UNIVERSITY of LIMERICK

OLLSCOIL LUIMNIGH

Autumn Report

Docker containers deployed using Bluemix

Thomas Flynn

16117743

Sean McGrath

Information & Network Security MEng

2017

Electronic & Computer Engineering

Table of Contents

1 Introduction and report outline

TODO: Revise description considerably as it was written weeks ago.

1.1 Project description

This project aims to provide a bridge between the university's wireless sensor network (WSN) mobile application and the data mining opportunities of the sensor information obtained. This will comprise of a highly scalable database and a simple web application developed using Docker containers deployed on the cloud.

Docker containers are lightweight self contained Linux Operating systems. They will be used to handle the load balancing aspects of the WSN. The Global Position System coordinates transmitted from the WSN will be stored in a graph database for data mining purposes. The database will need to be a non relational database, allowing the database to scale more efficiently.

The mobile sensor IoT application at this point is still in early development by another student. The application will track the Global Positioning System coordinates of the user when they have stepped onto the university's campus. This project will allow for a potential survey feature to be implemented in the mobile application. This would provide great data mining capabilities when combined with other sensor data.

The web application will comprise of a simple user interface that will provide the ability to query the sensor data stored in the scalable database. The data will be used to display the current status of the sensors to the user . Graph databases will allow for additional database filtering features to be implemented on the websitenin the later stages of the development life cycle.

In summary this project aims to provide a bridge between WSN and data mining. The learning outcomes of this project will provide me with a foundational knowledge in the areas of security, development and operations in relation to designing highly scalable and secure cloud applications.

1.2 Project aims and objectives

TODO: Revise

Primary objectives

1.) Create an instance that will transmit pseudo sensor data.

2.) Create an instance that will read sensor data.

3.) Develop a database that can store the pseudo data in an organized manner.

4.) Develop a web application that will display information from the database.

5.) Develop a web server to handle web application database queries.

TODO: Revise

Secondary objectives

1.) Scale up number of instances.

2.) Test load balancing aspects of increased scale.

3.) End to end testing of application security.

TODO: Revise

Learning outcomes

1.) Discipline and minimizing procrastination.

2.) Strengthen programming skills.

2.) Development operations.

4.) Application security.

Project summary diagram

TODO: Very high level abstraction diagram

1.3 Report outline

TODO: Discuss with supervisor

Literature survey

Summary of the various aspects of the project such as Docker, Cloud container services vendors, database, and application security research.

Theory

Expands upon in detail the areas discussed in the literature survey.

High level design

Description and diagram of how the multiple project components interact with each other from a high level of abstraction.

Detailed action plan

Describes the project management strategy, critical path, and identified risks.

2 Literature survey

TODO: Put the mention of the New Stack in the introduction, or simply write it better here.

2.1 Outline of research

The New Stack podcast proved to be a vital source of information for this thesis. The podcast provides analysis and explanations about application development and management at scale, which is how I first heard about Docker containers.

This section is laid out as follows

- Docker survey
- Cloud container service vendors survey
- Container deployment and management services
- Cloud organizations and platforms.
- NoSQL database survey
- Graph database survey
- IBM Cloudant database survey
- Graph visualization survey
- Programming languages survey

2.1 Docker research

Docker containers

Docker containers wrap a piece of software in a complete file system that contains everything needed to run: code, runtime, system tools, and system libraries. **[7]**

Lightweight

Containers running on a single machine share the same operating system kernel, they start instantly and use less RAM. Images are constructed from layered filesystems and share common files, making disk usage and image downloads much more efficient. **[7]**

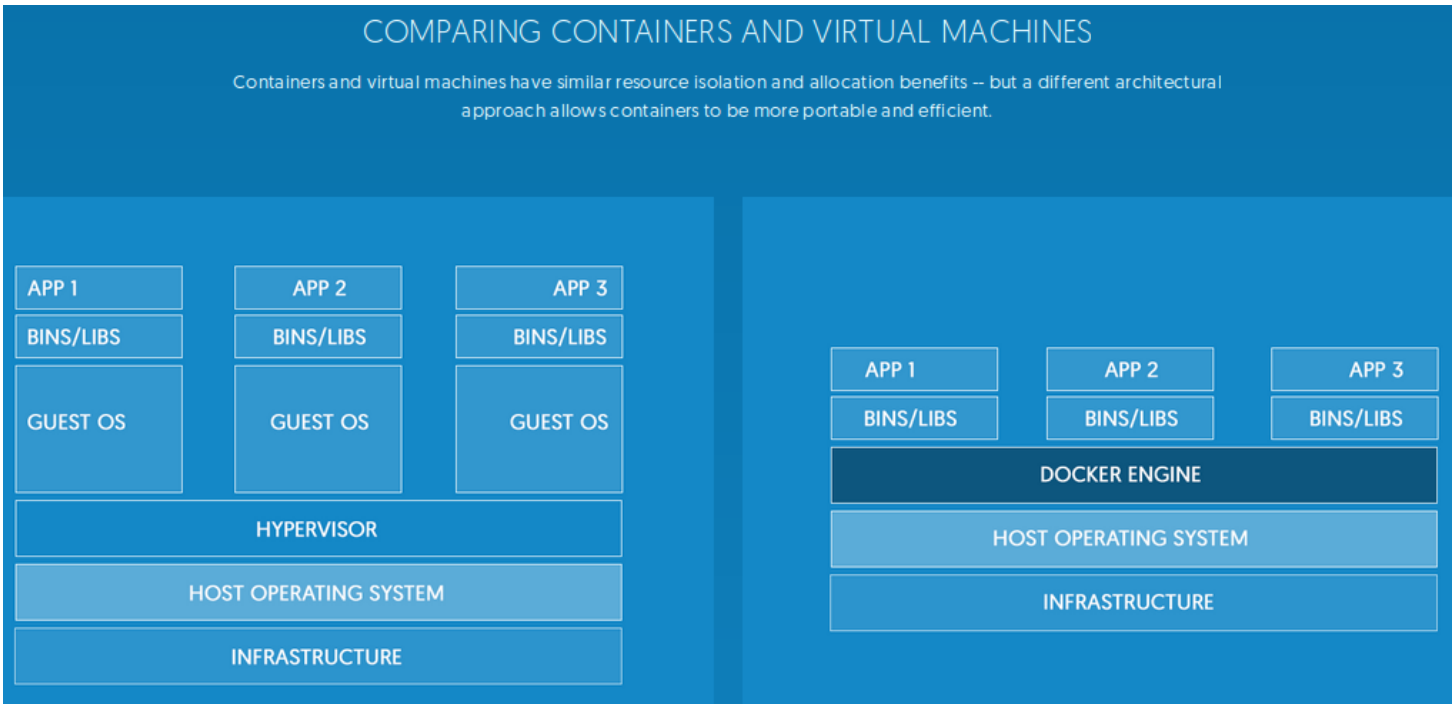
Open

Docker containers are based on open standards, enabling containers to run on all major Linux distributions and on Microsoft Windows -- and on top of any infrastructure. **[7]**

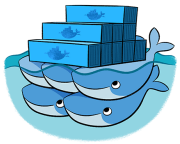
Secure by default

Containers isolate applications from one another and the underlying infrastructure, while providing an added layer of protection for the application. **[7]**

Comparing container and virtual machines



Docker Swarm



Docker Swarm provides native clustering capabilities to turn a group of Docker engines into a single, virtual Docker Engine. With these pooled resources, you can scale out your application as if it were running on a single, huge computer. **TODO:** ref

Compatible with Docker Tools

Docker Swarm serves the standard Docker API, so any tool which already communicates with a Docker daemon can use Docker Swarm to transparently scale to multiple hosts: Dokku, Docker Compose, Krane, Flynn, Deis, DockerUI, Shipyard, Drone, Jenkins and, of course, the Docker client itself. Includes ability to pull from private repos within DTR or Hub as well. **TODO:** ref

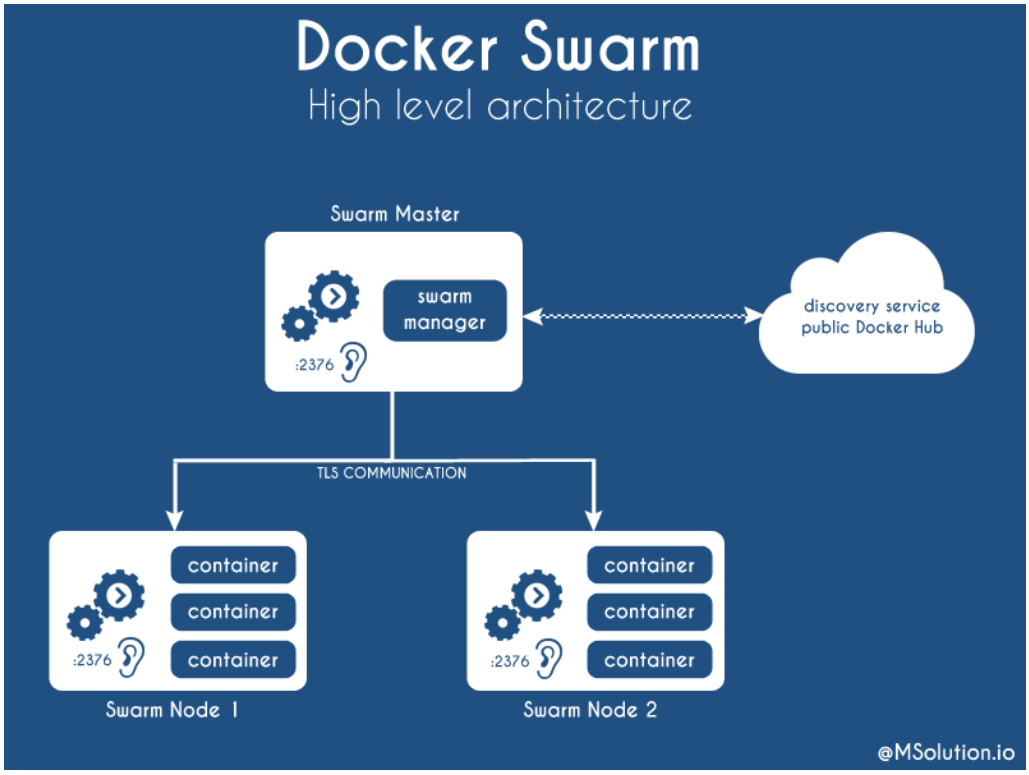
High Scalability and Performance

Swarm is production ready and tested to scale up to one thousand (1,000) nodes and fifty thousand (50,000) containers with no performance degradation in spinning up incremental containers onto the node cluster. **TODO:** ref

Integrated Networking and Volumes

As a Docker native solution, you can use Docker Networking, Volumes and plugins through their respective Docker commands via Swarm.

ode fails. Also includes error alerts when a node fails to join a cluster. **TODO:** ref



2.3 Container Service Vendors

Docker for IBM Bluemix

IBM and Docker offer integrated container solutions that can meet the diverse needs of enterprises. Supporting the creation and deployment of multi-platform, multi-container workloads across hybrid infrastructures, IBM and Docker accelerate application delivery and enable application lifecycle management for Dockerized containers. **[1]**

Docker for Azure

An integrated, easy-to-deploy environment for building, assembling, and shipping applications on Microsoft Azure, Docker for Azure is a native Azure application optimized to take optimal advantage of the underlying Azure IaaS services while giving you a modern Docker platform that you can use to deploy portable apps. Docker for Azure installs a Swarm of Docker Engines secured end to end with TLS by default, and is integrated with Azure VM Scale Sets for autoscaling, Azure Load Balancer and Azure Storage. **[3]**

Docker For AWS

An integrated, easy-to-deploy environment for building, assembling, and shipping applications on AWS, Docker for AWS is a native AWS application optimized to take optimal advantage of the underlying AWS IaaS services while giving you a modern Docker platform that you can use to deploy portable apps. Docker for AWS does not require any software installed. **[5]**

Docker for Google Cloud

Google Container Engine is a powerful cluster manager and orchestration system for running your Docker containers. Container Engine schedules your containers into the cluster and manages them automatically based on requirements you define (such as CPU and memory). It's built on the open source Kubernetes system, giving you the flexibility to take advantage of on-premises, hybrid, or public cloud infrastructure. **TODO:** ref

2.4 Container deployment and management vendors

Kubernetes

An open-source platform for automating deployment, scaling, and operations of application containers across clusters of hosts, providing container-centric infrastructure. **TODO:** ref

Kubernetes Features

- Deploy your applications quickly and predictably.
- Scale your applications on the fly.
- Seamlessly roll out new features.
- Optimize use of your hardware by using only the resources you need.

Apache Mesos

Mesos is built using the same principles as the Linux kernel, only at a different level of abstraction. The Mesos kernel runs on every machine and provides applications (e.g., Hadoop, Spark, Kafka, Elasticsearch) with API's for resource management and scheduling across entire datacenter and cloud environments. **TODO:** ref

Mantl

Mantl is an end-to-end solution for deploying and managing a microservices infrastructure. Mantl currently uses Apache Mesos as a cluster manager for microservices. **TODO:** ref

Mantl Features

- No vendor lock-in. Mantl runs equally well on any provider, saving you time and energy.
- Integrated tools like Cassandra, Spark & Hadoop.
- Service discovery, secret storage, load balancing, logging and more available right out-of-the-box.
- Run your services efficiently with multi-data center configuration and virtual networking tools.

TODO: ref

2.5 Cloud Organizations and platforms

Cloud Foundry

Cloud Foundry is an open source cloud platform as a service (PaaS) on which developers can build, deploy, run and scale applications on public and private cloud models. VMware originally created Cloud Foundry and it is now part of Pivotal Software. **TODO:** ref

Cloud Foundry is licensed under Apache 2.0 and supports Java, Node.js, Go, PHP, Python and Ruby. The open source PaaS is highly customizable, allowing developers to code in multiple languages and frameworks. This eliminates the potential for vendor lock-in, which is a common concern with PaaS. **TODO:** ref

Open Container Initiative

The Open Container Initiative (OCI) is a lightweight, open governance structure (project), formed under the auspices of the Linux Foundation, for the express purpose of creating open industry standards around container formats and runtime. The OCI was launched on June 22nd 2015. **TODO:** ref

Cloud Native Computing Foundation

The Cloud Native Computing Foundation (CNCF) is a nonprofit organization committed to advancing the development of cloud native technology and services by creating a new set of common container technologies informed by technical merit and end user value, and inspired by Internet-scale computing. As a shared industry effort, CNCF members represent container and cloud technologies, online services, IT services and end user organizations focused on promoting and advancing the state of cloud native computing for the enterprise. **TODO:** ref

OpenStack

OpenStack software controls large pools of compute, storage, and networking resources throughout a datacenter, managed through a dashboard or via the OpenStack API. OpenStack works with popular enterprise and open source technologies making it ideal for heterogeneous infrastructure. **TODO:** ref

2.5 NoSQL Database

A NoSQL database provides a mechanism for storage and retrieval of data which is modeled in means other than the tabular relations used in relational databases. **TODO:** ref

NoSQL features

1. The ability to horizontally scale “simple operation” throughput over many servers **TODO:** ref

2. The ability to replicate and to distribute (partition) data over many servers.

3. Simple call level interface or protocol (in contrast to SQL binding),

4. Efficient use of distributed indexes and RAM for data storage

5. The ability to dynamically add new attributes to data records.

NoSQL database types

Document store

Documents are addressed in the database via a unique *key* that represents that document. One of the other defining characteristics of a document-oriented database is that in addition to the key lookup performed by a key-value store, the database offers an API or query language that retrieves documents based on their contents.

Key value store

Data is represented as a collection of key-value pairs, such that each possible key appears at most once in the collection. The key-value model is one of the simplest non-trivial data models, and richer data models are often implemented as an extension of it. The key-value model can be extended to a discretely ordered model that maintains keys in lexicographic order.

Graph

This kind of database is designed for data whose relations are well represented as a graph consisting of elements interconnected with a finite number of relations between them. The type of data could be social relations, public transport links, road maps or network topologies.

2.6 Graph databases

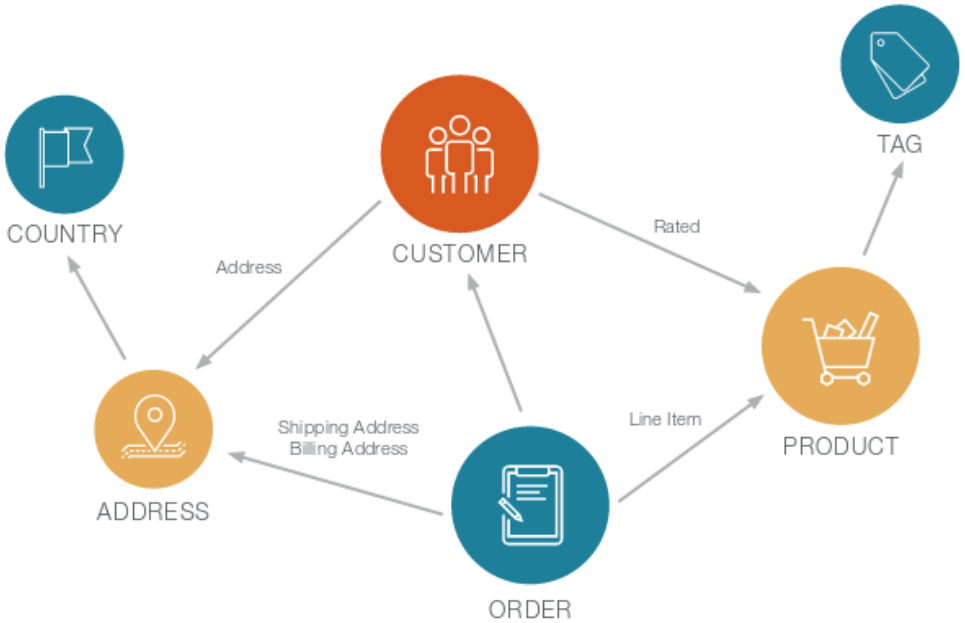
Graphs are the most efficient and intuitive way of working with data, mimicking the interconnectedness of ideas in the human mind. Neo4j is built from the ground up to harness the power of graphs for real-time, bottom-line insights. **TODO:** ref

Graph databases

Graph databases are based on graph theory. Graph databases employ nodes, edges and properties.

- **Nodes** represent entities such as people, businesses, accounts, or any other item you might want to keep track of. They are roughly the equivalent of the *record*, *relation* or *row* in a relational database.
- **Edges**, also known as *graphs* or *relationships*, are the lines that connect nodes to other nodes; they represent the relationship between them.
- **Properties** are pertinent information that relate to nodes.

Graph example figure



2.7 IBM Cloudant

IBM Cloudant is a managed NoSQL JSON database service built to ensure that the flow of data between an application and its database remains uninterrupted. Developers are then free to build more, grow more. **TODO:** ref

Global availability: Improve your app's performance

Cloudant's horizontal scaling architecture can handle millions of users and terabytes of data to grow seamlessly alongside your business. **TODO:** ref

Users are connected to the closest copy of the data, which reduces data access latency caused by cloud network overhead. **TODO:** ref

Data flexibility: JSON data store

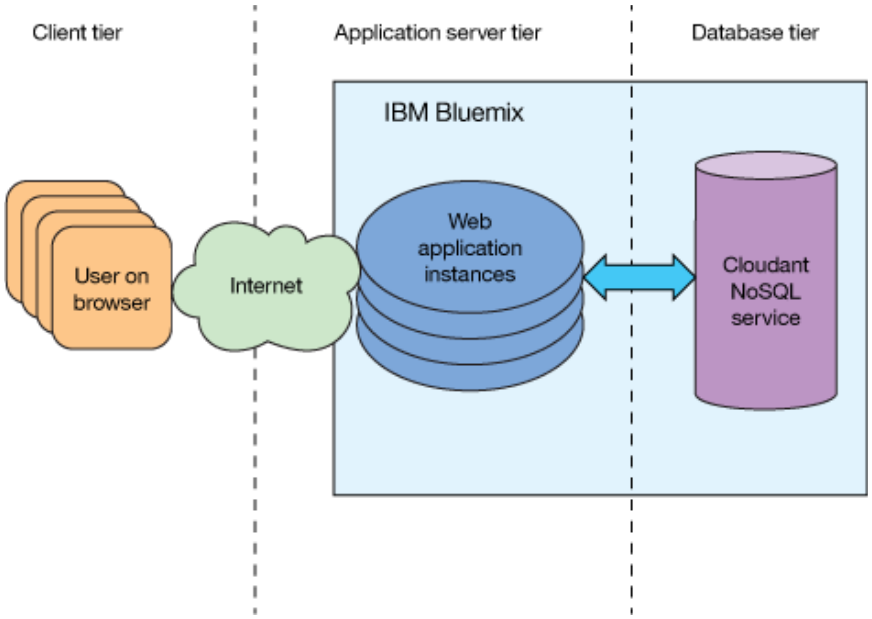
Cloudant's RESTful API makes every document in your database accessible as JSON. It is also compatible with Apache CouchDB™, enabling you to access an abundance of language libraries and tools. **TODO:** ref

Schema flexibility makes Cloudant an excellent fit for multi-structured data, unstructured data and fast-changing data models. **TODO:** ref

Advanced APIs: Integrated geospatial operations and search

Enhance web and mobile apps with geospatial operations that go beyond simple bounding boxes.

Seamlessly integrate web and mobile apps with fast, scalable, full-text Lucene indexing and search. **TODO:** ref



2.7 Graph visualization

Linkurious Graph Visualization

Linkurious uses Neo4j's graph database technology to offer you an easy solution to store, search and visualize graphs. You can start navigating inside your graph database. Search for properties, inspect nodes, and explore their relationships visually in your web browser. find any node in your database easily thanks to our built-in search engine. You can simply modify, add and remove nodes or relationships. You can also customize what is displayed, how the data is indexed. **TODO:** ref

Keylines Graph visualization

KeyLines is an out-of-the-box JavaScript solution for visualizing networks. KeyLines does the job of rendering data and responding to user interactions like clicking, touching, moving nodes, and more. These events are binded to customize what happens, and, most importantly, the data stays under control at all times: KeyLines is self-contained and needs no external connections. **TODO:** ref

Keylines example figure

2.8 Programming languages

NodeJS

As an asynchronous event driven JavaScript runtime, Node is designed to build scalable network applications. Upon each connection the callback is fired, but if there is no work to be done Node is sleeping. This is in contrast to today's more common concurrency model where OS threads are employed. Thread-based networking is relatively inefficient and very difficult to use. Furthermore, users of Node are free from worries of dead-locking the process, since there are no locks. Almost no function in Node directly performs I/O, so the process never blocks. Because nothing blocks, scalable systems are very reasonable to develop in Node. **TODO:** ref

Angular JS

AngularJS lets you write client-side web applications as if you had a smarter browser. It lets you use good old HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. It automatically synchronizes data from your UI (view) with your JavaScript objects (model) through 2-way data binding. To help you structure your application better and make it easy to test, AngularJS teaches the browser how to do dependency injection and inversion of control. **TODO:** ref

Java

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. **TODO:** ref

Perl

Perl 5 is a highly capable, feature-rich programming language with over 29 years of development. Perl 5 runs on over 100 platforms from portables to mainframes and is suitable for both rapid prototyping and large scale development projects. **TODO:** ref

PHP

What distinguishes PHP from something like client-side JavaScript is that the code is executed on the server, generating HTML which is then sent to the client. The client would receive the results of running that script, but would not know what the underlying code was. You can even configure your web server to process all your HTML files with PHP, and then there's really no way that users can tell what you have up your sleeve. **TODO:** ref

3 Theory

3.1 Outline of section

TODO: Explain layout of section and the journey of research.

TODO: Find out exactly what technologies I'll be using and make it obvious in this section.

3.x IoT Platform

IBM Watson

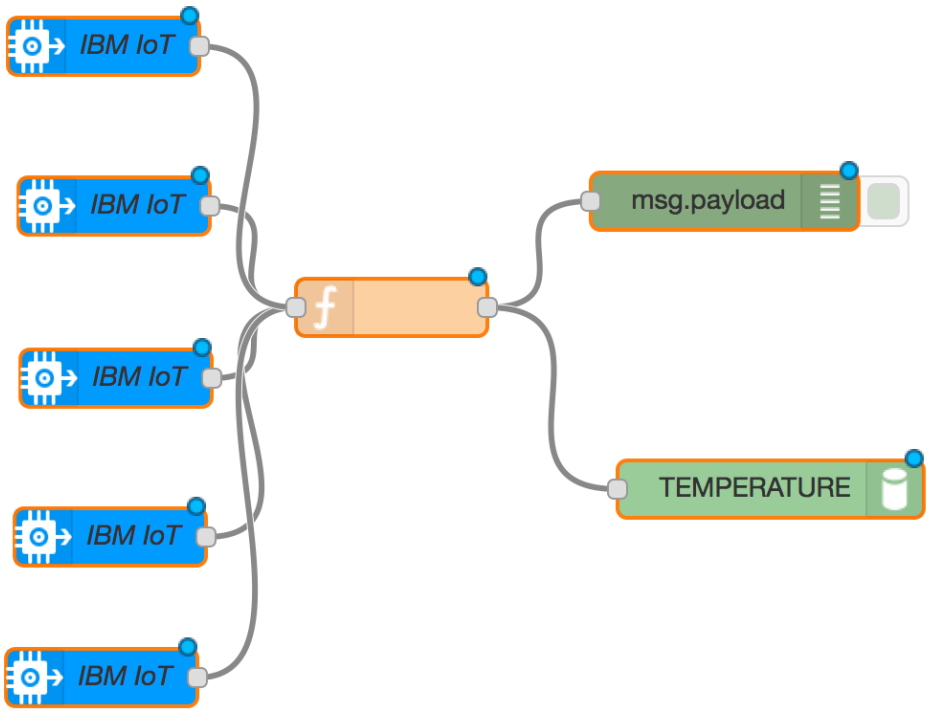
IBM Watson is a technology platform that uses natural language processing and machine learning to reveal insights from large amounts of unstructured data. **TODO:** ref

Node Red

Node-RED is a tool for wiring together the Internet of Things in new and interesting ways, including hardware devices, APIs, and online services. It is built on top of Node.js and takes advantage of the huge node module ecosystem to provide a tool that is capable of integrating many different systems. Its lightweight nature makes it ideal to run at the edge of the network. **TODO:** ref

Node-RED contains the following Watson IoT nodes that helps you to connect your devices, gateways and applications to Watson IoT Platform and create IoT solutions quickly.

- Watson IoT Node – A pair of nodes for connecting your device or gateway to the IBM Watson Internet of Things Platform. A device or gateway can use these nodes to send events and receive commands from the application.
- IBM IoT App Node – A pair of nodes for connecting your application to Watson IoT Platform. An application can use these nodes to receive device events and send commands back to the device.



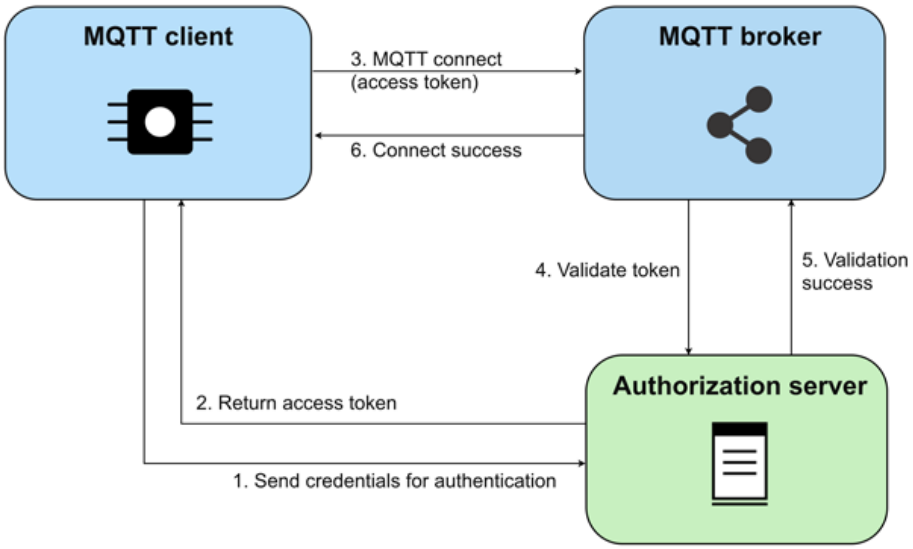
3.x IBM Websphere

IBM® WebSphere® MQ Telemetry provides real-time access to a range of mobile devices, remote sensors, actuators and other telemetry devices leveraging WebSphere MQ. It uses the MQ Telemetry Transport (MQTT) protocol to transport robust messages to even the smallest devices for near instantaneous data exchange. **TODO:** ref

MQTT Protocol

MQTT stands for MQ Telemetry Transport. It is a publish/subscribe, extremely simple and lightweight messaging protocol, designed for constrained devices and low-bandwidth, high-latency or unreliable networks. The design principles are to minimise network bandwidth and device resource requirements whilst also attempting to ensure reliability and some degree of assurance of delivery. These principles also turn out to make the protocol ideal of the emerging “machine-to-machine” (M2M) or “Internet of Things” world of connected devices, and for mobile applications where bandwidth and battery power are at a premium. **TODO:** ref

- TCP/IP port 1883 is reserved with IANA for use with MQTT.
- TCP/IP port 8883 is egistered, for using MQTT over SSL.
- Can pass a user name and password with an MQTT packet in V3.1 of the protocol. Encryption across the network can be handled with SSL, independently of the MQTT protocol itself

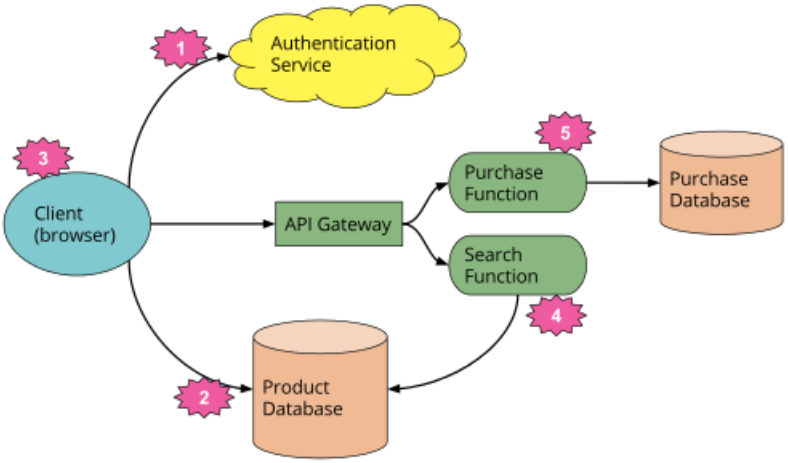


2.x Serverless Architecture

Serverless architectures refer to applications that significantly depend on third-party services (known as Backend as a Service or "BaaS") or on custom code that's run in ephemeral containers (Function as a Service or "FaaS") **TODO:** ref

Server model

Serverless model



TODO: ref

- The authentication logic in the server model has been replaced with a third party BaaS service.
- Allowed the client direct access to a subset of the database (for product listings).
- Some logic that was in the Pet Store server is now within the client, e.g. keeping track of a user session, understanding the UX structure of the application (e.g. page navigation), reading from a database and translating that into a usable view, etc.
- Some UX related functionality will need to be kept in the server, e.g. if it's compute intensive or requires access to significant amounts of data, then a FaaS function that responds to http requests via an API Gateway can be implemented.
- Can now replace the 'purchase' functionality with another FaaS function, choosing to keep it on the the server-side for security reasons, rather than re-implement it in the client.

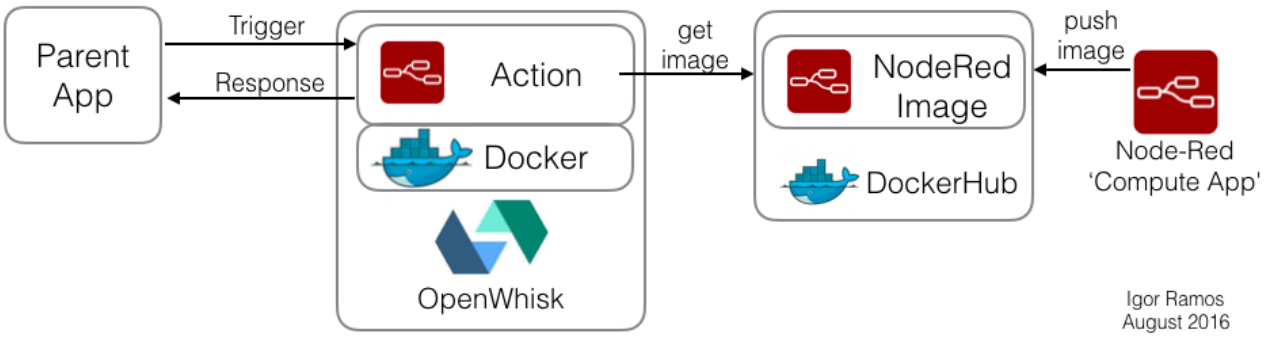
IBM OpenWhisk

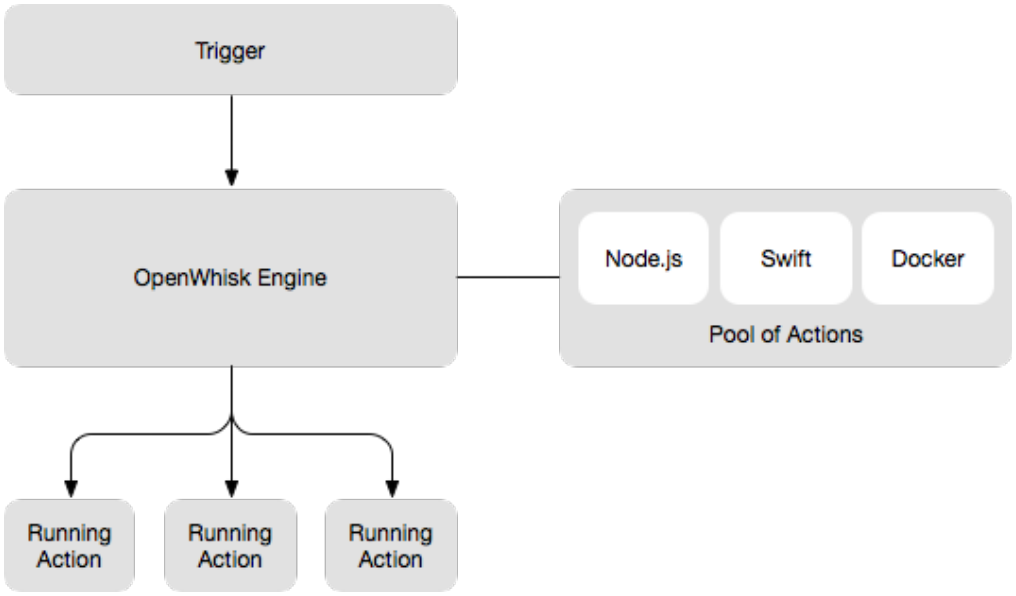
The OpenWhisk serverless architecture accelerates development as a set of small, distinct, and independent actions. By abstracting away infrastructure, OpenWhisk frees members of small teams to rapidly work on different pieces of code simultaneously, keeping the overall focus on creating user experiences customers want. **TODO:** ref

key architectural concepts:

- **Triggers:** A class of events emitted by event sources.
- **Actions:** Encapsulate the actual code to be executed which support multiple language bindings including NodeJS, Swift and arbitrary binary programs encapsulated in Docker Containers. Actions invoke any part of an open ecosystem including existing Bluemix services for analytics, data, cognitive, or any other 3rd party service.
- **Rules:** An association between a trigger and an action

  .
- **Packages:** Describe external services in a uniform manner.





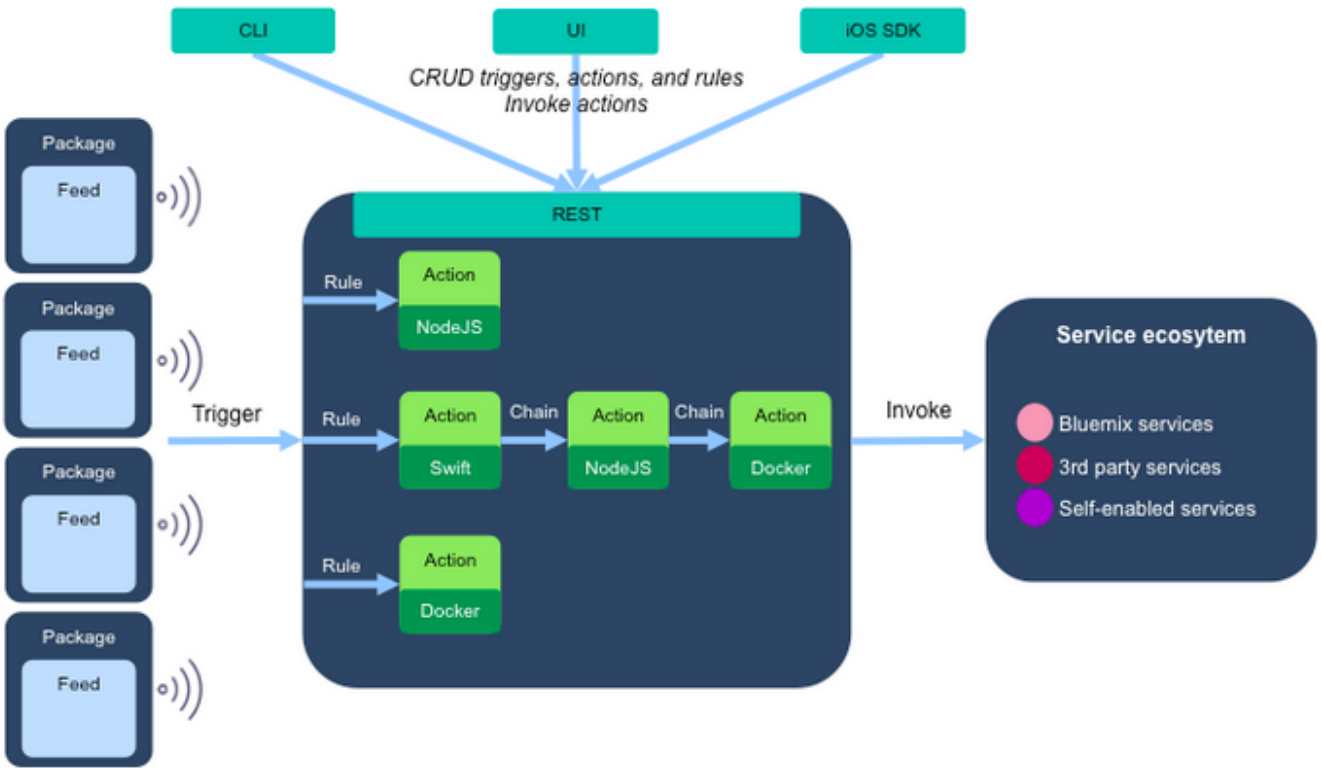
2.9 REST API

REST (REpresentational State Transfer) is an architectural style, and an approach to communications that is often used in the development of Web services. The use of REST is often preferred over the more heavyweight SOAP (Simple Object Access Protocol) style because REST does not leverage as much bandwidth, which makes it a better fit for use over the Internet. The SOAP approach requires writing or using a provided server program (to serve data) and a client program (to request data). **TODO:** ref

REST'S decoupled architecture, and lighter weight communications between producer and consumer, make REST a popular building style for cloud-based APIs. **TODO:** ref

REST architecture involves reading a designated Web page that contains an XML file. The XML file describes and includes the desired content. Once dynamically defined, consumers may access the interface. **TODO:** ref

REST Architecture example diagram



3.x Database technologies

3.2.1 IBM Graph platform

IBM Graph is an easy-to-use, fully-managed graph database service for storing and querying data points, their connections, and properties. IBM Graph offers an Apache TinkerPop3 compatible API and plugs into your Bluemix application seamlessly. **TODO:** ref

Features

Highly Available

Architecture ensures the service is always up and your data is always accessible.

Scale Seamlessly

Start small and scale on-demand as data grows.

Managed 24x7

Stack is monitored, managed, and optimized by experts 365 days a year.

IBM Graph Standard Plan

Plan	Features	Pricing
Standard	• 500 MB of free data storage • 25,000 API calls free per month	• $15.00 USD/GB • $0.20 USD/1000 API Calls

Geospatial Analytics service

Expand the boundaries of your application. Leverage real-time geospatial analytics to track when devices enter, leave or hang out in defined regions. Powered by IBM Streaming Analytics on Bluemix. **TODO:** ref

Features

- Monitor device locations in real-time.
- Connect to data sources that support the MQTT protocol and monitor devices as they move into geographic regions of interest.
- Control region monitoring using the geospatial API.

- Define geographic regions and control monitoring of regions using the geospatial application programming interface.

3.x IBM Bluemix DevOps Services

Connect with your GitHub repository

It's easy to link work items with GitHub code changes: use your tools to code, and manage your project with IBM® Bluemix® DevOps Services. You can reference a work item in GitHub comments before you push code changes, and the work item is updated with the new commit information. **TODO:** ref

Your work in one place

At a glance, see the projects that you own, belong to, and like. You can also see requests to join your project and invitations for you to join other projects. Just another way to help you stay organized.

Built-in source code management

Easily deliver code and seamless track changes from all developers. Each project gets a Git repository or a Jazz source code management repository and workspace where project members check in changes, associate code changes with work items, and view a history of recent updates.

Integrated source code editor

The Web IDE provides features to support these tasks:

- Import your source code.
- Copy a file by dragging it to a new directory.
- Edit code quickly. The editor includes several features:
    - Content assist for CSS, HTML, and JavaScript
    - Syntax highlighting for over 20 popular programming languages
    - Syntax highlighting, validation, and content assist for Cloud Foundry manifest files
    - Code validation
    - Keyboard shortcuts
    - Bracket and block-comment auto-completion
    - Key-bindings for vi and emacs users
    - Quick fixes to automatically solve common programming problems in JavaScript

3.x IBM Bluemix Security

IBM UrbanCode plug-in for application security testing

The IBM Application Security Testing for Bluemix plug-in enables you to run security scans on your web or Android apps that are hosted on Bluemix. This plug-in is developed and supported by the IBM UrbanCode™ Deploy Community on the IBM Bluemix DevOps Services platform. **TODO:** ref

Vulnerability advisor

Discover vulnerabilities and compliance policy problems in Docker images and learn new ways to improve images to meet best practices and upgrades to known industry fixes, regardless of the image source.

Integrated container monitoring

Log visibility and performance insight to the CPU, memory, and network utilization per container.

IBM Bluemix security features

The Bluemix platform secures data-in-transit by securing the end-user access to the application by using SSL through the network until the data reaches IBM DataPower Gateway at the boundary of the Bluemix internal network. IBM DataPower Gateway acts as a reverse proxy and provides SSL termination. From there to the application, IPSEC is used to secure the data as it travels from the IBM DataPower Gateway to the application. **TODO:** ref

Security for both data-in-use and data-at-rest is your responsibility as you develop your application. You can take advantage of several data-related services available in the Bluemix Catalog to help with these concerns. **TODO:** ref

Security of Bluemix applications

As an application developer, you must enable the security configurations, including application data protection, for your applications that run on Bluemix.

You can use security capabilities that are provided by several Bluemix services to secure your applications. All Bluemix services that are produced by IBM follow IBM secure engineering development practices. **TODO:** ref

3.x Load balancing and auto-scaling services

Load Impact service

Worlds #1 load testing tool - trusted by over 120,000 developers and testers. Unlimited testing, on-demand from multiple geographic locations. Create sophisticated tests using the simple GUI or connect directly to the platform via the API. **TODO:** ref

Load Impact service Standard Plan

- Max 100 concurrent users (VUs) per test
- 5 tests per month
- Max 5 minutes test duration
- Max 1 server metric agents

Auto-Scaling service

The Auto-Scaling for Bluemix service enables you to automatically increase or decrease the compute capacity of your application. The number of application instances are adjusted dynamically based on the Auto-Scaling policy you define. **TODO:** ref

Features

- Automatically add or remove resources to match the current workload.
- Define policy on metrics of interest.
- Visualize the current and historical values of performance metrics.
- Query the scaling activities based on status, time and type.

3.x Docker Container book

TODO: Read and Reference books

3.x Auto-scaling book

TODO: Read and Reference book

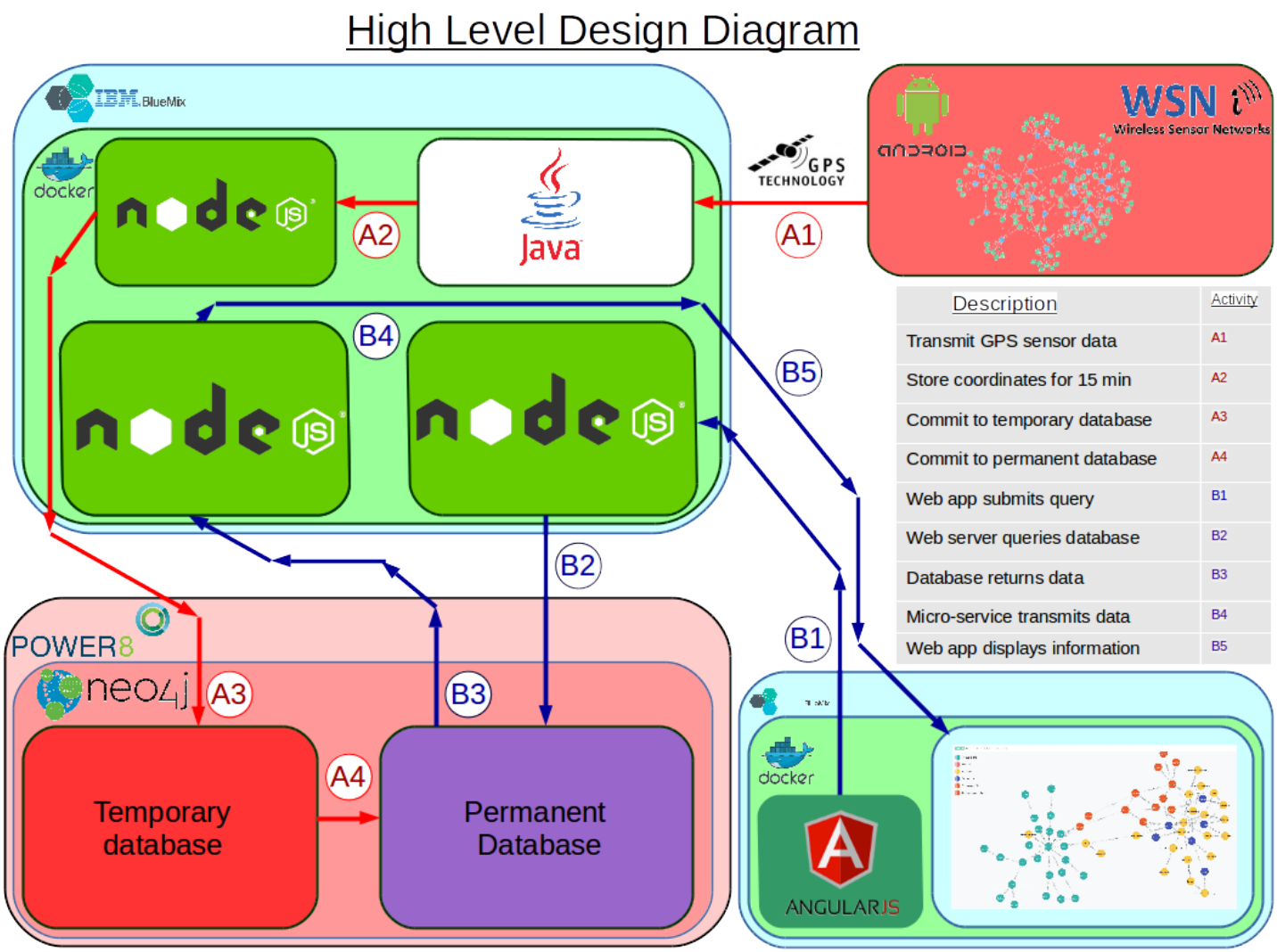
3.x Micro-services book

TODO: Read and Reference book

4 High level design

TODO: Attempted this far too early on and now have to revise it considerably.

4.1 High level design



TODO: UML diagrams of front and back-end , container orchestration diagrams, list of services diagram, IBM secure gateway connection diagram, serverless architecture diagram...

4.2 Software

4.3 Process related block 1

4.4 Process related block 2

4.5 Process related block 3

4.6 Process related block 4

5 Project

5.x Trello

Trello uses the kanban paradigm for managing projects, originally popularized by Toyota in the 1980s for supply chain management. Projects are represented by *boards*, which contain *lists* (corresponding to task lists). Lists contain *cards* (corresponding to tasks). Cards are supposed to progress from one list to the next (via drag-and-drop), for instance mirroring the flow of a feature from idea to implementation.

TODO: add reason for using Trello, description of labels and project board setup.

Project Board Labels

- Sdfs
- Sdf
- Sd
- Fs
- Sdf
- Sdf
- Sdf
- asda

5.x Toggl

Toggl is time tracking software that offers online time tracking and reporting services through their website along with mobile and desktop applications. Toggl tracks time based on tasks and projects, either through an interactive task timer or through manual entry.

TODO: Add reason for using Toggl, and add Autumn semester graphs in week 10

Autumn Bar Chart

Autumn Donut Chart

5.x Github

5.x.x About Github

GitHub is a web-based Git repository hosting service. It offers all of the distributed version control and source code management (SCM) functionality of Git as well as adding its own features. It provides access control and several collaboration features such as bug tracking, feature requests, task management, and wikis for every project.

GitHub offers both plans for private repositories, and free accounts which are commonly used to host open-source software projects. As of April 2016, GitHub reports having more than 14 million users and more than 35 million repositories, making it the largest host of source code in the world

5.x.x GitHub Account

Username: 16117743

Repositories:

INS-Thesis-Documentation

This repository contains commits of my weekly logs and semester reports.

INS-Thesis

This repository was setup before the project had a defined scope. This project will require a minimum of two separate repositories for the development of the front and back-end.

TODO: Add graph of weekly commits in week 10.

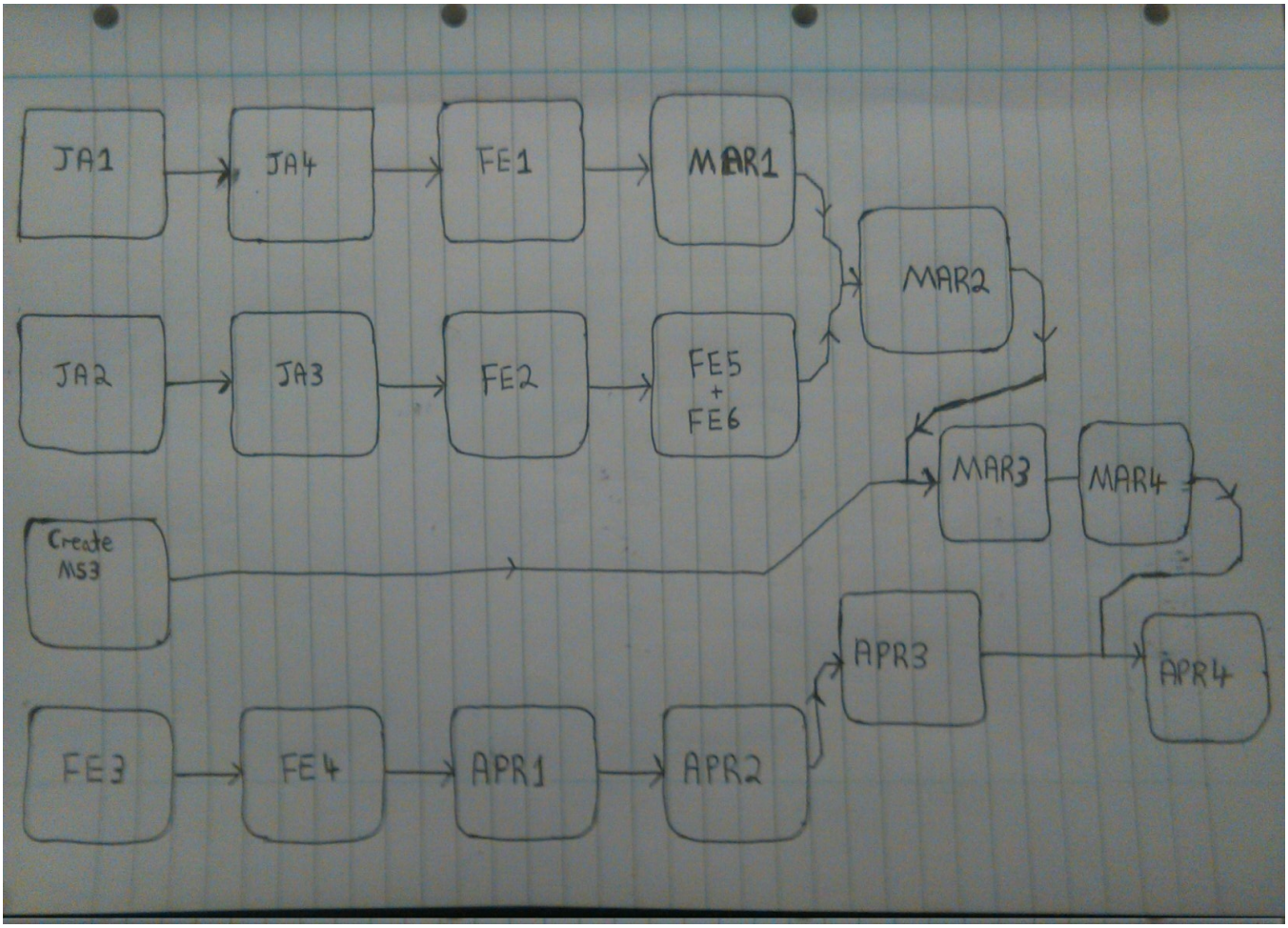
5.x Spring action plan

TODO: Revise specific tasks and assign realistic task duration.

Table of tasks for Spring semester

Task Description	Task Identity	Duration (days)	Preceding Task
Research similar microservices	JA1	5	
Research Gremlin	JA2	5	
Research IBM graph examples	JA3	5	JA2
Deploy and run MS1 that creates and transmits pseudo sensor data	JA4	5	JA1
Deploy and run MS2 that handles incoming data	FE1	5	JA4
Deploy a basic graph database on IBM graph with suitable pseudo data	FE2	5	JA3
Research Bluemix website deployment examples	FE3	3	
Deploy a basic website on Bluemix	FE4	3	FE3
Create permanent graph DB on POWER8	FE5	5	FE4
Create temporary DB on POWER8	FE6	5	FE5
Develop code for MS2 to store incoming data for a short period of time	MAR1	5	FE1
MS2 writes to temporary database every 15 minutes	MAR2	5	MAR1 & FE6
MS3 reads data from temporary database	MAR3	5	
MS3 writes to permanent POWER8 database	MAR4	5	MAR3
Develop map user interface for website	APR1	5	FE4
Identify important UL GPS coordinates for map user interface	APR2	5	APR1
Develop code for website to submit a simple database request	APR3	5	APR2
MS4 handles a simple website client database request	APR4	5	APR3 & MAR4

Spring Critical path map



Spring Gantt chart

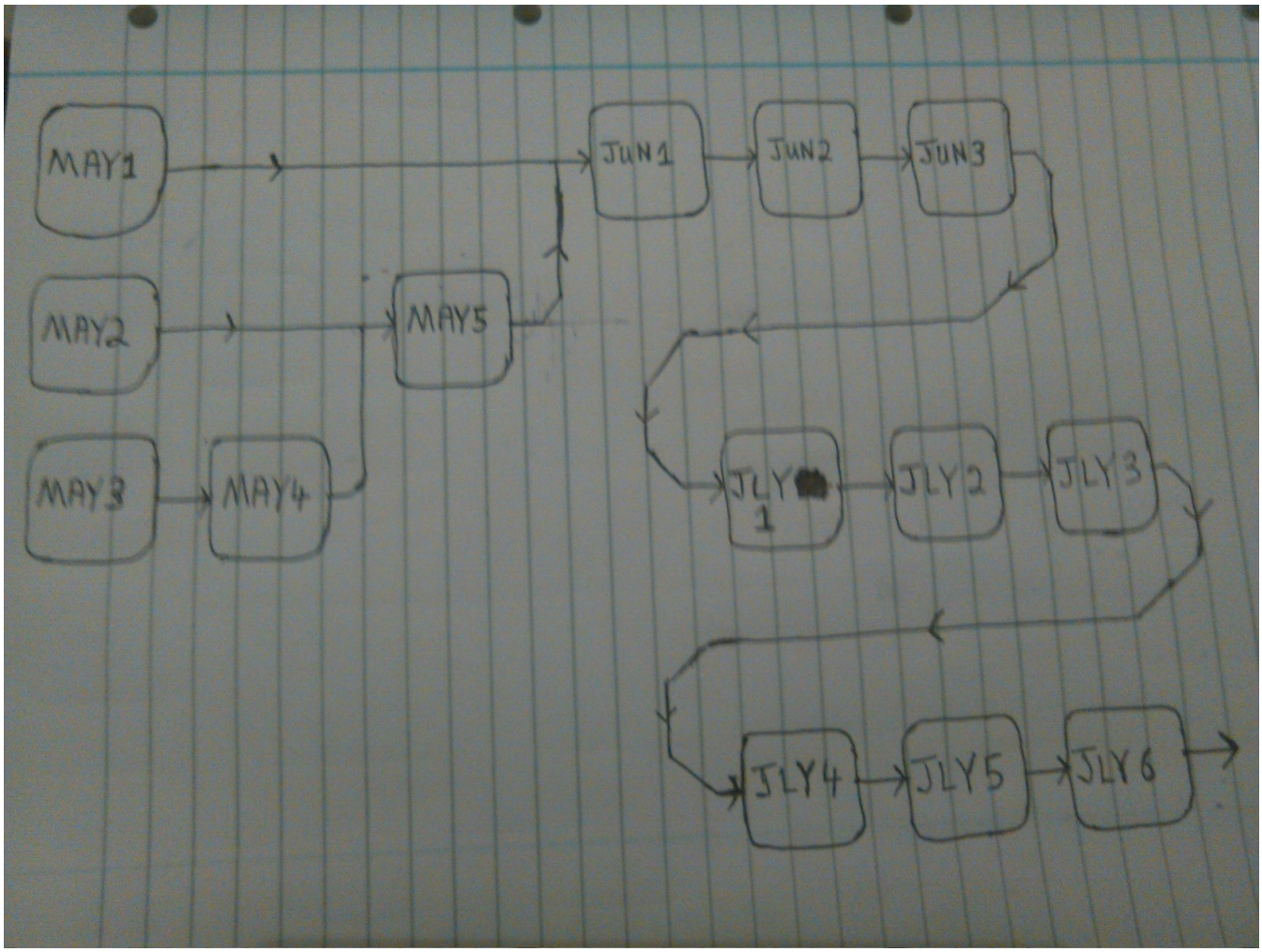
Task Identity	Start	Days	End
JA1	02/01/17	5	08/01/17
JA2	09/01/17	5	15/01/17
JA3	16/01/17	5	22/01/17
JA4	23/01/17	5	29/01/17
FE1	30/01/17	5	05/02/17
FE2	06/02/17	5	12/02/17
FE3	13/02/17	3	15/02/17
FE4	16/02/17	3	20/02/17
FE5	20/02/17	5	26/02/17
FE6	27/02/17	5	05/03/17
MAR1	06/03/17	5	12/03/17
MAR2	13/03/17	5	19/03/17
MAR3	20/03/17	5	26/03/17
MAR4	27/03/17	5	02/04/17
APR1	03/04/17	5	09/04/17
APR2	10/04/17	5	16/04/17
APR3	17/04/17	5	23/04/17
APR4	24/04/17	5	30/04/17

5.x Summer action planning

Table of tasks for Summer semester

Task Description	Task Identity	Duration	Preceding Task
Website displays DB readings on website map user interface	MAY1	5	APR4
Load balancing and scalability research	MAY2	5	
Increase funding	MAY3	5	
Increase MS1 instances	MAY4	5	MAY3
MS2 Automatically handles increase/decrease of incoming sensor data from MS1	MAY5	5	MAY2 & MAY 4
Test scalability of MS2 writing to temporary database	JUN1	9	MAY1 & MAY 5
Test scalability of MS3 reading from temporary database and writing to permanent database	JUN2	8	JUN1
End to end testing of website and database under increased scale	JUN3	3	JUN2
End to end testing of the whole system	JLY1	4	JUN3
Collect results	JLY2	2	JLY1
Analyze results	JLY3	3	JLY2
Discuss results with supervisor	JLY4	3	JLY3
Write up initial report and submit to supervisor for review	JLY5	3	JLY4
Edit initial report	JLY6	3	JLY5

Summer critical path map



Summer Gantt chart

Task Identity	Start	Days	End
MAY1	01/05/17	5	07/05/17
MAY2	08/05/17	5	14/05/17
MAY3	15/05/17	5	21/05/17
MAY4	23/05/17	5	29/05/17
MAY5	29/05/17	5	04/06/17
JUN1	05/06/17	9	15/06/17
JUN2	16/06/17	8	27/06/17
JUN3	28/06/17	3	02/07/17
JLY1	03/07/17	4	06/07/17
JLY2	07/07/17	2	10/07/17
JLY3	11/07/17	3	13/07/17
JLY4	14/07/17	3	18/07/17
JLY5	19/07/17	3	23/07/17
JLY6	24/07/17	3	26/07/17

5.x Agile implementation

TODO: Break tasks down into sprints.

5.x CI/CD DevOps

TODO: Discuss IBM DevOps pipeline delivery service and how I will implement my own pipeline.

6 Discussion

TODO: Extract the essential information from the weekly logs and reference them on GitHub.

6.x Week 5 summary

6.x Week 6 summary

6.x Week 7 summary

6.x Week 8 summary

6.x Week 9 summary

6.x Week 10 summary

7 Requirements of facilities and materials

7.1 Requirements

TODO: Find out exactly what technologies I'll be using.

7.2 Estimated cost

TODO: Find out how much it will cost.

8 References

TODO: A lot more referencing...

[1]"IBM", *Docker*, 2015. [Online]. Available: https://www.docker.com/IBM. [Accessed: 14- Oct- 2016].

[2]"IBM Bluemix - Containers", *Ibm.com*, 2016. [Online]. Available: http://www.ibm.com/cloud-computing/bluemix/containers/. [Accessed: 14- Oct- 2016].

[3]"Microsoft", *Docker*, 2015. [Online]. Available: https://www.docker.com/microsoft. [Accessed: 14- Oct- 2016].

[4]"Azure Container Service | Microsoft Azure", *Azure.microsoft.com*, 2016. [Online]. Available: https://azure.microsoft.com/en-us/services/container-service/. [Accessed: 14- Oct- 2016].

[5]"Get Started with Docker and AWS", *Docker*, 2015. [Online]. Available: http://www.docker.com/aws. [Accessed: 14- Oct- 2016].

[6]"Amazon EC2 Container Service – Docker Management – AWS", *Amazon Web Services, Inc.*, 2016. [Online]. Available: https://aws.amazon.com/ecs/. [Accessed: 14- Oct- 2016].

[7]"What is Docker?", *Docker*, 2015. [Online]. Available: https://www.docker.com/what-docker. [Accessed: 14- Oct- 2016].

[8]"IBM Bluemix Docs", *Console.ng.bluemix.net*, 2016. [Online]. Available: https://console.ng.bluemix.net/docs/services/graphdb/index.html. [Accessed: 15- Oct- 2016].

[9]"IBM Graph - API", *Ibm-graph-docs.ng.bluemix.net*, 2016. [Online]. Available: https://ibm-graph-docs.ng.bluemix.net/api.html. [Accessed: 15- Oct- 2016].

[10]"GremlinDocs", *Gremlindocs.spmallette.documentup.com*, 2016. [Online]. Available: http://gremlindocs.spmallette.documentup.com/. [Accessed: 15- Oct- 2016].

[11]A. TinkerPop, "Apache TinkerPop", *Tinkerpop.apache.org*, 2016. [Online]. Available: https://tinkerpop.apache.org/. [Accessed: 15- Oct- 2016].

[12]"Neo4j on IBM POWER8® - Neo4j Graph Database", *Neo4j Graph Database*, 2016. [Online]. Available: https://neo4j.com/neo4j-on-ibm-power8/. [Accessed: 15- Oct- 2016].

[13]"Neo4j Graph Database: Unlock the Value of Data Relationships", *Neo4j Graph Database*, 2016. [Online]. Available: https://neo4j.com/product/. [Accessed: 15- Oct- 2016].

[14]"Implementing a Containers-based Hybrid Cloud ERP environment using IBM Bluemix", *YouTube*, 2016. [Online]. Available: https://www.youtube.com/watch?v=9XbzUqCvl3I. [Accessed: 15- Oct- 2016].

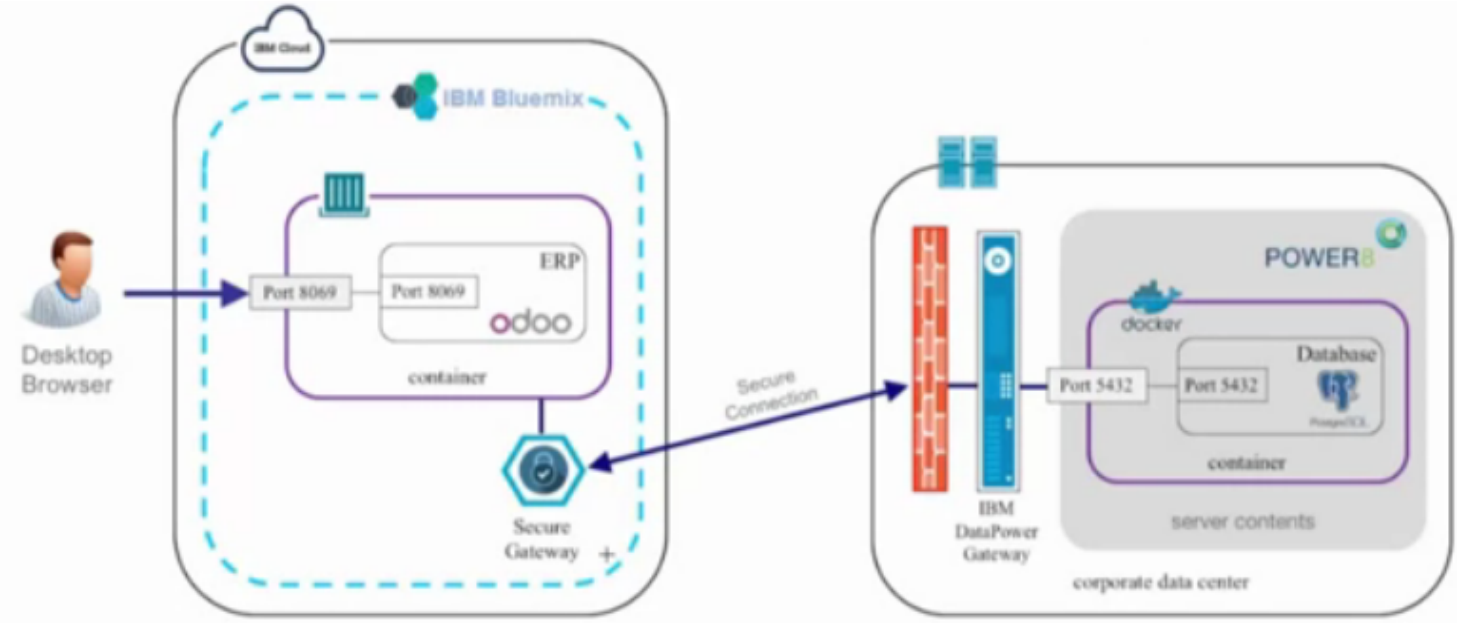
[15]"Neo4j Spatial: Finding Things Close to Other Things - Neo4j Graph Database", *Neo4j Graph Database*, 2011. [Online]. Available: https://neo4j.com/blog/neo4j-spatial-part1-finding-things-close-to-other-things/. [Accessed: 15- Oct- 2016].

[16]R. Catell, "Scalable SQL and NoSQL Data Stores", 2016. [Online]. Available: http://www.cattell.net/datastores/Datastores.pdf. [Accessed: 16- Oct- 2016].

[17]N. Staff, "Neo4j 2.2.0 – Massive Write & Read Scalability, Faster Cypher Query Performance, Improved Developer Productivity - Neo4j Graph Database", *Neo4j Graph Database*, 2015. [Online]. Available: https://neo4j.com/blog/neo4j-2-2-0-scalability-performance/. [Accessed: 16- Oct- 2016].

[20]T. Shields, "The Forrester Wave™: Application Security, Q4 2014", *Forrester.com*, 2016. [Online]. Available: http://www.forrester.com/pimages/rws/reprints/document/109101/oid/1-ZHIZ47. [Accessed: 20- Oct- 2016].

IBM Bluemix and POWER8 configuration example



The above figure demonstrates an example of an IBM container running within IBM Bluemix (left hand side) connecting to a database running within a Docker container that is deployed with IBM's POWER8 (right hand side). The IBM container is linked with the secure gateway service which helps provide a secure connection to the database.

To connect to the database you have to go through 2 layers. The first is a firewall. The second is the IBM DataPower gateway which will make a connection with the secure gateway on the Bluemix side. Behind these 2 layers is the POWER8 system which is hosting the Docker environment.

IBM DataPower Gateway

IBM Secure Gateway Service