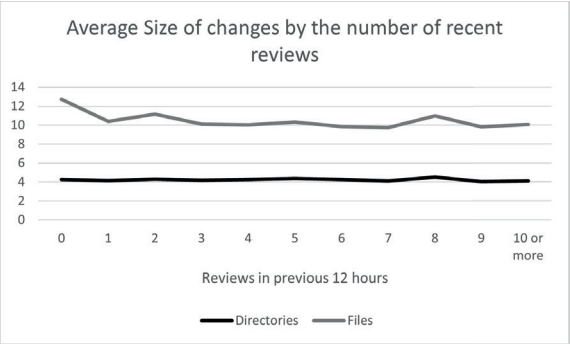
3. Bug Fixes:

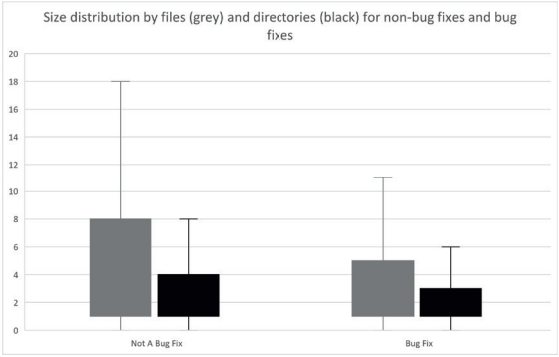
The contingency table for changes that were and were not defective for bug fix and non-bug fix changes is presented. This has a chi-squared statistic of 832.9 which has a p value of 3.9×10^{-183} which is clearly statistically significant. The odds ratio is 2.09. When looking at security-related defects only, the odds ratio is 2.41

Observed Values	Is not known to be defective	Was Defective	Total
Was not a bug-fix	550571	20395	570966
Was a bug-fix	22233	1718	23951
Total	572804	22113	594917

Effect of the size of changes

For the effects that were observed, it is necessary to consider the sizes of the changes being reviewed. Below are the average size of changes by recent reviews and a box plot for the size range for non-bug fixes and bug fixes. Correlation for directories and files to change counts are -0.099 and -0.590 respectively. Neither are significant and the file counts don't correlate to defect rates (0.176, p=0.604). The smaller sizes of big fix changes compared to non bug fix changes means that not only do bug fixes introduce more defects, but they do so in a smaller space.





Conclusions and Future Work

No evidence was found to suggest that the day of the week affects the rate at which defects enter the chromium project. This might reflect on the quality and consistency of the software development practices being utilised. This study provides no support for the hypothesis that fatigue is the cause of variation in efficacy for code reviews. Indeed the correlation is in the other direction (although below the level of significance). To suggest a fatigue-related effect for code reviewers, another mechanism would need to be proposed, such as preferential selection of easier reviews when the reviewer is tired. An analysis that incorporated complexity measures for the code that was studied. An analysis of the social relationships of practitioners to each other may also be interesting to pursue, bug fixes were shown to have more defects, despite being smaller changes, but when there are multiple reviewers for a change does the degree of scrutiny that the change is subjected to affected by the reputations of the submitter or other reviewers?

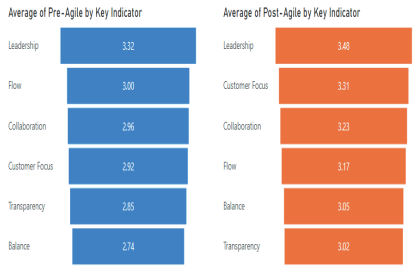
Evaluation of the implementation of the Agile Methodology for IT Ops in a Managed Services Org

Shane O' Donovan
Department of Computing, TU Dublin, Tallaght, Ireland
x00169653@mytudublin.ie

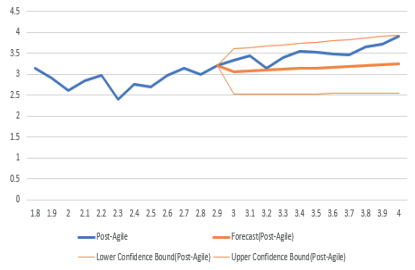
Introduction

As part of a move towards a more DevOps model, Agile is the transformational framework of choice. The growth of IaC and X-as-a-Service blurring the lines between software and service, an area where Managed Service businesses are looking to capitalize.Physical product is now being delivered as a service, with the service now becoming the product.Based on core principles of Agile, which closely align to operational excellence within in a traditional Managed Services, I examined the impact an Agile Transformation has on six of these values.It is through the lens of these 6 values that the wider cultural impact of Agile is seen, outside of standard Agile metrics of Velocity, Lead time and Burndown charts, etc. Focus on the social, cultural and governance elements facilitates measurement of the impact of moving from top-down management of large, siloed teams to small autonomous cross functional units that focus on collaboration over competition.This research will examine how a large Managed Services organization has adopted and scaled Agile across its org, while measuring the impact it's having across the business. To support this, a Pre & Post survey was carried out across 3 customer accounts and the effect of Agile measured in terms of Customer Focus, Transparency, Collaboration, Balance, Flow and Leadership.

Pre & Post Indicator Comparison



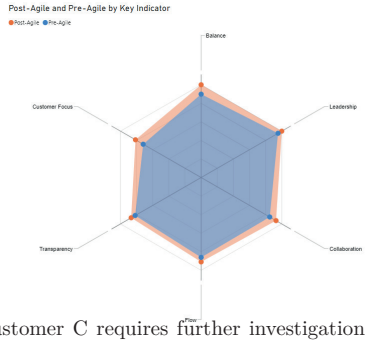
Agility Forecast



Research Questions

RQ 1: Does the implementation of Agile positively or negatively impact delivery of IT operations in a Managed Services business across the following six areas; Customer Focus, Transparency, Collaboration, Balance, Flow and Leadership?

Based on the below radar graph, a cumulative increase can be seen across the board following the 3-month Agile transformation.



RQ 2: Do patterns exist in the measurement results captured in the implementation of the methodology?

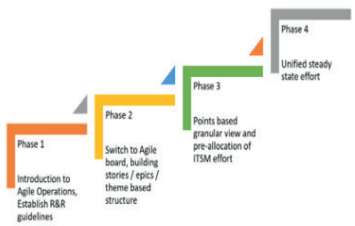
Increases across the board overall, no one category standing out. Balance, Customer Focus and Collaboration show a slightly greater increase than Flow, Leadership and Transparency. There is a noticeable peak across all 3 customers in Leadership, both Pre and Post Agile. Customers A & B displayed a marked improvement across the board, going from 'Limited' to 'Good'. However, Customer C requires further investigation based on findings of an effort by the team to manipulate results, conspiring to give a higher than true reflection, due to pressure felt amongst the team.

RQ 3: What conclusions can be drawn from these patterns, if any?

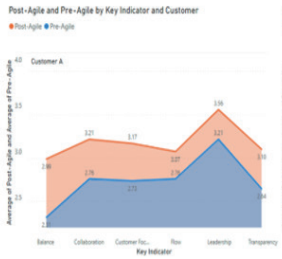
Findings show Leadership is the highest scoring category, both pre and post transformation. From this it can be inferred it has the largest part to play in the over transformation. Collaboration and Customer Focus display a marked increase at the account level Post Transformation. Based variance, Pre & Post transformation, there's both an overall increase in agility and a narrowing in the range, indicating increased consensus, a key indicator in high performing teams.

Overview

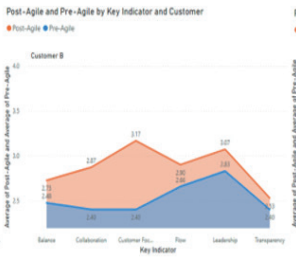
As part of a move towards a more a DevOps way of working an Agile Transformation program kicked off in early 2021. A phased roll out of agile across 3 pilot account over a 3 month period lay the foundation for Agile-at-Scale. A Pre and Post Agile survey was used to capture the leveles of agility across the teams, measuring impact in terms of Customer Focus, Transparancy, collaboration, Balance, Flow and Leadership.



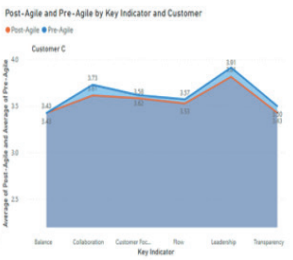
Agile Transformation



Customer A



Customer B



Customer C

Conclusion

In conclusion a positive effect is seen implementing Agile across IT Operations in a Managed Services Business in the areas of Customer Focus, Transparency, Collaboration, Balance, Flow and Leadership. With consistently strong Leadership results Pre & post transformation, it's evident this plays a large part in both organizational culture and the transformation. Further investigation is needed into the effect Agile Leadership has on a transformation. As well as it's relation to strategic flexibility and how it influences a transformation.

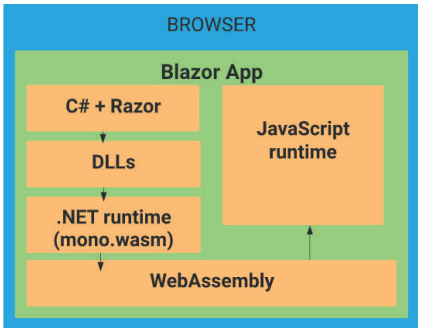
Investigation of the performance of a Blazor Client Side Web Application versus JavaScript

Ann Marie Sexton
Department of Computing, TU Dublin, Tallaght, Ireland
X00169647@myTUDublin.ie

Introduction

This work investigates the use of Web Assembly (WASM) and Blazor (C#) technologies as an alternative to traditional JavaScript for building web applications, and the impact on performance of the resulting Single Page Application (SPA) across different browsers. As relatively new technology since being released in May 2020, Blazor makes it possible to create a full-blown web application using C# and WebAssembly. Any computation can be done on the browser side with the aim being a very quick response to the user with no latency. Blazor uses razor files and the name Blazor comes from the combination of the words Browser and Razor. It uses the Razor syntax which is a mix of C# and HTML. At a high-level, the C# and Razor code runs on the .NET framework in the browser, creating DLLs which are compiled to the low-level binary WebAssembly format which is loaded by the JavaScript runtime, and runs in the same sandbox as JavaScript, to render the page and allow the application to access the same APIs, Web Sockets and the DOM (Document Object Model), as JavaScript.

Architecture



Recommendations

From the results of the experiments and taking into consideration the results of previous research, the recommendation to web application developers would be to handpick the computational heavy transactions for use with WebAssembly and continue to use JavaScript for everything else for the time being. No doubt this recommendation will change in the future as we come to think about WebAssembly as both a revolution and an evolution.

Research Questions

RQ 1: What conclusions have previous studies been able to draw from comparisons between WebAssembly and JavaScript?

Previous studies have shown that WebAssembly is faster than JavaScript by a multiple of times. This was consistent across all previous research that was studied, however it was clear that a number of factors influenced the magnitude of the performance gain, from the number of lines of code in the benchmark tests to the browser it was tested on.

RQ 2: How does WASM/Blazor compare to JavaScript across the three widely used browsers: Google Chrome, Edge, and Firefox?

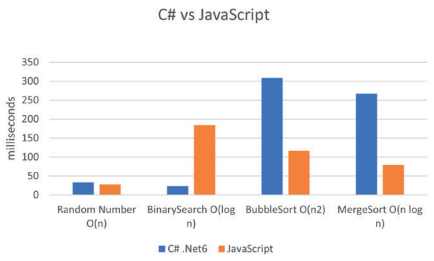
The WASM and Blazor combination did not perform as well as expected against JavaScript on any of the browsers when running either of the sorting algorithm, and this disproves the original hypothesis.

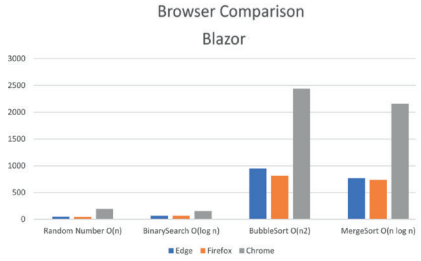
RQ 3: How does WASM/Blazor compare to JavaScript when tested across multiple versions of .NET Framework?

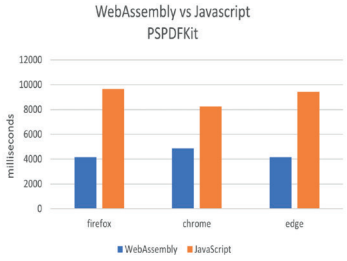
The results for Blazor/WebAssembly running on Firefox were better than Chrome or Edge, when tested on .NET 6 and using Ahead of Time (AoT) compilation, versus the same tests carried out on .NET 5. This was as expected, however it was originally hypothesised that the overall performance of the Blazor/WebAssembly combination would be significantly better than JavaScript which was proved to be false.

Overview

There were two separate sets of benchmarking tests carried out to compare performance of Blazor/WebAssembly versus JavaScript. For the first set of tests, a simple Blazor/Webassembly client side web application was built which presented buttons in the UI to execute functions of increasing complexity. These functions were written in both C# and JavaScript and the code was written to be as similar to each other as possible in terms of complexity and lines of code. To introduce, and be able to gauge, complexity in the code, a number of algorithms were identified and used, such as Linear Search O(n), Binary Search O(log n), Bubble Sort O(2) and Merge Sort O(n log n). The second set of tests were carried out using the PSPDFKit benchmarking tests which were run on localhost, to eliminate any latency overhead.







Conclusion

The desire to achieve near native speed and performance is a very valid pursuit as this is the holy grail when it comes to web application development and this is one of the biggest contributors to what will determine if WebAssembly will achieve the status it appears to be heading for or whether it will end up in the virtual scrapheap with Flash, Silverlight, Active X, Java Applets, and all of the other client side development efforts that almost made it but just fell short.

An investigation on the use of cloud-based GIS software

Daniel da Silva
Department of Computing, TU Dublin, Tallaght, Ireland
X00169678@myTUDublin.ie

Introduction

The construction of public infrastructure is highly dependent on information captured during the project planning phases, including localised surveys, information related to the location of existing utilities, health as safety assessments, and others. Access to up-to-date information in the field can be a strategic business advantage to improve quality and control over construction work. However, the access and updates of geographically spread information are not without challenges, and selecting the correct software to allow accessing information from multiple locations and from mobile devices is essential. This research compares and contrasts different software products that offer a platform for manipulating geographic information with a focus on comparing the traditional desktop-based applications with the available Geographic Information System (GIS) platforms based on the cloud environment as SaaS (Software as a Service). To do so, this paper investigates the market leaders of the segment of Geographic Information Systems and the challenges encountered by users of these platforms of using the cloud-based counterparts, or compatible companion, for these software tools. This study also investigates the role of GIS mobile applications as tools in the use of GIS platforms

Hands-On Testing

Based on performance, the desktop tools presented a clear advantage compared to the cloud-based GIS software tested in this research. The most critical difference noted during the tests was related to the speed and flexibility of the desktop tools concerning specific GIS analysis transformation tasks.

Based on the usability comparison, the web-based tools presented an advantage when compared directly to their desktop counterpart. However, when extending the comparison to all tools tested, both ESRI products, ArcGIS Online and ArcGIS Pro, presented a better overall score, influenced mainly by the data compatibility due to their position as the market leader, since their data formats are predominant in the domain covered by this research.

Between the two web-based tools available, ArcGIS Online demonstrated a better performance in the tests, primarily based on the most stabilised and broader offering of apps to perform specific functions around the data in the platform.

Survey and Interviews

Regarding the use of web-based tools, the survey revealed that among the users of ArcGIS Pro, 82% have already worked with ArcGIS Online, while 67% of the users of QGIS have used a web-based tool alongside QGIS.

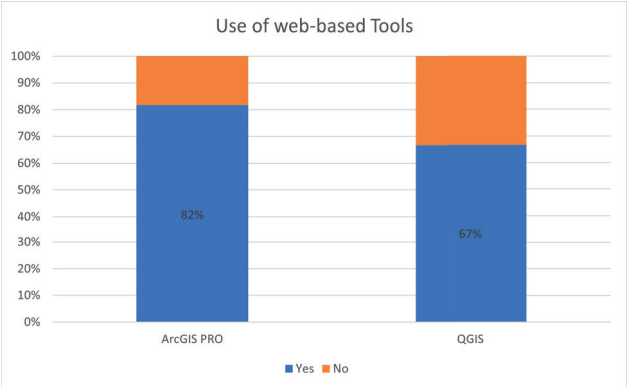
When questioned if they were able to perform the same tasks on the web-based GIS tools as they do on the desktop application, 50% of the users of QGIS that answered the question said they were be able to use the cloud platform for all their tasks. The positive answer was from 67% among ArcGIS users.

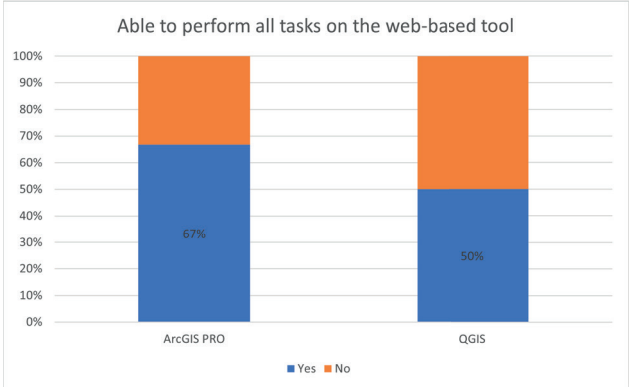
When questioned about the level of functionalities offered by the desktop applications and the cloud counterparts, all three specialists agreed that the limitation of functionalities and performance on the online applications set limits on what can be achieved at the moment using strictly SaaS GIS software.

The mobile usability comparison identified the most outstanding contribution of the cloud-based GIS tools, besides the data management, in the proposed context of the construction of public utilities that is the flexibility to access and manipulate data using mobile devices such as smartphones and tablets. As detailed by the survey and interviews, mobile applications have been used to provide access to data on remote locations, such as drawings on construction sites and capture information, including tracking construction progress.

Although all specialists raised current technology limitations or other factors that limit the use of the current GIS technologies in the cloud, the use of mobile applications for data capture and visualisation was mentioned by everyone as a great advantage of cloud-based GIS software.

Results





Conclusions and Future Work

The discoveries made in this research leads to the conclusion that the cloud-based GIS tools, as it stands on the current market, are not a total replacement for the desktop software but a complementary tool that increases the usability and capabilities of the GIS software.

Future work possibilities in this area include a more exhaustive investigation of SaaS GIS tools and case study of their application on specific construction scenarios, as well as a comparison of the uses of GIS SaaS tools with GIS PaaS (Platform as a Service), such as ArcGIS Enterprise and open-source option as GeoServer.

ML Model as Serverless FaaS with MLOps

Muhammad Adnan
Department of Computing, TU Dublin, Tallaght, Ireland
x00169651@mytudublin.ie

Introduction

"ML Model as serverless function with MLOps, would it be applicable and beneficial?"

- The research report will comprehensively focus viability, performance, scalability and cost effectiveness.
- Aims to focus a novel methodology for research and hypothesis to provide the outcome.
- It will include practical work using ML Model as AWS lambda function and deploy into Kubeflow (MLOps).
- The outcomes will highlight viability running ML Model as FaaS and practical approach required for it.
- It will also highlight prospects for organisations to use MLOps as FaaS, and how it would be beneficial.

MLOps

The MLOps (Machine Learning Operation) is a combination of development processes, best practices and technologies, which provides a scalable and manageable centralised resources to automate and scale the deployment and management of trusted ML models and applications in production environment.

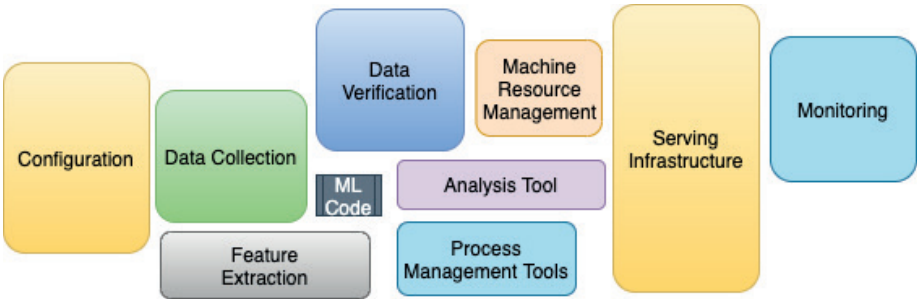
Serverless

The serverless is a cloud computing service model in which infrastructure orchestration is managed by the cloud providers. End users don't have worry about any underlying infrastructure (Servers, VM Storage, Security etc.

Function-as-a-Service (FaaS)

The "Function as a Service" is type of cloud computing service, with this users can run source code as cloud function and they don't have to worry about any underlying infrastructure. AWS Lambda, Azure function and Google cloud function.

Model



AWS Lambda Memory and vCPU Comparison

Provisioned Memory (MB)	Allocated Cores	vCPU Quota
1024	1	0.58
2048	2	1.15
3072	2	1.73
4096	3	2.31
5120	3	2.88
6144	4	3.46
7168	5	4.04
8192	5	4.61
9216	6	5.19
10240	6	5.77

Provisioned Memory (MB)	Allocated Cores
128 - 1769	1
1770 - 3538	2
3539 - 5307	3
5308 - 7076	4
7077 - 8845	5
8846 - 10240	6

Conclusions and Future Work

With server-full frameworks for machine learning models have large difficulties for many users, such as AI, data scientists, engineers, and developers, it is usually expensive if used infrequently or inefficiently. The proposed methodology and approach for assessing serverless AWS lambda FaaS with MLOps services (Kubeflow) would be the focus of this thesis paper. The purpose of this study is to conduct a realistic assessment of the practicality of good options. The fundamental purpose and objective of this research are to understand enterprises and organizations to consider and implement new cloud technology stacks for serverless ML FaaS and MLOps to decrease volatility, risk, and reduce execution costs while achieving optimum benefits.

NOTES