

The end-to-end demonstration of health IOT data capture, transfer, manipulation, feedback and visualization using “CISCO Kinetic” platform

Submitted to the College of Science and Engineering in partial fulfilment of the requirements for the degree of Master of Science (Computer Science) at Flinders University – Adelaide, Australia

Gihan Gunasekara (GUNA0053) 2170011
Supervisor: Prof Trish Williams
COMP9700: Master Research (18 Units)

Declaration

I certify that this thesis does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person except where due reference is made in the text.

A handwritten signature in blue ink, appearing to read 'Gihan', with a long horizontal stroke extending to the right.

Gihan Gunasekara

Acknowledgement

Foremost, I would like to thank my thesis supervisor Prof. Trish. Williams, College of Science and Engineering, Flinders University for her patience, motivation, enthusiasm, immense knowledge, expertise and steering me in the right direction. Since the beginning, the door was always open to discuss and clarify about my research or writing.

I would also like to thank Ms. Ginger Mudd, Research Officer, Flinders Digital Health Research Centre, Flinders University for her valuable guidance and tremendous support throughout the research study and as the second reader of this thesis.

My special thanks to Mr. Kim Hawtin, Associate Lecturer, College of Science and Engineering, Flinders University for supporting me by answering my technical questions in different points.

I am particularly grateful for the assistance given by my ex business manager Ms. Carolyn Vowels.

In addition, I would like to thank my wife Gayani, for providing me continuous support, encouragement and wise counsel throughout my years of study. You are always there for me.

Thank you

Table of Contents

Declaration.....	ii
Acknowledgement	iii
Acronyms.....	vi
List of Figures.....	vii
List of Tables	viii
Abstract.....	1
1. Introduction.....	2
1.1. Background	2
1.2. Purpose of the Study	2
1.3. Aim of the Project	2
1.4. Objectives of the Study	2
1.5. Research Questions	2
1.6. Significance of the Study	2
2. Literature Review	4
2.1. IoT Architectures	5
2.1.1. Three level and Five level Architecture	5
2.1.2. Cloud and Fog based Architecture	5
2.1.3. Social IoT Architecture	7
2.2. IoT Capabilities	7
2.2.1. Monitoring	7
2.2.2. Controlling.....	7
2.2.3. Optimizing	8
2.2.4. Autonomy	8
2.3. Internet of Things in Health.....	8
2.4. IoT Platforms.....	10
2.5. Other IoT Application domains.....	12
2.5.1. Smart Cities and Environments	12
2.5.2. Industrial Automation.....	13
2.5.3. Supply and Logistics	13
2.5.4. Agriculture.....	13
2.6. Cisco Kinetic	14
2.6.1. Cisco Kinetic Platform	14
2.6.2. Dataflow in Kinetic	15
2.6.3. Cisco IoT Solutions	15
2.6.3.1. Oil & Gas: Refineries & Processing Plants	16

2.6.3.2.	Manufacturing.....	16
2.6.3.3.	Kinetic Cities	16
2.6.3.4.	Transportation	16
2.7.	Literature Review Summary and Research Questions/Challenge	17
3.	Methodology	18
3.1.	Theory Supporting the Study.....	18
3.2.	Methodology Selection	18
3.3.	Research/ Study design	19
3.3.1.	Step 1: Identify Problem & Motivate	19
3.3.2.	Step 2: Define Objectives of a Solution	19
3.3.3.	Step 3: Design & Development	20
3.4.	Limitation of Design Science	21
4.	Results	22
4.1.	Demonstration	37
4.2.	Evaluation	40
4.3.	Communication	40
5.	Discussion.....	41
6.	Conclusion	45
7.	References.....	46

Acronyms

IoT	Internet of Things
IoHT	Internet of Health Things
P2P	Person to person
M2P	Machine to person
M2M	Machine to machine
RFID	Radio frequency identification
SHD	Smart health devices
MEMS	Micro Electromechanical Systems
NFC	Near-field communication
LPWAN	Low power wide area network
SDKs	Software development kits
ERP	Enterprise resource planning
API	Application programming interfaces
GMM	Gateway Management Module
EFM	Edge & Fog Processing Module
DCM	Data Control Module
MQTT	Message Queuing Telemetry Transport
JSON	Java Script Object Notation
QOS	Quality of Service

List of Figures

Figure 1 - IoT Connectivity (Bradley et al., 2015)	4
Figure 2- IoT Cloud Architecture for a Remote Patient Monitoring System.....	6
Figure 3- A Typical Architecture of Edge Computing Network (Yu et al., 2018).	6
Figure 4– General View of the IoHT with Four Layers (da Costa et al., 2018)	9
Figure 5- The Eight Main Components of an IoT Platform (Iot-analytics.com, 2015)	11
Figure 6- Dataflow in Cisco Kinetic platform	15
Figure 7- Design Science Research Cycles (Hevner, 2007)	18
Figure 8- Design Science Research Methodology Process Model (Peffer et al., 2007).....	19
Figure 9- Publishing of Test Messages to a Topic “TestMyMQTT”	24
Figure 10 - Figure 10 - Subscribing to the topic “TestMyMQTT” to receive the Test Messages	25
Figure 11- Publishing of Test Messages to a Topic “TestMyMQTT”	25
Figure 12- Subscribing to the topic “TestMyMQTT” to receive the test messages	26
Figure 13-Successful Connection to the MQTT broker in Cisco Kinetic from the Mac OS	27
Figure 14 - Unsuccessful Connection to the MQTT broker in Cisco Kinetic from the Mac OS	27
Figure 15- Successful Connection to the MQTT broker in Cisco Kinetic from the Raspberry Pi	28
Figure 16 - Unsuccessful Connection to the MQTT broker in Cisco Kinetic from the Raspberry Pi	28
Figure 17 – MQTT Server and the topic structure	29
Figure 18- Process of Adding a Sub Server for “SenseHat”	30
Figure 19 - Process of Adding a Sub Server for “Smart Watch”	31
Figure 20 - Process of Subscribe a Topic to the “SenseHat”	32
Figure 21- Process of Subscribe a Topic to the “Smart Watch”	32
Figure 22- Process of Creating a Folder for “Smart Watch”	33
Figure 23- Process of Creating a Folder for “SenseHat”	33
Figure 24– Process of Adding a Dataflow for “SenseHat”	34
Figure 25- Process of Adding a Dataflow for “Smart Watch”	34
Figure 26- Sample “JSON” data file for sensor reading from the SenseHat	35
Figure 27– Start to End of an Activity	36
Figure 28– Sample “JSON” Data File for Treadmill, Walking and Weight Training	36
Figure 29– Dataflow Map from a Sensing Devices to Kinetic Platform	37
Figure 30- Data Reading from the Raspberry Pi SenseHat Sensors	37
Figure 31- Data Reading from the Raspberry Pi SenseHat Sensors Published to the MQTT Topic “Sensehat1”	38
Figure 32- Reading Data from JSON file that was produced from the Suunto 9 Smart Watch.....	38
Figure 33- Data Reading from the JSON file Published to the MQTT Topic “Suunto9A”	39

List of Tables

Table 1- Brief Summary of Sensing Technologies in IoHT	10
Table 2- Software Installation Information for the Mac OS and Raspberry Pi	23

Abstract

The Internet of Things (IOT) is a widely used concept across many industries, as it can be used to collect real time data in an environment using many devices. These devices can be smart phones, tablets, appliances, vehicles, sensors and actuators. The underlying connectivity is a combination of people, things and data. This project explores how to capture and extract data that relates to everyday interactions with the technology to provide a proof-of-concept to support the broader Campus Mental Wellness project. The Campus Mental Wellness project aims to enhance the experience of university students, improve engagement of university services and contribute to monitoring and promotion of the wellbeing of students. This project aims to demonstrate the capabilities of the Cisco Kinetic platform and the level to which dataflow modelling is possible using such a platform.

The project used a design science methodology to investigate how health IoT data can be extracted, manipulated and moved to various applications depending on the output requirements. This allowed modelling of the end-to-end dataflow which tracks upstream data from IoT devices to the Cisco Kinetic Platform and subsequently to a storage, analytics or visualisation system. Data-flow mapping and creating a visualization exemplar of the capability of the Cisco Kinetic platform is fundamental to understanding the platform and its potential capability. The resulting conceptual dataflow framework for the health IoT end-to-end process is vital to explore the possibilities for health IOT data use in the Campus Mental Wellness project using this platform.

The result of this project is to provide a proof-of-concept of the end-to-end dataflow of health IOT data capture, transfer, manipulation, feedback and visualization using Cisco Kinetic platform. The impact is that it will make modelling of new device data easier for researchers, as well as contribute to helping non-technical (clinical) collaborators in the mental wellness project to understanding the capture and use of data for the project. This proof-of-concept will be used to explore further how specific health IOT devices can be used to meet the broader aims of the parent Campus Mental Wellness research project.

1. Introduction

1.1. Background

The project explores how to capture and extract data that relates to everyday interactions with the technology to provide a proof-of-concept for dataflow mapping to support the broader Campus Mental Wellness project which aims to "enhance the experience of university students, improve engagement of university services and contribute to monitor and promote wellbeing of students". (*Prof. T. Williams, Flinders Digital Campus Mental Wellness Project*).

1.2. Purpose of the Study

The end-to-end demonstration of health Internet of Things (IOT) data capture, transfer, manipulation, feedback and visualization using "Cisco Kinetic" IOT platform.

1.3. Aim of the Project

To explore the possibilities for health IOT data for use in the mental wellness analysis and interventions for university students. This will include demonstrating the capabilities of the Cisco Kinetic IOT platform and the level to which dataflow modelling is possible using such a platform.

1.4. Objectives of the Study

- Dataflow mapping and visualization exemplar of the capability of the Cisco Kinetic platform.
- Conceptual framework for the health IoT end-to-end process.

1.5. Research Questions

- A. How can a map and visualization of the dataflow in Cisco Kinetic platform be developed?
- B. Can an end-to-end dataflow be modelled incorporating the Cisco Kinetic platform?

1.6. Significance of the Study

If a proof-of-concept can be demonstrated using "Cisco Kinetic" this will allow further development and analysis of university student IoT and other data streams to improve the

mental wellness of the students at university. This will assist non-technical collaborators in the project to understand how data can be collected and analysed using this IoT platform.

This thesis includes the results of a Literature Review that focuses on whether or not the Kinetic Platform has been used in Health IoT systems. The Literature Review is followed by the results of using Design Science as a research methodology for developing a proof-of-concept. The results of this research is described in detail including all of the steps involved in developing a Health IoT dataflow and a proof-of-concept that uses the Cisco Kinetic platform.

2. Literature Review

The literature review firstly describes IoT architectures and capabilities, and then analyses the use of IoT in Health, the associated IoT Platforms and other IOT application domains. Subsequently, there is a review of the Cisco IoT solutions and Cisco Kinetic platform.

With the high availability of the broadband internet and fast-growing internet technologies, the use of the internet to connect objects and devices to collect real time data for different purposes is in action. As a result, Internet of Things (IoT) is progressively becoming more popular.

IoT is a widely used concept across many industries, as it can be used to collect real time data in the environment using many devices. These devices can be smart phones, tablets, appliances, vehicles, sensors and actuators (Bradley et al., 2015). The underlying connectivity is a combination of People, Things and Data. The communication is across the person to person (P2P), machine to person (M2P) and machine to machine (M2M), as shown in Figure 1.

As described by Mahdavinejad et al (2018) Internet of Things (IoT) consist of four main components: Sensors, Processing networks, Data analysis and System Monitoring. Authors further explain that advances were made to IOT due to frequent use of RFID tags, affordability of sensors, high end web technology developments and changes made to communication protocols.

This image has been removed due to copyright restrictions

Figure 1 - IoT Connectivity (Bradley et al., 2015)

2.1. IoT Architectures

A standard for IoT architecture has not been defined, as it depends on how the developers want to connect the devices to the internet and the expected level of the scalability, performance, security and interoperability. The IoT architecture will depend on the domain (Ray, 2016) and the architecture of the IoT can be treated as physical, virtual or a hybrid system with a collection of active devices: sensors, actuators, IOT protocols, communication layers.

Sensing → Identification → Actuation → Communication → Management

Various IoT architectures have been proposed (Sethi and Sarangi, 2017) by many researchers: Three level architecture, Five level architecture (perception layer, transport layer, processing layer, application layer, business layer), Cloud & FOG based Architecture, Social IoT and Machine to Machine distributed architecture.

2.1.1. Three level and Five level Architecture

The three level architecture is the basic architecture defined for IoT (Wu et al., 2010) and consists of a perception layer, a network layer and an application layer. The purpose of the perception layer is object identification and information gathering; a network layer transmits and processes information; and an application layer delivers the services to the user (Sethi and Sarangi, 2017). The five level architecture is the enhanced version of the three level basic architecture and includes the perception layer, transport layer, processing layer, application layer and business layer (Wu et al., 2010). The transport layer handles the transfer of the sensor data from perception to process layer. The processing layer is to store, analyze and process data. Managing the applications, business models and profit models is done by the business layer.

2.1.2. Cloud and Fog based Architecture

In the cloud based architecture, Applications sit on the top level, the Cloud in the middle level and the Network of smart things in the bottom level (Sethi and Sarangi, 2017). Cloud architecture is more popular due to the services (infrastructure, platform, storage, software) provided by the Cloud and the great flexibility and scalability.

Figure 2 shows the IoT cloud architecture for a remote patient monitoring system.

This image has been removed due to copyright restrictions

Figure 2- IoT Cloud Architecture for a Remote Patient Monitoring System
(Hassanaliheragh et al., 2015).

Fog architecture is an extended version of the Cloud architecture (Bonomi et al., 2014) to support and address very low latency, geo distributed and fast mobile applications and large control systems over the cloud.

Figure 3 shows a typical architecture of edge computing network.

This image has been removed due to copyright restrictions

Figure 3- A Typical Architecture of Edge Computing Network (Yu et al., 2018).

2.1.3. Social IoT Architecture

Social IoT is based on the social relationship of the objects, where the architecture is based on a server side and an object side (Atzori et al., 2011). The server side consists of three layers: base layer, component layer and application layer. The object side consists two layers: object layer and object abstraction layer or the social layer. The base layer has a database to store metadata, details of devices, attributes and relationship details. The Component layer is to handle interaction with devices and querying. The application layer is to provide services to the user. The object layer allows the devices to connect with each other and to communicate between them through pre-defined protocols. Managing the application execution, running queries are handled by the social layer (Sethi and R. Sarangi, 2017).

2.2. IoT Capabilities

Using the best available technology in the business is a wise decision that business owners can make in the modern business world to obtain the competitive advantage and to provide the best customer services. According to Porter & Heppelmann (2014), the level of intelligence and connectivity enables new functions and the capabilities of IoT can be categorized in to four main areas: Monitoring, Controlling, Optimizing and Autonomy.

2.2.1. Monitoring

Using sensors and data sources, the device's operating conditions, safety parameters, predictive maintenance and other environmental factors can be monitored by sending performance alerts or any other change alerts to users or groups (Porter & Heppelmann, 2014). This has been successful due to the powerful tools built on IoT technology (Tao et al., 2014).

2.2.2. Controlling

IoT can be controlled remotely as long as the devices are connected through wire or wirelessly to the network. Therefore, embedded devices can be controlled to meet the need. If the sensors detect a drop in the oxygen level of a particular work room, evacuation can be arranged or pump more oxygen by other mean (Porter & Heppelmann, 2014).

2.2.3. Optimizing

IoT allows optimization of device performance in many ways because of the real time monitoring and controlling capabilities provided by the devices. Historical data and analytics collected through the devices can be used to make important decisions to increase performance of the production lines and resources utilization (Porter & Heppelmann, 2014).

2.2.4. Autonomy

The sum of monitoring, controlling and optimizing lead to autonomy. Moreover, autonomy reduces the need of physical operators and improves the health and safety in the environment, supporting smooth operations in remote locations (Porter & Heppelmann, 2014).

2.3. Internet of Things in Health

Due to the nature of the applicability of IoT in numerous areas, the healthcare sector is focusing on many areas where IoT can apply. The use of IoT based healthcare methods will improve the quality of a patient's life and the effectiveness and efficiency of the treatments and tests.

The Internet of Health Things (IoHT) contains inter-connectable objects that have the capacity to exchange and process data to improve the health of patients (da Costa et al., 2018). These objects can be biosensors, wearable devices and other medical devices to collect and combine data about vital signs of patients to monitor a patient's health status (da Costa et al., 2018). Further machine learning techniques can be used to transform the data gathered by these objects in to meaningful information. The patient centric view proposed by (da Costa et al., 2018) consist of four layers: acquisition, storage, processing and presentation.

Figure 4 shows the general view of IoHT with four layers.

This image has been removed due to copyright restrictions

Figure 4– General View of the IoHT with Four Layers (da Costa et al., 2018)

The Acquisition layer collects data from smart health devices (SHO) via Bluetooth, Wi-Fi and other protocols. The storage layer stores data. The processing layer is for data analysis using advanced algorithms, machine learning techniques and the presentation layer shows the results from previous layers.

Sensing and identification technologies are used to recognize physical objects and obtain information related to health (Qi et al., 2017). IoT sensing technology is capable of obtaining various health data from the commercial wearable devices and mobile applications using hybrid sensors (Qi et al., 2017).

Table 1 shows a brief summary of sensing technologies in IoHT.

Sensor category	Sub category	Description	Models
Wearable sensors	Inertial sensors	small scale Micro Electro Mechanical Systems (MEMS) which measure physical activity	Accelerometer, Gyroscopes, Pressure sensors, Magnetic field sensors
	Physiological sensors	personal data relate to specific health conditions	Spirometer, Electrooculography, galvanic skin response
	Image sensor	emotions, activities captured as video or image format	SenseCam
	Location sensors	based on a specific location	GPS, Blood pressure cuff, Electrocardiogram
Ambient sensors	Environmental sensors		Thermometer, Hygrometer
	Binary sensors		Window contact, Light switch, Door contact
	Location sensors		RFID, Infra-red, Zigbee
	Tags		RFID tags, NFC tags

Table 1- Brief Summary of Sensing Technologies in IoHT

2.4. IoT Platforms

The IoT platform is considered the central backbone of the Internet of Things infrastructure (Iot-analytics.com, 2015). The platform refers to a multi-layer technology that enables an easy provision, easy management and automation of interconnected devices.

An IoT platform is a multi-layer technology that enables straightforward provisioning, management, and automation of connected devices within the Internet of Things universe (Kaa IoT platform 2019). It connects hardware, however diverse, to the Cloud by using flexible connectivity options through cellular, satellite, Wi-Fi, Bluetooth, RFID, NFC, LPWAN and Ethernet (Leverage.com, 2016). It includes enterprise-grade security mechanisms such as antivirus/ antimalware protection, inbuilt firewall systems, intrusion prevention/detection systems, multi device authentication, certificate and cryptographic key enabled features (Press,

2017), and broad data processing powers with enhanced batch processing/ real-time processing technologies: “Hadoop5”, “Storm”, “S4” (Malek et al., 2017) . For developers, an IoT platform provides a set of ready-to-use features that greatly speed up development of applications for connected devices as well as take care of scalability and cross-device compatibility. The eight main components of an IoT platform described by IoT-Analytics.com (2015) is shown in the Figure 5.

This image has been removed due to copyright restrictions

Figure 5- The Eight Main Components of an IoT Platform (IoT-Analytics.com, 2015)

Each component of the IoT platform is described below:

- Database
Database management system distributed across the sensor nodes. This should be capable of catering for big data capable of storing both structured and unstructured data.
- Connectivity & Normalization
The connectivity layer brings the various protocols and data into a single software interface.
- Device Management
This module ensures that interconnected devices are up and running. It also ensures software versions and applications are up to date. Managing software updates/

firmware, provisioning of devices and remote configuration are some of the tasks carried out by the device management module.

- Processing & Action Management

The captured data is processed.

- Data Visualization

Transforms the data analytics into a visualization, identifying patterns and trends.

- Analytics

Performs advanced analytics and queries from IoT captured data streams.

- Additional tools

Development tools for developers such as prototype testing platforms and management tools for management focused activities such as day-to-day operations and reporting functions.

- External Interfaces

External interfaces are to integrate third party applications and systems such as software development kits (SDKs), enterprise resource planning (ERP) systems, application programming interfaces (APIs) and other gateways.

2.5. Other IoT Application domains

IoT is considered a multidisciplinary concept, because of the wide variety of technologies it uses, the various capabilities of the devices, the different strategies used and its market profile (Gardašević et al., 2017). Due to emerging IoT activities, it has been used across many domains.

2.5.1. Smart Cities and Environments

IoT solutions have been used to improve the quality of day-to-day life in dynamic urban environments (Gardašević et al., 2017). Areas covered: service management, environmental pollution control, traffic management. Maintenance of garbage bins in major cities were one of the successful area. Bins are monitored using sensors and notified relevant maintenance

teams when the bins needed emptying and removing. This helps to bin collecting and any bin replacements to schedule in advance to minimise the disturbance with garbage collection trucks in the roads. Within the smart city concept, IoT has been used widely to improve the road traffics. Traffics on the roads were continuously monitored using IoT devices during peak, off peak and holidays traffic to identify traffic patterns and helped to adjust the timing and accommodate the commuters and drivers to keep moving with minimum disturbance. Further IoT devices used to monitor real-time traffic incidents.

2.5.2. Industrial Automation

IoT technologies have been used to lower the cost and improve the product quality by automating the processes in major areas such as inventory management, quality control and product flow monitoring (Edureka, 2019). The Operation and maintenance of plants in remote locations from a central location is highly benefited to a company with regard to time and cost (Breivold and Sandström, 2015). I.e. – Intelligent applications such as` detecting machine failures, calibration of instruments in remote plants, security automation.

2.5.3. Supply and Logistics

IoT in supply chain and logistics perform an important role. “Logistics 4.0” is a high technological IoT system that transforms a factory to a “Smart Factory” (Hoey, 2018). According to Forbes.com (2019), this is a combination of edge computing and IoT that is a real time system, and which provides sense and feedback while providing maximum level of security to data.

2.5.4. Agriculture

The IoT based system “TracoVino” is used to sense the environmental parameters such as rainfall, soil humidity and temperature in vineyards and upload the data to the Cloud. (Iotworldtoday.com, 2019). IoT based “beehive scale” used in beekeeping to avoid “honey bee collapse disorder” by sending alerts to bee keeper’s mobile phone when a beehive’s weight changes (Iotworldtoday.com, 2019). The “Smart Greenhouse” is a widely used IoT system to control the environmental parameters using sensors of a nursery (Edureka, 2019).

2.6. Cisco Kinetic

Cisco Kinetic is a scalable, open system and adaptable platform where IoT devices can connect securely, allocate compute power and produce the right data for the correct apps at the correct time. High level security technologies are integrated with the software and in network layers. (Cisco Kinetic Overview, 2017)

2.6.1. Cisco Kinetic Platform

By using three integrated modules, Kinetic platform addresses all levels of delivering significant business outcomes to clients. The Cisco Kinetic platform consists of three modules:

Gateway Management Module (GMM) – This is a secure, scalable Cloud based module that enables remotely the provisioning, configuration, deployment, management and monitoring of the Cisco gateways through a highly secured VPN tunnel.

Edge & Fog Processing Module (EFM) – This is an open and modular component which is capable of incorporating third party micro services. EFM can add computing power to the distributed network anytime, anywhere as needed by allowing connections of a wide-range of devices and sensors to the network. Also, this module can execute complex rules on data in motion to optimise the dataflow. Further EFM can be used for advanced monitoring and diagnosing of equipment.

Data Control Module (DCM) – Installing and managing fog applications is done by this module. The DCM is capable of aggregating data from multiple sensors, delivering data from multiple assets to multiple applications, interconnecting cloud applications etc.

Key features;

- Software Connectors
- Data Transformer
- Edge & Fog Processing
- Data Models
- Data Controls
- Northbound Application Connectors
- Cloud-based Gateway Management

2.6.2. Dataflow in Kinetic

The Cisco Kinetic platform enables the connection of IoT devices to the network easily and to extract and securely control streams of incoming data from devices, move data to various applications for further actions depending on the user need.

Figure 6 shows the dataflow in Cisco Kinetic platform.

This image has been removed due to copyright restrictions

Figure 6- Dataflow in Cisco Kinetic platform

The process consists of three stages;

Extract Data – extract data from all connected IoT devices to the network (protocol independent).

Compute Data – Edge and Fog used to compute the data (reduce latency, efficient in resources utilization).

Move Data – move data programmatically to multiple apps in multiple locations, can enforce policies for data privacy, security and ownership. (multi-cloud, multi-party, multi-locations).

2.6.3. Cisco IoT Solutions

Cisco has introduced the Kinetic solutions to businesses across many industries to meet business objectives and to reach better business outcomes. The following section briefly discusses the industries in which Kinetic solutions have been adopted.

2.6.3.1. Oil & Gas: Refineries & Processing Plants

The “Cisco Connected Refinery” solutions focus mainly on achieving real-time business benefits by using intelligence. The intelligence is derived by obtaining data from the refinery plants. Predictive maintenance techniques are used to identify machine and equipment failures, gas and other dangerous chemical leakages detection, task automation. Advanced security techniques to detect any threat landscape are also implemented using Cisco Kinetic. (Cisco.com, 2019)

2.6.3.2. Manufacturing

“Cisco Kinetic Manufacturing” platform enables real-time access to data sources without interrupting equipment and to integrated data storages across manufacturing, processing and control rooms while maintaining visibility and control. (Cisco.com, 2019)

2.6.3.3. Kinetic Cities

This is a cloud-based platform, that enables and facilitates the automation, high secure data sharing across the existing community infrastructures, applications, solutions and all connected devices (Cisco, 2019). Using the network as the infrastructure foundation, this approach meets the needs of present and future managed cities and business services. This IoT platform, supports data aggregation, normalization and analyzing. The Cisco platform consists of server-side scripts, platform centric libraries, object relational DB systems and markup languages to support the kinetic platform’s core and extended functions to perform effectively and efficiently (Cisco, 2019).

2.6.3.4. Transportation

In transportation, Kinetic covers many areas such as: Aviation by providing new services to clients & workers for greater mobility and collaboration; in Maritime by offering efficient traffic management, cargo routing and enhancing a port’s security features; in Mass Transit by location tracking of fleet vehicles, predictive maintenance for fleet vehicles, Wi-Fi connection to passengers; in Rail by improving operating efficiencies, Wi-Fi for trains, simplified maintenance; in Roadways by reducing traffic congestions, route rerouting to avoid busy interchanges (Cisco, 2019).

2.7. Literature Review Summary and Research Questions/Challenge

The results of the Literature Review demonstrates that the Kinetic Platform has been used for many industrial applications including smart cities, manufacturing, transportation, oil and gas but not for Health IoT projects.

It also shows that there is a great deal of activity in Health IoT such as monitoring patient vital signs.

But there is no indication in the research literature that the Cisco Kinetic Platform is being used in Health IoT systems. Thus, the research outlined in this thesis addresses the following questions:

- a) How can a map and visualization of the dataflow in Cisco Kinetic platform be developed for Health IoT?
- b) Can an end-to-end dataflow be modelled incorporating the Cisco Kinetic platform?

3. Methodology

3.1. Theory Supporting the Study

As IoT technology is evolving rapidly, it opens up many opportunities for healthcare. IoT devices can collect valuable and important data to give additional insight. Adding IoT infrastructure can convert a healthcare facility into a smart healthcare facility enabling health monitoring and management, remote care, vital sign monitoring etc. But modeling the end-to-end flow is complex. This research focuses on developing a map of the dataflow and a model of the end-to-end dataflow using “Design Science” research. As Hevner (2007) explains, design science research consists of three inherent research cycles: “Relevance Cycle”, “Design Cycle” and “Rigor Cycle”. Figure 7 shows the design science research cycles.

This image has been removed due to copyright restrictions

Figure 7- Design Science Research Cycles (Hevner, 2007)

The Relevant Cycle bridges the gap between contextual environment and the design science activities in the research. The Design Cycle is defined as an iterative activity iterating between “Build Design Artefact & Process” and “Evaluation” of the research. The knowledge base such as theories & methods, experience & expertise, Meta artefacts and the design science activities are connected by the Rigor Cycle (Hevner, 2007).

3.2. Methodology Selection

As explained by Peffers et al (2007), a methodology will help Information System (IS) researchers to construct and present superior “design science research” in IS. Further the authors explain that, a design science research methodology consists of three elements:

“Conceptual Principles” which is referred to as the meaning of the design science research, “Practice Rules” and “Process” as the way of carrying out the research and presenting it. Moreover, Hevner et al (Hevner et al., 2004) describe Design Science as a problem solving and solution oriented process.

Figure 8 shows the Design Science Research Methodology Process Model.

This image has been removed due to copyright restrictions

Figure 8- Design Science Research Methodology Process Model (Peffer et al., 2007)

3.3. Research/ Study design

3.3.1. Step 1: Identify Problem & Motivate

Cisco Kinetic has been a reliable and robust IoT platform for many industries. This project looks at the use of the Cisco Kinetic platform for Health IoT data and how it can be extracted, manipulated and moved into various applications depending on the output requirements. On the other hand, this research looks at modelling end-to-end information flow which upstream data from IoT devices to the Kinetic platform and subsequently to a storage, analytics or visualisation system.

3.3.2. Step 2: Define Objectives of a Solution

The Cisco Kinetic platform has been heavily used in manufacturing, transportation, smart cities and oil & gas: refineries & processing plants. The overall objective is to construct a conceptual

framework for health IoT end-to-end process and establish a proof-of-concept for the use of Kinetic platform in the Health sector.

3.3.3. Step 3: Design & Development

In the Cisco Kinetic platform, data can be collected through external data sources, devices, sensors or from the web. (I.e. Raspberry Pi SenseHat, Smart Watch). Using a programming language like Python, programming code can be developed to read the data from sensors and publish to Message Queuing Telemetry Transport (MQTT) or to read a data source and convert it to a Java Script Object Notation (JSON) file format which can then be used as an input. The JSON file format is commonly used to store information in an organized manner for easy access. Also, data from a device such as a smart watch can be used with an application programme interface (API). Further a data stream from a third party such as the bureau of meteorology (BOM) website also can feed into Kinetic. In this example, a BOM data file is huge and using a program written in the Python language, it is possible to extract only the relevant data and publish to the MQTT. The format and the refresh time interval of the data will be based on what we want to achieve.

a) Selecting IoT devices

To conduct this research, two devices were selected as IoT devices. “Suunto 9 Baro” which is a durable multisport GPS smart watch and a “SenseHat” which is an electronic board; an add-on component to a Raspberry Pi computer.

Suunto 9 Baro Smart Watch - Key features such as measuring wrist heart rate, over eighty sports modes to track, GPS navigation, intelligent battery modes, weather functions and fused track for more accurate track and distance readings. The watch can be connected with a smart phone via the “Suunto” mobile app. This smart watch is capable of “Activity Tracking” which includes step counter, calories burned, activity targets, activity history, calorie burn rate and heart rate during activities, daily minimum heart rate tracking and “Sleep Tracking” which includes sleep duration, average heart rate during sleep, bed time, time awake.

Raspberry Pi SenseHat – An add-on electronic board consisting of a grid of eight by eight RGB LED pixel board, a joy stick and six sensors: Gyroscope, Accelerometer, Magnetometer, Temperature, Barometric Pressure and Humidity. The Raspberry Pi is a single board computer

with a quad core processor, 1GB SDRAM, dual band wireless LAN, Gigabit Ethernet port, Bluetooth and Micro SD format for the operating system (OS) and data storage.

Step 4: Demonstration

The demonstration step in design science involves the actual use of a resulting artefact to “solve” a problem through any number of ways including “experimentation” (Peppers et al., 2007).

In this research it is expected that by using Python programs, data is captured from IoT devices and transmitted through the MQTT communication protocol. Further the data should appear in the dashboard of the Kinetic Platform and specifically in the MQTT channels.

Step 5: Evaluation

Evaluation is considered a vital element of the design science research process and it delivers important feedback to the development phase. Furthermore, the completeness and effectiveness of a design artefact depends on how far it satisfies the identified requirements and constraints of the problem (Hevner et al., 2004).

Step 6: Communication

The communication step in design science is closely associated with the knowledge of the discipline culture (Geerts, 2011). The purpose of this step is to socialise and educate others in the discipline about the problem, the solution and the relative value, which may include innovation and effectiveness, to other researchers and stakeholders.

3.4. Limitation of Design Science

The methodology relies on iteration as part of the methodology and that the Evaluation and Communication activities often identify where revision in design are necessary – thus the whole design science process creates a cyclic re-iteration loop with the design and the method can result in persistent revisions if the scope is not well defined. Further Carlsson (2005) suggests that consideration of real world application and events as in critical realism, is needed during the research process to discern between the theoretical and practical application of the methodology.

4. Results

Step 1: Identify Problem & Motivate

The problem and motivation for this research is to better understand and visualise the dataflow using IOT devices and an IOT platform.

Step 2: Define Objectives of a Solution

The objective of a solution is to construct an easily understandable conceptual framework for health IoT end-to-end process and establish a proof- of- concept for the use of Kinetic platform in the Health sector.

Step 3: Design & Development

Having identified and selected suitable devices (as explained in the research design section 3.3, the result below demonstrates the process that was used to design and develop the solution.

Installing SenseHat on Raspberry Pi:

1. Update Advance Package Tool (APT)
sudo apt-get update
2. Install SenseHat package
sudo apt-get install sense-hat
3. Reboot the Raspberry Pi
sudo reboot

b) Installing the software for the study

Table 2 shows the software installation information for the Mac OS and Raspberry Pi

OS SW	Mac OS (High Sierra 10.13.6)	Raspberry Pi (Raspbian)
Python3	Prerequisites for Python3 - Install Xcode – which is Apple’s Integrated Development Environment (IDE) - Install Brew – Package manager for Mac OS Install Python3 - \$ brew install Python3	Install Python3 - sudo apt-get install Python3-pip
Paho-MQTT	Open Terminal window - pip3 install paho-mqtt	Open Terminal window - pip3 install paho-mqtt
Mosquitto	Open Terminal window - \$ brew install mosquitto	Open Terminal window - sudo apt-get install mosquitto - sudo apt-get install mosquitto-clients

Table 2- Software Installation Information for the Mac OS and Raspberry Pi

Message Queuing Telemetry Transport (MQTT) described as a machine to machine IoT connectivity protocol and has many important characteristics such as light weight, low power consumption, minimized data packets, effective and efficient in information distribution (Mqtt.org, n.d.). Python is an open-source programming language which is considered as interpretable, high level and general purposed. Version 3.7 is used in this study. “Mosquitto is an open source message broker which implements the MQTT protocol” (Eclipse Mosquitto, 2018).

c) Testing the Message Queuing Telemetry Transport (MQTT) and Mosquitto Client on Mac OS

To test this scenario, it is necessary to open two separate terminal windows in Mac OS and publish the topic from one terminal window and subscribed to the published topic from the other terminal window.

The syntax of the command to publish the topic:

```
mosquitto_pub -h "name of the host" -t "topic name" -m "message"
```

Steps to follow:

- Open a terminal window
- Type `mosquitto_pub -h localhost -t TestMyMQTT -m TestMSG-1`
- Press "Enter"

Figure 9 show the publishing of three test messages to a topic "TestMyMQTT"

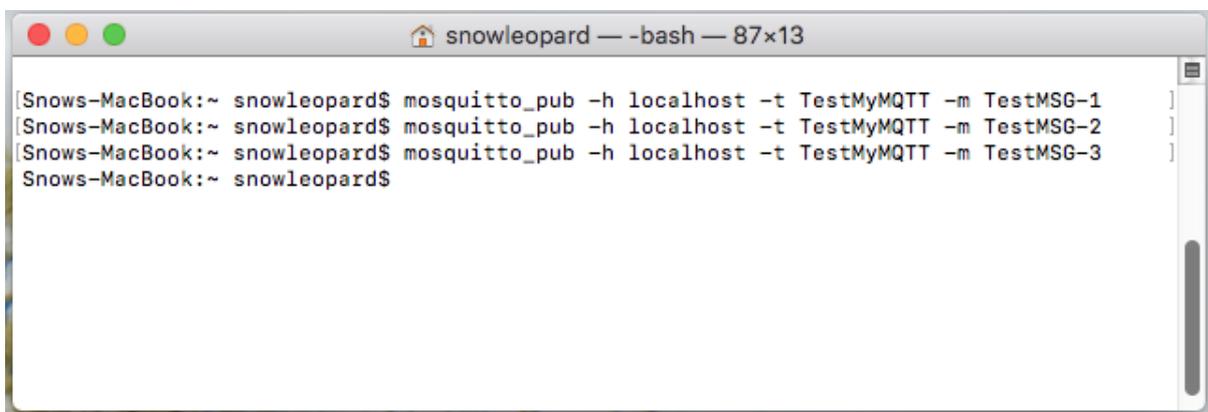
A screenshot of a terminal window titled "snowleopard" with a window size of "87x13". The terminal shows three lines of commands being executed: `mosquitto_pub -h localhost -t TestMyMQTT -m TestMSG-1`, `mosquitto_pub -h localhost -t TestMyMQTT -m TestMSG-2`, and `mosquitto_pub -h localhost -t TestMyMQTT -m TestMSG-3`. Each command is followed by a closing bracket "]" on the same line, indicating successful execution. The prompt changes from `[Snows-MacBook:~ snowleopard$` to `Snows-MacBook:~ snowleopard$` after the third command.

Figure 9- Publishing of Test Messages to a Topic "TestMyMQTT"

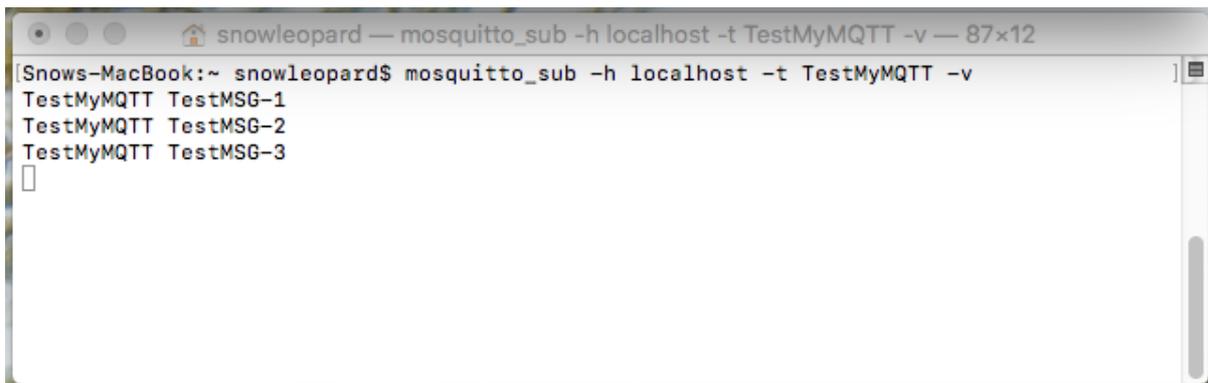
The syntax of the command to subscribe to the topic:

```
mosquitto_sub -h "name of the host" -t "topic name" -v
```

Steps to follow:

- Open a terminal window
- Type `mosquitto_sub -h localhost -t TestMyMQTT -v`
- Press "Enter"

Figure 10 shows the subscribing to the topic “TestMyMQTT” to receive the test messages

A terminal window titled "snowleopard — mosquitto_sub -h localhost -t TestMyMQTT -v — 87x12". The prompt is "Snows-MacBook:~ snowleopard\$". The command entered is "mosquitto_sub -h localhost -t TestMyMQTT -v". The output shows three lines of received messages: "TestMyMQTT TestMSG-1", "TestMyMQTT TestMSG-2", and "TestMyMQTT TestMSG-3". A cursor is visible on the line following the last message.

```
snowleopard — mosquitto_sub -h localhost -t TestMyMQTT -v — 87x12
[Snows-MacBook:~ snowleopard$ mosquitto_sub -h localhost -t TestMyMQTT -v
TestMyMQTT TestMSG-1
TestMyMQTT TestMSG-2
TestMyMQTT TestMSG-3
]
```

Figure 10 - Figure 10 - Subscribing to the topic “TestMyMQTT” to receive the Test Messages

d) Testing the Message Queuing Telemetry Transport (MQTT) Client on Raspberry Pi

To test this scenario in Raspberry Pi, it is necessary to open two separate terminal windows and publish the topic from one terminal window and subscribed to the published topic from the other terminal window.

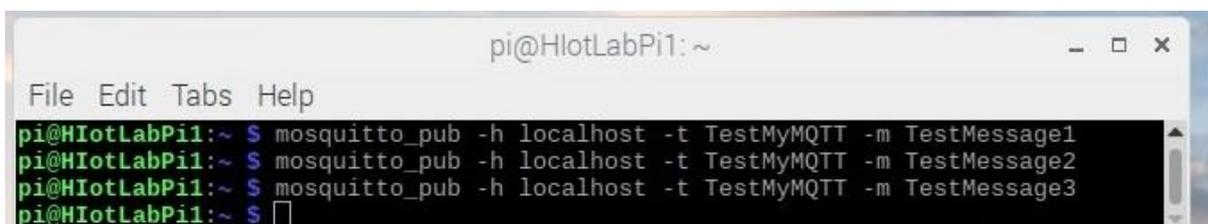
The syntax of the command to publish the topic:

```
mosquitto_pub -h “name of the host” -t “topic name” -m “message”
```

Steps to follow:

- Open a terminal window
- Type `mosquitto_pub -h localhost -t TestMyMQTT -m TestMessage1`
- Press “Enter”

Figure 11 show the publishing of three test messages to the topic “TestMyMQTT”

A terminal window titled "pi@HlotLabPi1:~". The prompt is "pi@HlotLabPi1:~". The command entered is "mosquitto_pub -h localhost -t TestMyMQTT -m TestMessage1". The output shows three lines of published messages: "mosquitto_pub -h localhost -t TestMyMQTT -m TestMessage1", "mosquitto_pub -h localhost -t TestMyMQTT -m TestMessage2", and "mosquitto_pub -h localhost -t TestMyMQTT -m TestMessage3". A cursor is visible on the line following the last message.

```
pi@HlotLabPi1:~
File Edit Tabs Help
pi@HlotLabPi1:~ $ mosquitto_pub -h localhost -t TestMyMQTT -m TestMessage1
pi@HlotLabPi1:~ $ mosquitto_pub -h localhost -t TestMyMQTT -m TestMessage2
pi@HlotLabPi1:~ $ mosquitto_pub -h localhost -t TestMyMQTT -m TestMessage3
pi@HlotLabPi1:~ $
```

Figure 11- Publishing of Test Messages to a Topic “TestMyMQTT”

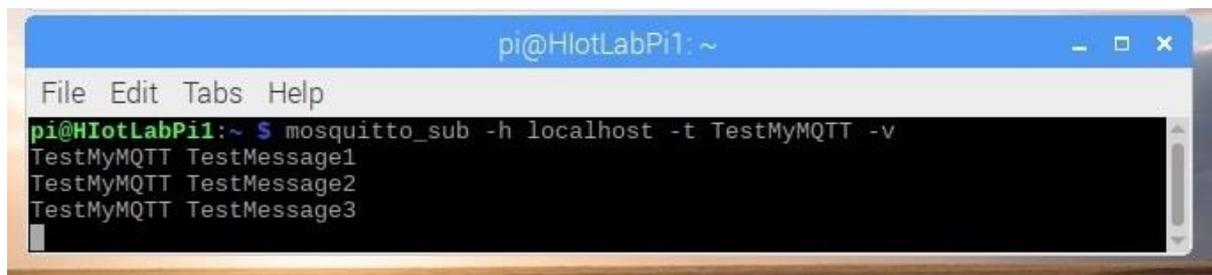
The syntax of the command to subscribe to the topic:

```
mosquitto_sub -h "name of the host" -t "topic name" -v
```

Steps to follow:

- Open a terminal window
- Type `mosquitto_sub -h localhost -t TestMyMQTT -v`
- Press "Enter"

Figure 12 shows the subscribing to the topic "TestMyMQTT" to receive the test messages



```
pi@HlotLabPi1: ~  
File Edit Tabs Help  
pi@HlotLabPi1:~ $ mosquitto_sub -h localhost -t TestMyMQTT -v  
TestMyMQTT TestMessage1  
TestMyMQTT TestMessage2  
TestMyMQTT TestMessage3
```

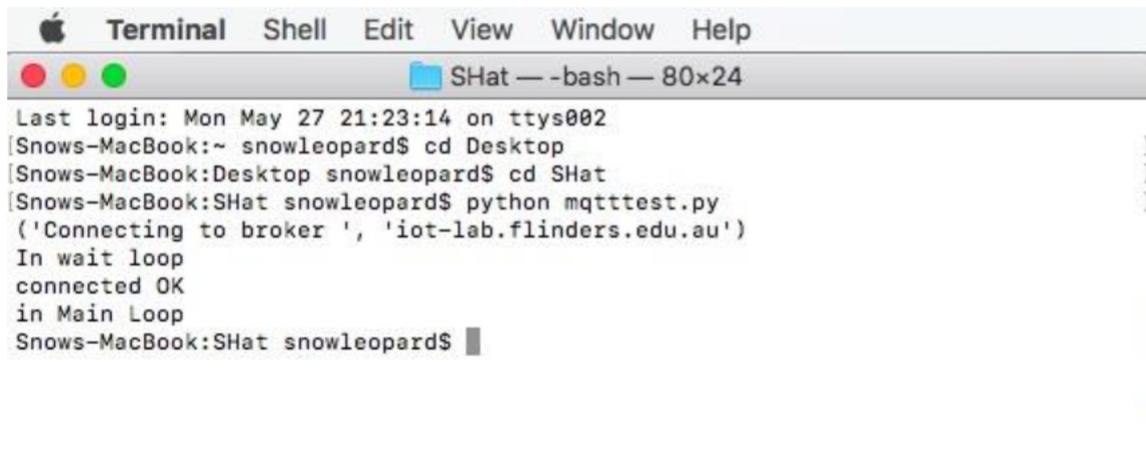
Figure 12- Subscribing to the topic "TestMyMQTT" to receive the test messages

After successfully testing the MQTT protocol on Mac OS and Raspberry Pi, the next step is to test the connection between client devices (Mac OS and Raspberry Pi) and the Cisco Kinetic MQTT.

e) Testing the connection between Mac OS and the Cisco Kinetic MQTT Broker

A Python program was written and used to test the connection. When this code is executed in a terminal window, a message prints on the command window to indicate the connection is success or not.

Figure 13 shows the successful connection to the MQTT broker in Cisco Kinetic from the Mac OS.



```
Terminal Shell Edit View Window Help
SHat — -bash — 80x24
Last login: Mon May 27 21:23:14 on ttys002
[Snows-MacBook:~ snowleopard$ cd Desktop
[Snows-MacBook:Desktop snowleopard$ cd SHat
[Snows-MacBook:SHat snowleopard$ python mqttest.py
('Connecting to broker ', 'iot-lab.flinders.edu.au')
In wait loop
connected OK
in Main Loop
[Snows-MacBook:SHat snowleopard$
```

Figure 13-Successful Connection to the MQTT broker in Cisco Kinetic from the Mac OS

Figure 14 shows the unsuccessful connection to the MQTT broker in Cisco Kinetic from the Mac OS.



```
SHat — -bash — 122x17
connected OK
in Main Loop
[Snows-MacBook:SHat snowleopard$ clear

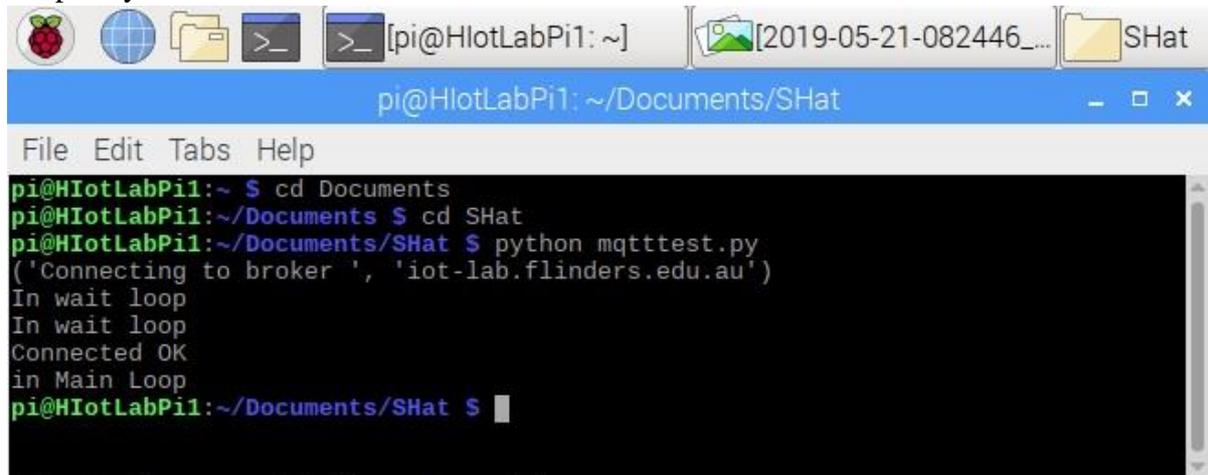
[Snows-MacBook:SHat snowleopard$ python mqttest.py
('Connecting to broker ', 'iot-lab.flinders.edu.au')
Traceback (most recent call last):
  File "mqttest.py", line 27, in <module>
    client.connect(broker, port) #connect to broker
  File "/Library/Python/2.7/site-packages/paho/mqtt/client.py", line 839, in connect
    return self.reconnect()
  File "/Library/Python/2.7/site-packages/paho/mqtt/client.py", line 962, in reconnect
    sock = socket.create_connection((self._host, self._port), source_address=(self._bind_address, 0))
  File "/System/Library/Frameworks/Python.framework/Versions/2.7/lib/python2.7/socket.py", line 575, in create_connection
    raise err
socket.error: [Errno 61] Connection refused
[Snows-MacBook:SHat snowleopard$
```

Figure 14 - Unsuccessful Connection to the MQTT broker in Cisco Kinetic from the Mac OS

f) Testing the connection between Raspberry Pi and the Cisco Kinetic MQTT Broker

The same Python program was used on the Raspberry Pi to test the connection.

Figure 15 shows the successful connection to the MQTT broker in Cisco Kinetic from the Raspberry Pi.

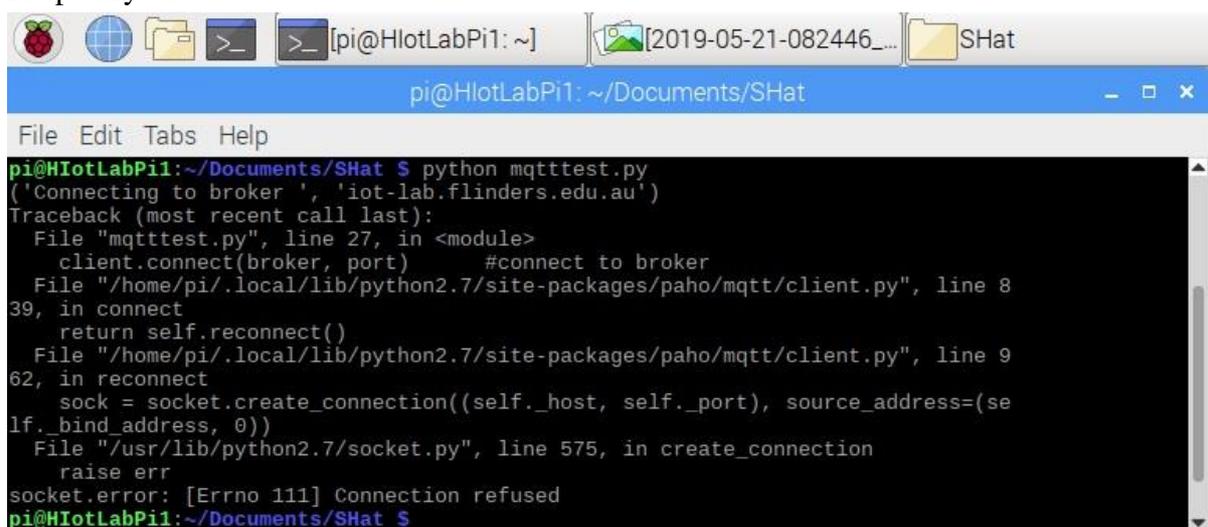


```
pi@HlotLabPi1: ~] [2019-05-21-082446_... SHat
pi@HlotLabPi1: ~/Documents/SHat
File Edit Tabs Help
pi@HlotLabPi1:~ $ cd Documents
pi@HlotLabPi1:~/Documents $ cd SHat
pi@HlotLabPi1:~/Documents/SHat $ python mqtttest.py
('Connecting to broker ', 'iot-lab.flinders.edu.au')
In wait loop
In wait loop
Connected OK
in Main Loop
pi@HlotLabPi1:~/Documents/SHat $
```

Figure 15- Successful Connection to the MQTT broker in Cisco Kinetic from the Raspberry Pi

To check an unsuccessful connection, the MQTT link on the Kinetic platform was stopped and then the Python program was executed again to see the results.

Figure 16 shows the unsuccessful connection to the MQTT broker in Cisco Kinetic from the Raspberry Pi.



```
pi@HlotLabPi1: ~] [2019-05-21-082446_... SHat
pi@HlotLabPi1: ~/Documents/SHat
File Edit Tabs Help
pi@HlotLabPi1:~/Documents/SHat $ python mqtttest.py
('Connecting to broker ', 'iot-lab.flinders.edu.au')
Traceback (most recent call last):
  File "mqtttest.py", line 27, in <module>
    client.connect(broker, port) #connect to broker
  File "/home/pi/.local/lib/python2.7/site-packages/paho/mqtt/client.py", line 839, in connect
    return self.reconnect()
  File "/home/pi/.local/lib/python2.7/site-packages/paho/mqtt/client.py", line 962, in reconnect
    sock = socket.create_connection((self._host, self._port), source_address=(self._bind_address, 0))
  File "/usr/lib/python2.7/socket.py", line 575, in create_connection
    raise err
socket.error: [Errno 111] Connection refused
pi@HlotLabPi1:~/Documents/SHat $
```

Figure 16 - Unsuccessful Connection to the MQTT broker in Cisco Kinetic from the Raspberry Pi

g) Configuring Cisco Kinetic MQTT to Publish and Subscribe

Any use of Kinetic requires the setup of some relevant communication protocol. In this project, two MQTT servers were created. One server for the Suunto 9 Smart watch data and the other for the Raspberry Pi SenseHat data. Only one topic per server was published but it is possible to have any number of topics published to one MQTT server. The main reason behind to create two separate servers: SenseHat, Smart Watch were to test the data streams separately and to identify any inconsistencies throughout the flow.

Figure 17 Shows the MQTT Server structure and the topics.

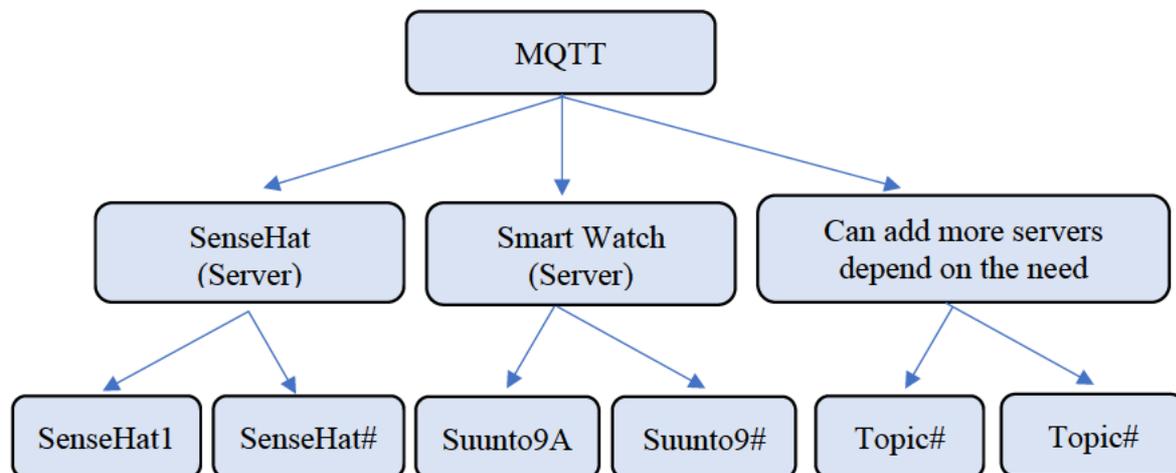


Figure 17 – MQTT Server and the topic structure

From the Kinetic Dataflow Editor, is used to create the mqtt servers. Right click on “mqtt”, select “Add Server” and fill in the following details accordingly and click “invoke”.

name: MQTT-SenseHat
url: tcp://localhost:1883
username: leave blank
password: leave blank
clientId: SenseHat1
qos: 1

Figure 18 shows the process of adding an MQTT server for “SenseHat”.

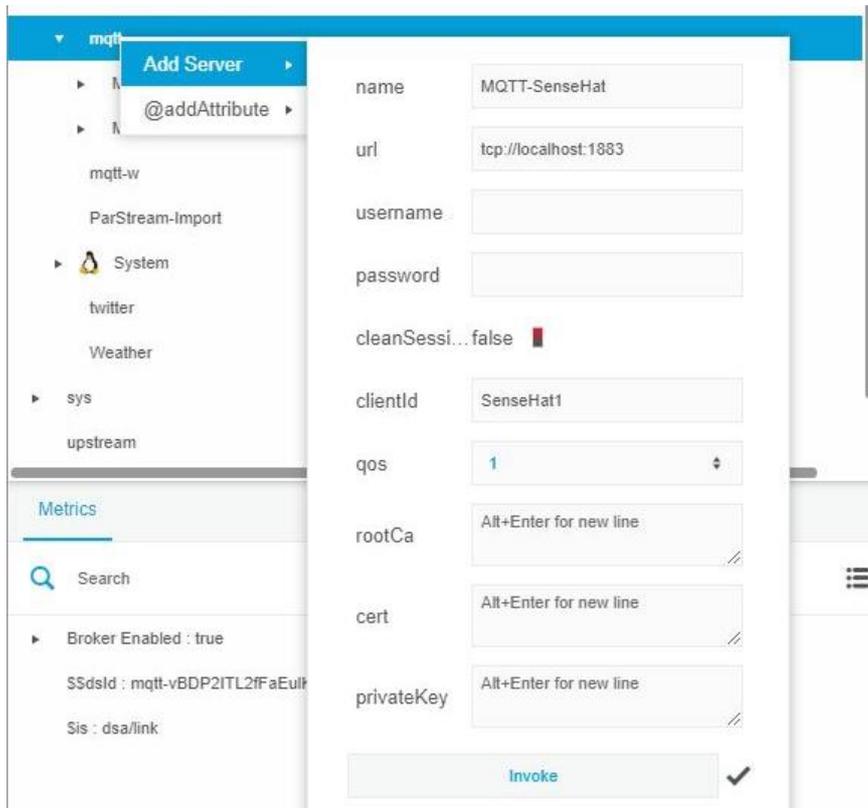


Figure 18- Process of Adding a Sub Server for “SenseHat”

To add an MQTT server for the Suunto 9 Smart Watch, from the Dataflow Editor, right click on “mqtt”, select “Add Server” and fill in the following details accordingly and click “invoke”.

name: MQTT-SmartWatch

url: tcp://localhost:1883

username: leave blank

password: leave blank

clientId: Suunto9A

qos: 1

Figure 19 shows the process of adding an MQTT server for “Smart Watch”.

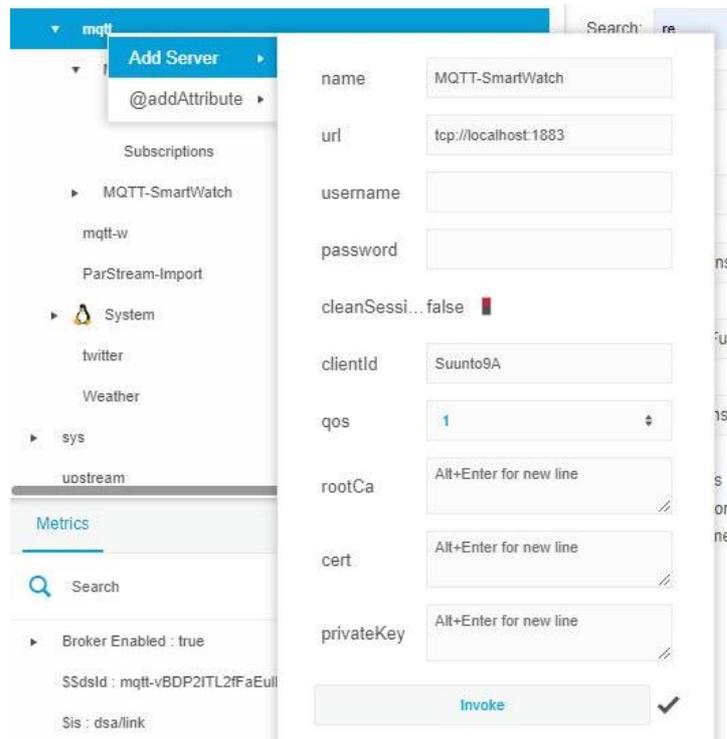


Figure 19 - Process of Adding a Sub Server for “Smart Watch”

Once the MQTT servers and topics (also called channels) were created for each device, the next step is to configure the “Subscriptions”. Expand the “MQTT-SenseHat” server, right click on “Subscriptions” and fill in the following details accordingly and click “invoke”.

name: Sensors

topic: SenseHat1/#

Figure 20 shows the process of subscribing a topic to “SenseHat”.

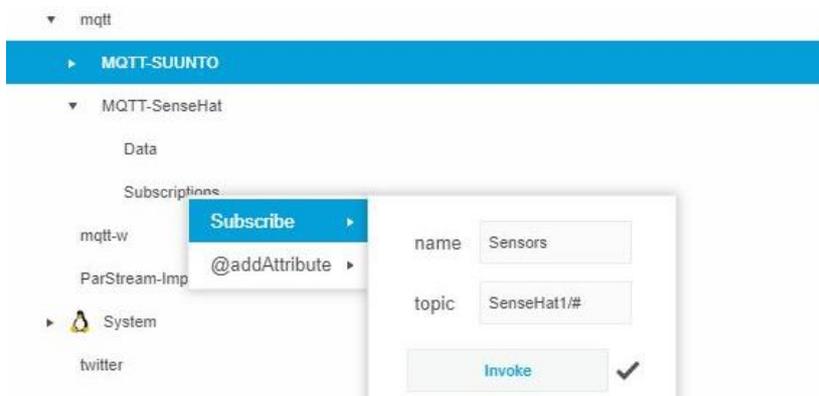


Figure 20 - Process of Subscribe a Topic to the “SenseHat”

To create a Subscription to the Smart Watch, expand the “MQTT-SmartWatch” server, right click on “Subscriptions” and fill the following details accordingly and click “invoke”.

name: SmartWatch

topic: Suunto9A

Figure 21 shows the process of subscribe a topic to “Smart Watch”.

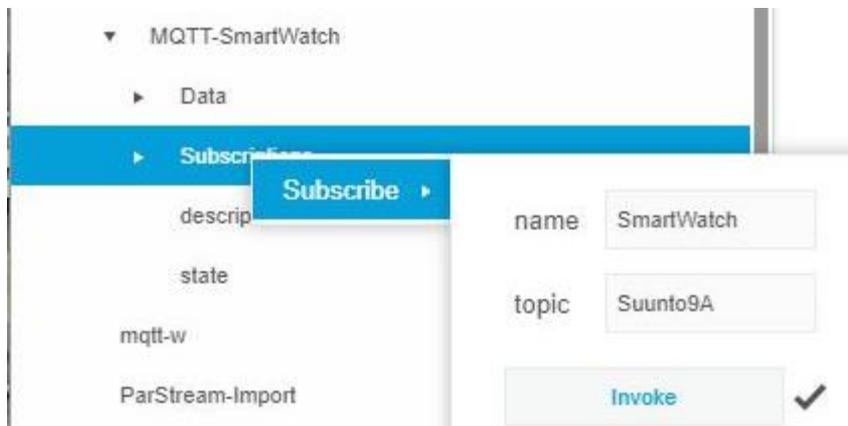


Figure 21- Process of Subscribe a Topic to the “Smart Watch”

Once the “Subscriptions” are configured, the next step is to create two separate folders for each device and create the “dataflow” within those folders.

To create a folder for the Smart Watch, right click on the “Dataflow”, select “Create Folder” type the name as “SmartWatch”, then click “Invoke”. Figure 22 shows the process of creating a folder for “Smart Watch”.

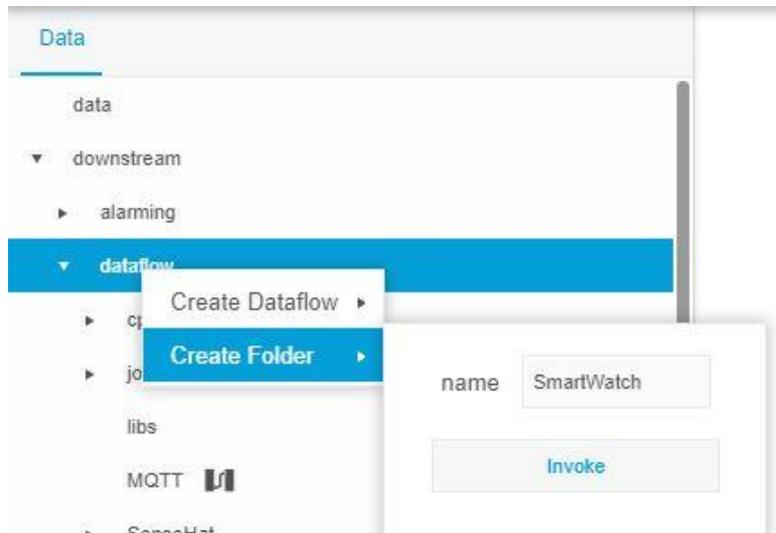


Figure 22- Process of Creating a Folder for “Smart Watch”

To create a folder for the SenseHat, right click on the “Dataflow”, select “Create Folder” type the name as “SenseHat”, then click “Invoke”.

Figure 23 shows the process of creating a folder for “SenseHat”.

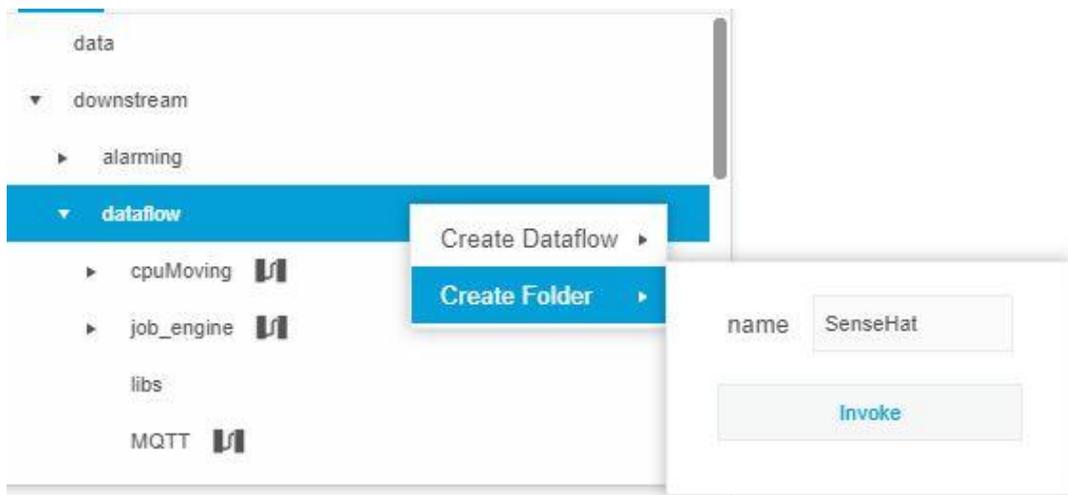


Figure 23- Process of Creating a Folder for “SenseHat”

Once the folders were created, the next step is to create a separate “Dataflow” for each device. To create a “Dataflow”, select the folder we created from the previous step, right click and select “Create Dataflow”, type the name as “SenseHat1”, click “Invoke”.

Figure 24 shows the process of adding a dataflow “SenseHat1”.

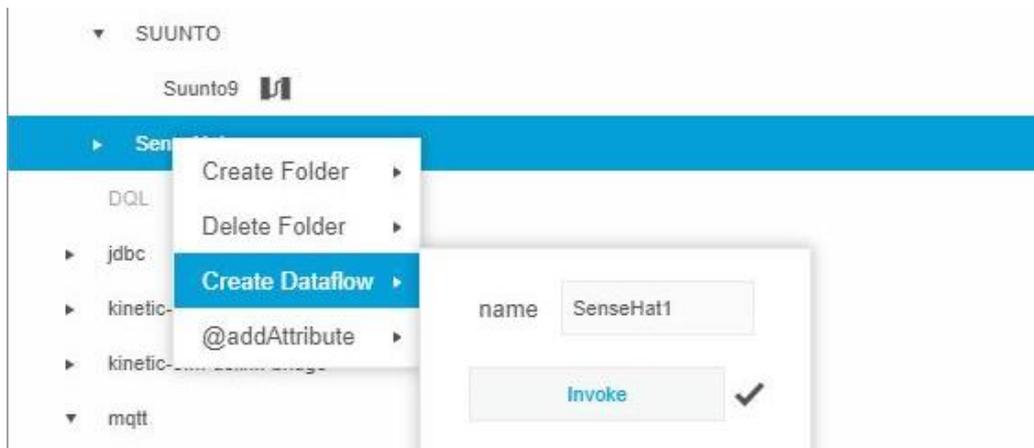


Figure 24– Process of Adding a Dataflow for “SenseHat”

To create a “Dataflow” for the Smart Watch, select the “SmartWatch” folder we created from the previous step, right click and select “Create Dataflow”, type the name as “Suunto9”, click “Invoke”.

Figure 25 shows the process of adding a dataflow “Smart Watch”.

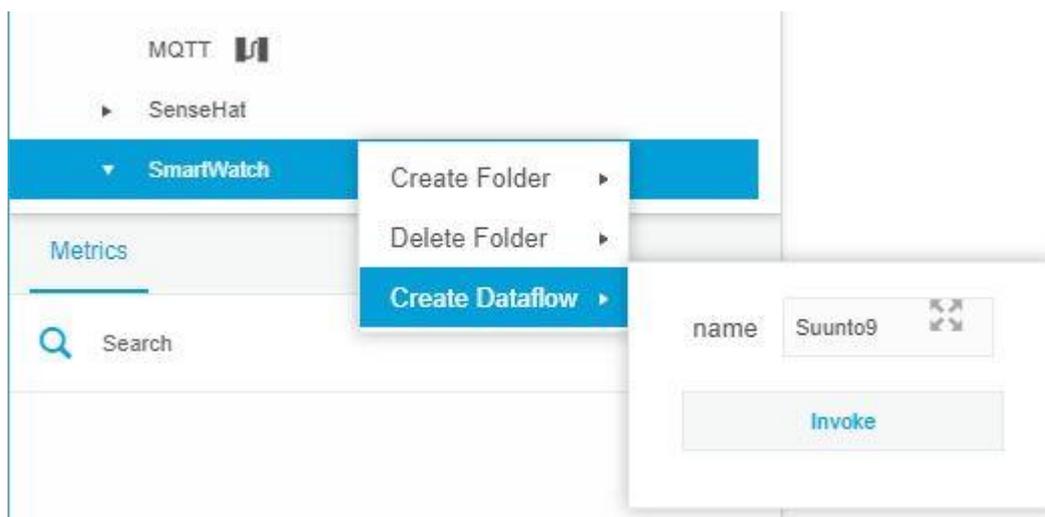


Figure 25- Process of Adding a Dataflow for “Smart Watch”

h) Data capture from the Devices

There are various data capture methods that exist for IoT devices. The data capturing from the devices is based on the type of device, data generating mechanism, nature of the data and how it produced the data. The following sections describe the process of data capturing from selected devices are explained.

Raspberry Pi SenseHat

Using the “Python” programming language, a program is written and used to read the data which is detected by the sensors which are inbuilt to the SenseHat electronic board. For this study, three sensors were selected to capture data: Humidity, Pressure and Temperature.

Figure 26 shows a sample of a “JSON” data file for sensor reading from the SenseHat which created using the Python programme.

```
{
  "SH_Sensors": [
    {
      "Humidity": 23.0,
      "Pressure": 1023.0,
      "Temperature": 36.0,
      "Time": "2019-05-21T10:58:41.052000"
    }
  ]
}{
  "SH_Sensors": [
    {
      "Humidity": 27.0,
      "Pressure": 1023.0,
      "Temperature": 36.0,
      "Time": "2019-05-21T10:58:44.055685"
    }
  ]
}{
  "SH_Sensors": [
    {
      "Humidity": 27.0,
      "Pressure": 1023.0,
      "Temperature": 36.0,
      "Time": "2019-05-21T10:58:47.060296"
    }
  ]
}{
  "SH_Sensors": [
    {
      "Humidity": 26.0,
      "Pressure": 1023.0,
      "Temperature": 36.0,
      "Time": "2019-05-21T10:58:50.065264"
    }
  ]
}
```

Figure 26- Sample “JSON” data file for sensor reading from the SenseHat

Further Python programming code can be modified to publish the sensor reading data to the MQTT topic in the Cisco Kinetic platform.

Smart Watch

Select the mode of the exercise from the Smart Watch, i.e. - Basic Walk, Basic Running, Trail running, Cycling etc. Start the activity. Once the activity is finished, stop the activity from the watch. Then the watch will save the activity data to the watch itself. Figure 27 shows the steps involved in start to end of an activity.



Figure 27– Start to End of an Activity

Usually the watch can be connected with the mobile phone via Bluetooth or laptop/desktop PC via a USB cable. Once you open the “Suunto” app from the mobile phone or the watch connected to laptop, the activity data will sync between the watch and the phone. The data files are stored with the extension of “fit” which is a file type commonly used in smart fitness devices. Using “Python” programming language, a program is written and used to read the “fit” data file and then convert it to a JSON file format which can be used as an input. Figure 28 shows a sample “JSON” data files for running on a treadmill, walking and weight training.

```
Treadmill
{
  "record": [
    {
      "timestamp",
      924080330
    },
    {
      "altitude",
      3101
    },
    {
      "heart_rate",
      64
    },
    {
      "cadence",
      40
    },
    {
      "distance",
      200
    }
  ]
},
{
  "record": [
    {
      "timestamp",
      924080331
    },
    {
      "altitude",
      3101
    },
    {
      "heart_rate",
      64
    },
    {
      "cadence",
      40
    }
  ]
}

Weight Training
{
  "record": [
    {
      "timestamp",
      924593618
    },
    {
      "altitude",
      3380
    },
    {
      "heart_rate",
      71
    },
    {
      "temperature",
      28
    }
  ]
},
{
  "record": [
    {
      "timestamp",
      924593619
    },
    {
      "altitude",
      3380
    },
    {
      "heart_rate",
      71
    },
    {
      "temperature",
      28
    }
  ]
}

Walking
{
  "record": [
    {
      "timestamp",
      923369777
    },
    {
      "altitude",
      2693
    },
    {
      "heart_rate",
      69
    },
    {
      "vertical_speed",
      0
    }
  ]
},
{
  "record": [
    {
      "timestamp",
      923369778
    },
    {
      "altitude",
      2693
    },
    {
      "heart_rate",
      69
    },
    {
      "vertical_speed",
      0
    }
  ]
}
}
```

Figure 28– Sample “JSON” Data File for Treadmill, Walking and Weight Training

4.1. Demonstration

The JSON file created from the smart watch reading and the sensor reading from the SenseHat can feed in to the Message Queuing Telemetry Transport (MQTT) in the Kinetic platform. The data is published to the topics that were created in Figure 19 and Figure 20.

Figure 29 shows the dataflow map from the sensing devices to Kinetic platform.

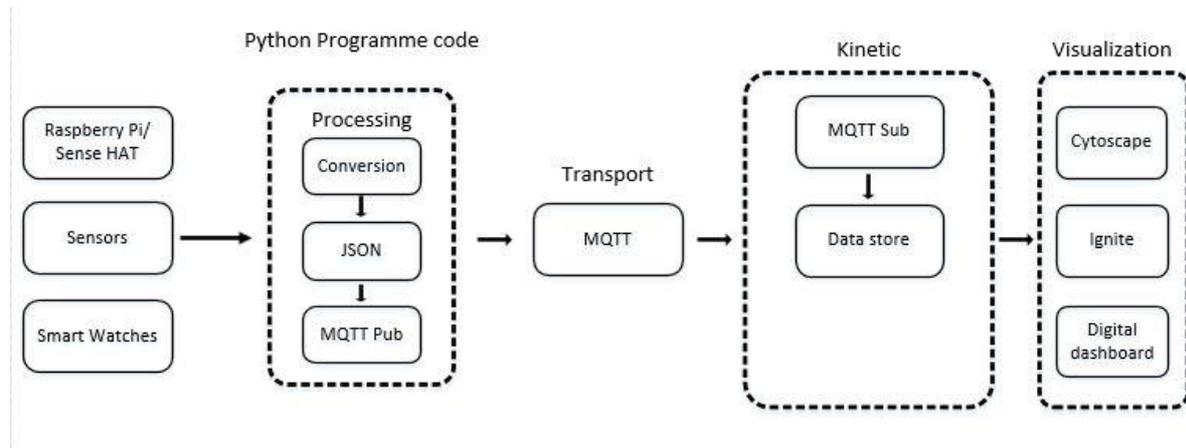


Figure 29– Dataflow Map from a Sensing Devices to Kinetic Platform

a) Publishing Data from Raspberry Pi to Kinetic MQTT Topic “SenseHat1”

A Python program was written and used to read the sensors in the SenseHat device and to publish to the topic “SenseHat1”. Figure 30 shows the data reading from the Raspberry Pi SenseHat sensors using the Python programming code.

```
pub_sh.py x
pi@HlotLabPi1: ~/Documents/SHat
File Edit Tabs Help
pi@HlotLabPi1:~ $ cd Documents
pi@HlotLabPi1:~/Documents $ cd SHat
pi@HlotLabPi1:~/Documents/SHat $ chmod +x pub_sh.py
pi@HlotLabPi1:~/Documents/SHat $ python pub_sh.py
['28-05-2019 09:48:01', {'Temperature': 36.0}, {'Pressure': 1017.0}, {'Humidity': 31.0}]
Connected with result code0
['28-05-2019 09:48:11', {'Temperature': 36.0}, {'Pressure': 1017.0}, {'Humidity': 31.0}]
['28-05-2019 09:48:21', {'Temperature': 36.0}, {'Pressure': 1017.0}, {'Humidity': 31.0}]
['28-05-2019 09:48:31', {'Temperature': 36.0}, {'Pressure': 1017.0}, {'Humidity': 32.0}]
['28-05-2019 09:48:41', {'Temperature': 36.0}, {'Pressure': 1017.0}, {'Humidity': 32.0}]
['28-05-2019 09:48:51', {'Temperature': 36.0}, {'Pressure': 1017.0}, {'Humidity': 32.0}]
['28-05-2019 09:49:01', {'Temperature': 36.0}, {'Pressure': 1017.0}, {'Humidity': 32.0}]
['28-05-2019 09:49:11', {'Temperature': 36.0}, {'Pressure': 1017.0}, {'Humidity': 31.0}]
['28-05-2019 09:49:21', {'Temperature': 36.0}, {'Pressure': 1017.0}, {'Humidity': 31.0}]
['28-05-2019 09:49:31', {'Temperature': 36.0}, {'Pressure': 1017.0}, {'Humidity': 32.0}]
['28-05-2019 09:49:41', {'Temperature': 36.0}, {'Pressure': 1017.0}, {'Humidity': 31.0}]
['28-05-2019 09:49:51', {'Temperature': 36.0}, {'Pressure': 1017.0}, {'Humidity': 31.0}]
```

Figure 30- Data Reading from the Raspberry Pi SenseHat Sensors

Figure 31 shows the data reading from the Raspberry Pi SenseHat sensors published to the MQTT topic “SenseHat1” using the Python programming code.

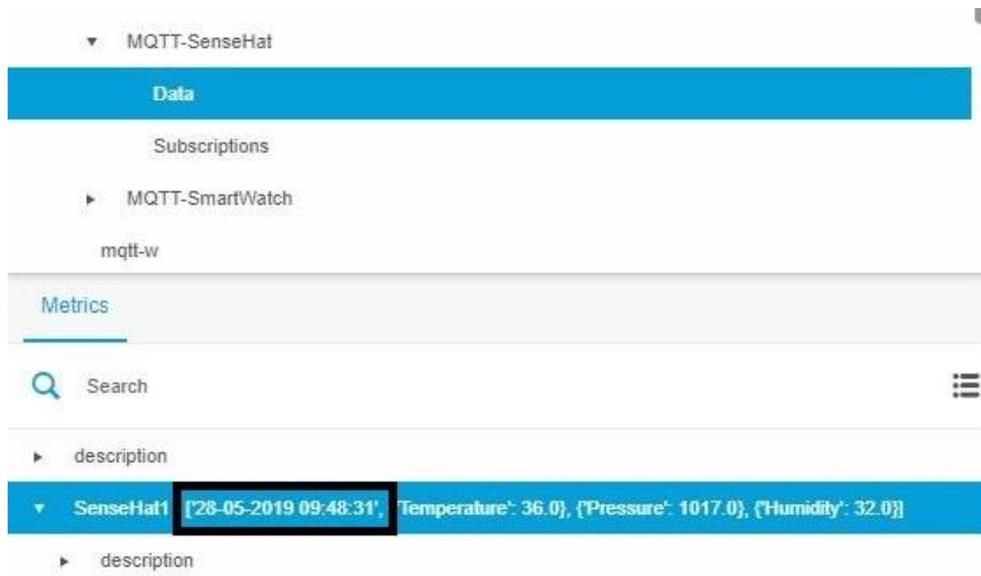


Figure 31- Data Reading from the Raspberry Pi SenseHat Sensors Published to the MQTT Topic “Sensehat1”

As we can see the highlighted areas of Figure 30 and Figure 31 with the timestamp record demonstrates that the data are published.

b) Publishing Data from Mac OS to Kinetic MQTT Topic “Suunto9A”

A Python program was written and used to read the “JSON” file created from the Smart Watch reading and for publishing the data to the topic “Suunto9A”.

Figure 32 shows the data reading from the JSON file that was produced from the Suunto 9 Smart Watch using the Python programming code.

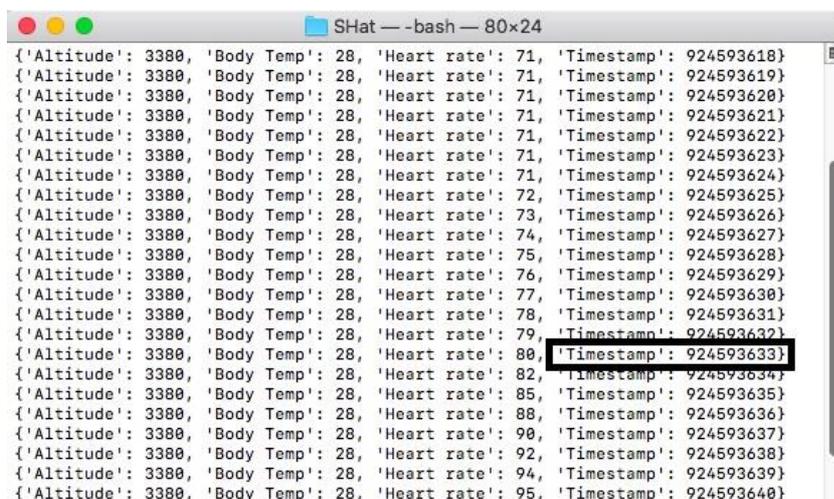


Figure 32- Reading Data from JSON file that was produced from the Suunto 9 Smart Watch

Figure 33 shows the data reading JSON file published to the MQTT topic “Suunto9A” using the Python program.

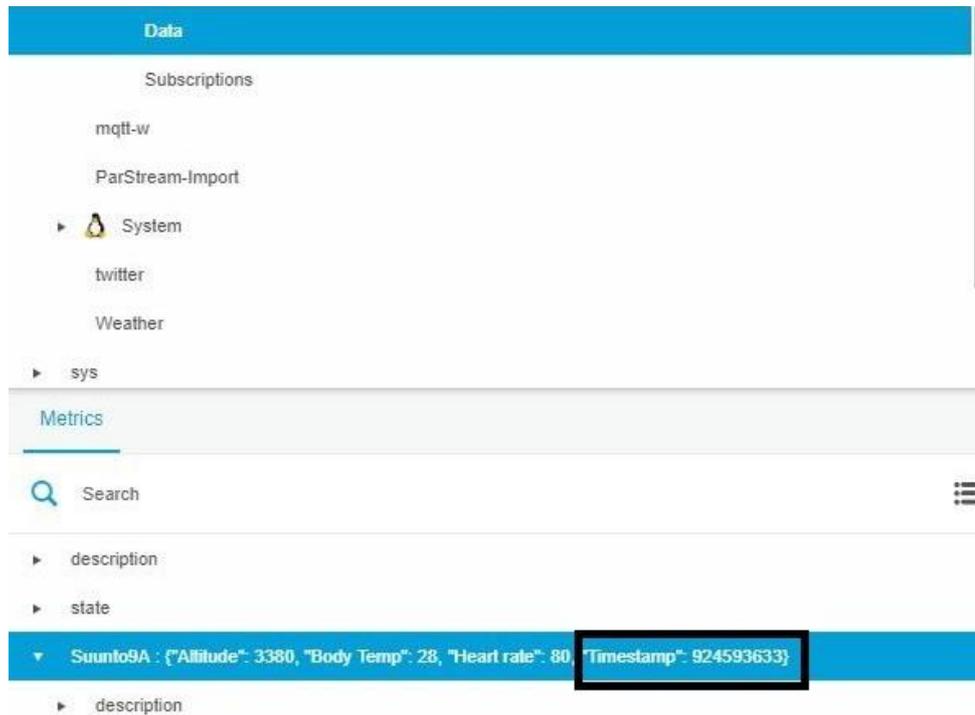


Figure 33- Data Reading from the JSON file Published to the MQTT Topic “Suunto9A”

As we can see from the highlighted areas of Figure 32 and Figure 33 with the timestamp record, the data are published to the “Suunto9A” topic as expected.

4.2. Evaluation

While it is an essential part of the design science process, the evaluation of artefacts can be problematic as there is no consensus on how this evaluation should be performed (Peppers et al., 2012). As the artefact type in this research is a ‘construct’ of a dataflow that has been constructed from a set of experimental steps, one method of evaluation, described by Zhang et al (1992) is using transparent evaluation, reflection and learning along the selected design process. In this research, this evaluation and cross checking with reflection using the two different devices to model the data source flow was used. In addition, another evaluation method is technical experimentation as to evaluate the performance or correctness of the artefact to assess the validity of the constructs used. This experimentation was undertaken to test the functional performance using real-world data.

4.3. Communication

Consistent with the intention of the communications step of the design science methodology, the research is documented in this thesis and will be used in future Flinders Digital Health IoT Lab projects as a reference for using the Kinetic platform. The dataflow will be used in a forthcoming workshop about the Campus Mental Wellness project to help participants understand the platform and its potential uses. Further, the results of this project will form part of an academic paper on Health IoT use.

5. Discussion

Selecting the IoT devices: Suunto 9 Baro and Raspberry Pi SenseHat for this study based on the capability of detecting and recording important data which need to conduct this study. The main purpose of using a Smart Watch was to capture data based on physical activities and the Raspberry Pi SenseHat to capture environmental data which can be adapted to improve or control the climate of healthcare or smart home environments.

A connection protocol is needed to establish a connection between IoT and the Kinetic Platform. As an ISO standard for the connectivity protocol, MQTT was used. MQTT was highly successful as the communication protocol between the devices and the Kinetic platform. Python language is used to develop the necessary codes to read data from devices, write data to “JSON” format and publish the data to channels in the Kinetic platform. Continuous data streams are published with pre-defined intervals to the Kinetic platform to bring the test results to the next level.

An end-to-end dataflow can be modelled incorporating Cisco Kinetic platform and the Internet of Health Things. Selecting the devices to conduct this study, which data to capture from devices, how to read the data from the devices, formatting of the data output from devices, interpretation of the data, establishing connections between the devices and the Kinetic platform and testing the connections: these tasks were all challenging. Certain technical skills were required and some had to be developed along the way. A decision was made to use a Smart Watch and a Raspberry Pi SenseHat because of the ability to detect and collect various data. The Smart Watch originally produced the data output file in a “fit” format. This data needed to be converted to a “JSON” format for use as an input. The Python programming language was used to do the conversion from “fit” to “JSON”. On the other hand, it was possible to read the data directly from the sensors in the Raspberry Pi SenseHat and publish them directly to the relevant topic in the Kinetic platform using a Python program. No conversion of data was required for this step.

The greatest problem during this study was to establish and test the connections between IoT devices and the Cisco Kinetic platform. In part, the challenge was learning how to setup the MQTT server in the Kinetic platform. The literature and modern web discussions suggested the development of a Python program to test the connections.

For future studies, this type of a programming code can be used to test the connections between the device and the platform before publishing the data streams.

In this study, two separate MQTT sub servers were deployed in the Kinetic platform: “MQTT-SenseHat” and “MQTT-SmartWatch”. This is to test separately each data stream published by the relevant device. But it is possible to achieve the same results by using a single server and creating multiple topics in the one MQTT server.

To develop the dataflow map, initially interaction points and key elements needed to be identified. In this study, as previously explained, the first step was to choose the IoT devices and data types to capture. The next step was to investigate how data can be captured from the IoT devices, in which format and how to read or store the data. Then it had to be determined how the data can be transferred. The study, location was limited to the University. The dataflow map that was developed during this study is illustrated in the Figure 29 (Dataflow Map from a Sensing Devices to Kinetic Platform).

In this thesis, visualization is related to two different contexts: visualization of the dataflow in the Kinetic platform and the visualization of the “data” using a visualization module (e.g. Ignite, Cytoscape). Moving to the next level, this data can be transferred to a database and/or to the visualization module to produce graphs or patterns. The visualization of the data that is published to the Kinetic platform is the subject of future research.

Visualization of the dataflow is illustrated. Once data is published to the pre-defined MQTT channels, the data stream can be visualized in a table format as shown in figure 34 and figure 35. The Kinetic platform provides computational power to produce the right data for the correct applications at the correct time. The Kinetic platform does not itself provide data analytics in a visual form. At best, it provides for computations and the transportation of resulting computations to other platforms that are capable of data visualizations.

Many of the challenges in developing the proof-of-concept relate to a lack of “hands on” control over the technology. The work involved communicating with and relying on other parties to provide accurate information or to resolve technical issues encountered. The turnaround time for solving issues could be 24 to 48 hours. This in turn, affected the time involved and available for testing and achieving the proof-of-concept.

Implementation challenges – Lack of control over the infrastructure: Connecting IoT devices to the campus network is not a straight forward exercise. It involved many steps. As a starting point, devices need to be registered in the campus network, obtaining a valid username/password. Initially devices were not able to connect to Kinetic platform through wireless due to the security rules implemented by the university. The facility was available only through the wired network. This was a challenge as the sensors need to be positioned in a variety of different locations. This was eventually resolved.

Technological challenges –Ensuring the connection between IoT devices and the Kinetic platform. This type of testing is essential to identify any connection barriers to data transmission. This will help to make sure the data transferred to the platform is not lost in between. The Kinetic platform version was upgraded due to an instability. Still a bug exists on the system which consumes the system memory and leads to a periodic system restart. Due to limited user permissions, this was something that a third party had to complete on request. There was also little documentation and no technical support available in relation to the use of Kinetic. Learning how to set up and use the MQTT server on Kinetic involved searching for and reading or watching a variety of internet sources along with a good deal of trial and error.

Design challenges – Need to learn python programming, as this was used to read data from sensors and write or transfer to communication channels in the Kinetic platform. There was no prior experience with Python. Fortunately there are many useful internet resources on this subject which facilitated rapid learning.

Visualization of the actual data transmitted or computed by Kinetic requires the use of separate modules. In this case, Ignite a popular platform used in Australian health environments was made available. A virtual server was commissioned to run Ignite in preparation for data visualization. There is no user documentation available for Ignite. All setup is conducted by the service provider. Furthermore, it was later learned that it requires use of the Kinetic DCM module to retrieve data from Kinetic. This is a module that is not available to the University.

Future work

As mentioned in the discussion, Ignite module can be used to visualize the data. Establishing the communication between the Kinetic platform and the Ignite module is work for the future. Any other wearable and ambient sensors discussed in Table 1 can be used to capture data and to use for various studies using the framework.

6. Conclusion

The Internet of Things (IoT) is a widely used concept across many industries, as it can be used to collect a wide variety of real time data from people and environments using many devices. This research study aimed to find out how Health IoT data can be extracted, manipulated and moved to various applications depending on the output requirements. By using two IoT devices: a Raspberry Pi SenseHat and a Smart Watch “Suunto 9 Baro”, the MQTT communication protocol and the Python programming language, the end-to-end demonstration of health IoT data capture, transfer, manipulation, feedback and visualization using “Cisco Kinetic” platform was successfully addressed.

The dataflow model developed during the study was methodically tested to prove the accuracy and it was found that it was consistent throughout the testing. Therefore, constructing an end-to-end dataflow model that upstream data from IoT devices to the Cisco Kinetic Platform was successful.

By addressing the research questions, the overall objective of constructing a framework for health IoT end-to-end process and establishing a proof-of-concept using “Cisco Kinetic” for health sector was successfully achieved.

7. References

Atzori, L., Iera, A. & Morabito, G. 2011. Siot: Giving A Social Structure To The Internet Of Things. *Ieee Communications Letters*, 15, 1193-1195.

Bonomi, F., Milito, R., Natarajan, P. & Zhu, J. 2014. Fog Computing: A Platform For Internet Of Things And Analytics. In: Bessis, N. & Dobre, C. (Eds.) *Big Data And Internet Of Things: A Roadmap For Smart Environments*. Cham: Springer International Publishing.

Bradley, D., Russell, D., Ferguson, I., Isaacs, J., Macleod, A. & White, R. 2015. The Internet Of Things – The Future Or The End Of Mechatronics. *Mechatronics*, 27, 57-74.

Breivold, H. P. & Sandström, K. Internet of Things for Industrial Automation -- Challenges and Technical Solutions. 2015 IEEE International Conference on Data Science and Data Intensive Systems, 11-13 Dec. 2015 2015. 532-539.

Carlsson, S. A. (2005). Developing Information Systems Design Knowledge: A Critical Realist Perspective. *Electronic Journal of Business Research Methods*, 3(2), 93-102.

Cisco Kinetic Overview. (2017). Retrieved From <https://www.cisco.com/c/dam/en/us/solutions/collateral/internet-of-things/kinetic-at-a-glance.pdf>

Cisco.Com. (2019). [Online] Available At: <https://www.cisco.com/c/dam/en/us/solutions/collateral/internet-of-things/kinetic-oil-gas.pdf> [Accessed 21 Mar. 2019].

Cisco.Com. (2019). [Online] Available At: <https://www.cisco.com/c/dam/en/us/solutions/collateral/internet-of-things/cisco-kinetic-mfg-at-a-glance.pdf> [Accessed 21 Mar. 2019].

Cisco. (2019). *Cisco Digital Retail Solutions*. [Online] Available At: <https://www.cisco.com/c/en/us/solutions/industries/retail.html?Dtid=Osscdc000283> [Accessed 21 Mar. 2019].

Cisco. (2019). *Cisco Kinetic For Cities*. [Online] Available At: <https://www.cisco.com/c/en/us/solutions/industries/smart-connected-communities/kinetic-for-cities.html?Dtid=Osscdc000283> [Accessed 21 Mar. 2019].

Da Costa, C. A., Pasluosta, C. F., Eskofier, B., Da Silva, D. B. & Da Rosa Righi, R. 2018. Internet Of Health Things: Toward Intelligent Vital Signs Monitoring In Hospital Wards. *Artificial Intelligence In Medicine*, 89, 61-69.

Eclipse Mosquitto. (2018). *Eclipse Mosquitto*. [Online] Available At: <https://mosquitto.org/> [Accessed 10 May 2019].

Edureka. (2019). *Iot Applications | Internet of Things Examples | Real World Iot Examples | Edureka*. [Online] Available At: <https://www.edureka.co/blog/iot-applications/> [Accessed 27 Feb. 2019].

Forbes.Com. (2019). *Forbes Insights: Logistics 4.0: How IoT Is Transforming The Supply Chain*. [Online] Available At: <https://www.forbes.com/sites/insights-inteliot/2018/06/14/logistics-4-0-how-iot-is-transforming-the-supply-chain/#36da8096880f> [Accessed 27 Feb. 2019].

Gardašević, G., Veletić, M., Maletić, N., Vasiljević, D., Radusinović, I., Tomović, S. & Radonjić, M. 2017. The Iot Architectural Framework, Design Issues And Application Domains. *Wireless Personal Communications*, 92, 127-148.

Hassanalieragh, M., Page, A., Soyata, T., Sharma, G., Aktas, M., Mateos, G., Kantarci, B. & Andreescu, S. Health Monitoring And Management Using Internet-Of-Things (Iot) Sensing With Cloud-Based Processing: Opportunities And Challenges. 2015 Ieee International Conference On Services Computing, 27 June-2 July 2015 2015. 285-292.

Hevner, A. 2007. *A Three Cycle View Of Design Science Research*.

Hevner, A. R., March, S. T., Park, J. & Ram, S. 2004. Design Science In Information Systems Research. *Mis Quarterly*, 28, 75-105.

Hoey, B. (2019). *What Is Logistics 4.0?*. [Online] Blog.Flexis.Com. Available At: <https://blog.flexis.com/what-is-logistics-4.0>

Hobbs, A. (2018). Five Ways The Internet Of Things Is Transforming Businesses Today | Internet Of Business. Retrieved From <https://internetofbusiness.com/5-ways-the-internet-of-things-is-transforming-businesses-today/>

Iot-Analytics.Com. (2015). [Online] Available At: <http://iot-analytics.com/wp/wp-content/uploads/2016/01/white-paper-iot-platforms-the-central-backbone-for-the-internet-of-things-nov-2015-vfi5.pdf> [Accessed 27 Oct. 2018].

Iotworldtoday.Com. (2019). *11 Innovative Iot Use Cases*. [Online] Available At: <https://www.iiotworldtoday.com/2016/12/07/11-innovative-iiot-use-cases/> [Accessed 27 Feb. 2019].

Kaa Iot Platform. (2019). What Is the Internet of Things Platform - All about IoT Technology and Applications. [Online] Available At: <https://www.kaaproject.org/what-is-iiot/> [Accessed 27 Oct. 2018].

Leverage.Com. (2016). Simple Explanations - Connecting the Internet of Things. [Online] Available At: <https://www.leverage.com/blogpost/connecting-the-internet-of-things> [Accessed 27 Oct. 2018].

Mahdavinejad, M. S., Rezvan, M., Barekatin, M., Adibi, P., Barnaghi, P. & Sheth, A. P. 2018. Machine Learning For Internet Of Things Data Analysis: A Survey. *Digital Communications And Networks*, 4, 161-175.

Malek, Y. N., Kharbouch, A., Khoukhi, H. E., Bakhouya, M., Florio, V. D., Ouadghiri, D. E., Latre, S. & Blondia, C. 2017. On The Use Of Iot And Big Data Technologies For Real-Time Monitoring And Data Processing. *Procedia Computer Science*, 113, 429-434.

Mqtt.Org. (N.D.). *Mqtt*. [Online] Available At: [Http://Mqtt.Org/](http://Mqtt.Org/) [Accessed 27 Mar. 2019].

Peffers, K., Tuunanen, T., Rothenberger, M. & Chatterjee, S. 2007. A Design Science Research Methodology For Information Systems Research. *J. Manage. Inf. Syst.*, 24, 45-77.

Peffers, K., Rothenberger, M., Tuunanen, T., & Vaezi, R. (2012). *Design Science Research Evaluation* (Vol. 7286). Berlin, Heidelberg: Berlin, Heidelberg: Springer Berlin Heidelberg.

Press, G. (2017). 6 Hot Internet of Things (Iot) Security Technologies. [Online] Forbes.Com. Available At: <https://www.forbes.com/sites/gilpress/2017/03/20/6-hot-internet-of-things-iot-security-technologies/#781da6811b49> [Accessed 27 Oct. 2018].

Porter, M., & Heppelmann, J. (2014). *How Smart, Connected Products Are Transforming Competition*. Harvard Business Review.

Qi, J., Yang, P., Min, G., Amft, O., Dong, F. & Xu, L. 2017. Advanced Internet Of Things For Personalised Healthcare Systems: A Survey. *Pervasive And Mobile Computing*, 41, 132-149.

Ray, P. P. 2016. A Survey On Internet Of Things Architectures. *Journal Of King Saud University - Computer And Information Sciences*, 30, 291-319.

Sethi, P. & R. Sarangi, S. 2017. *Internet Of Things: Architectures, Protocols, And Applications*.

Sethi, P. & Sarangi, S. R. 2017. Internet Of Things: Architectures, Protocols, And Applications. *Journal Of Electrical And Computer Engineering*, 2017, 25.

Tao, F., Zuo, Y., Xu, L. D., Lv, L. & Zhang, L. 2014. Internet Of Things And Bom-Based Life Cycle Assessment Of Energy-Saving And Emission-Reduction Of Products. *Ieee Transactions On Industrial Informatics*, 10, 1252-1261.

Wu, M., Lu, T. J., Ling, F. Y., Sun, J. & Du, H. Y. Research On The Architecture Of Internet Of Things. *Icacte 2010 - 2010 3rd International Conference On Advanced Computer Theory And Engineering, Proceedings, 2010*. V5484-V5487.

Yu, W., Liang, F., He, X., Hatcher, W. G., Lu, C., Lin, J. & Yang, X. 2018. A Survey On The Edge Computing For The Internet Of Things. *Ieee Access*, 6, 6900-6919.

Zhang, X., Olfman, L., And Firpo, D. (2011). An Information Systems Design Theory For Collaborative Eportfolio Systems. In *Proceedings Of System Sciences (Hicss)*, 2011 44th Hawaii International Conference, 1-10.