Integrated Cloud Cockpit: Making Surveillance and Maintenance of Oil Pipelines Practicable

Garima Srivastava
SAP Labs India Pvt. Ltd.
garima.srivastava@sap.com

Yeshwant More
SAP Labs India Pvt. Ltd.
y.more@sap.com

Jenifer Sam
SAP Labs India Pvt. Ltd.
jenifer.sam@sap.com

***Abstract*—Building a secure platform to connect distributed products, software and hardware simplifies digitalization of businesses. This calls for platforms that can help companies ideate, prototype, validate and develop using cutting edge technologies such as Artificial Intelligence, Internet of Things, Block Chain, Big Data and Analytics.**

In this paper, we illustrate how such Integrated Cloud Cockpits (ICC) provide a seamless environment with services that facilitate development of next generation smart applications that are scalable, resilient and provide decreased price-performance ratio. We support this research by presenting an empirical study on how intelligent surveillance and predictive maintenance of oil pipelines is made practicable by employing collaborative cloud services. We have leveraged the services provided by Google Cloud Platform to show how existing cloud platforms can suffice the needs of a secure and fully featured enterprise use case.

I. INTRODUCTION

One of the prominent reasons why the software industry is shifting to cloud computing is the need for fast innovation. Software vendors should be able to provide their innovations to the customer for testing, learning and improvising at a rapid pace. For this purpose, Integrated Cloud Platforms are working like a one stop destination to provide services that ease the development and maintenance of such a new breed of next generation applications. These cloud platforms have a mix of diverse services that cover all aspects of software development. For instance, there are services that provide the infrastructure for development (development environment, database instance, authorization/authentication etc.) as well as there are technical and functional services like speech to text, blockchain service, big data management and other such services.

These Cloud Platforms suffice the needs of end to end development and also aid in solving business problems via intelligent software in the recent times. One such very tough business problem has always been maintenance and surveillance of oil and gas pipelines. There are multiple reasons why this was one of the most cumbersome scenarios to be handled via a software. As pipelines are spread across geographies and in extreme remote areas, it's nearly impossible to have human inspection planned in an optimized manner. Even to find the exact location of the failure requires a lot of time due to the vast spread of the pipeline. It's also not very easy to plan a maintenance of these pipelines on a short duration due to unavailability of experts at the required locations. In the below sections, we first explore the key characteristics and features of cloud platforms [1], and then we dive into the reference architecture, details and design of the case study.

II. KEY CHARACTERISTICS OF SERVICE BASED INTEGRATED CLOUD PLATFORMS

The preliminary meaning of software and hardware resources has been redefined by cloud computing in a way that both hardware as well as software is available as a service on the cloud platform. This helps in reducing the total time for development and ownership for software organizations.

Following are a few cloud platform design characteristics [4][7], that we were able to gather for the development of new generation connected applications:

A. Service based Offering

Service based offering of the resources is the key to reduce development and maintenance efforts as well as increases future scalability. For consumer, these services can also be purchased in any quantity at any time [2].

B. Efficient Use of New Development Paradigms

These Cloud Platforms should be able to provide all new technological paradigms as service offerings. For instance, there can be Artificial Intelligence services, Deep Learning services, IoT (Internet of Things) services, Block Chain as a service, as well as other technical services available that can be consumed on need basis.

C. Robust Integration Framework

To create well-organized smart applications that can read connected machines and devices as well as have the ability to connect to heterogenous systems, in-built Integrated services are required by software vendors. These services should be able to read data from devices or systems effectively and then transform it to be able to be use in consuming applications [8].

D. End to end Development Environment

The concept of Platform as a Service (PaaS) enables the users to create their application without investing efforts on the infrastructure. The Integrated Cloud Platforms like Google Cloud Platform or Microsoft Azure already allow application developers to focus on the development without the need of setting up the web integrated development environment (Web IDE) on the local machine [15].

Presented at WinTechCon-2010, organized by IEEE CAS Bangalore Chapter, IEEE Bangalore Section, and IEEE WiE Council Bangalore

E. Strong Authorization and authentications Concepts

As security is the most crucial factor for any application, cloud platforms provide in built security services that make the authorization and authentication aspects easily implementable.

III. HIGH LEVEL ARCHITECTURE EVOLUTION FOR THE CASE STUDY

Historical Way of Handling Pipeline Monitoring

Handling of a complex use case like automated Oil pipeline surveillance was never an easy task. However, even before cloud computing there were ways to achieve part of this use case [14].

One such instance was to use leak detection systems to check the leakage of flow in the pipelines. These leak detection systems would use to be either external or internal, depending on where they were installed on the pipeline. Additionally, these leak detection systems (LDS) would require also an integration with existing Supervisory Control and Data Acquisition (SCADA) systems [20]. There were several challenges faced in such kind of an architecture as mentioned below:

- These leakage systems could not give the accurate geographical coordinates of the leak due to the vastness of the pipeline.
- Monitoring was done via SCADA systems which were rudimentary since receiving the exact information on real-time basis was difficult.
- Too many physical devices (leak detection systems) made the whole setup difficult to manage both from operation as well as maintenance perspective.
- As the monitoring was mostly done via SCADA systems, there was not much flexibility to get a specific software to do this monitoring with respect to KPIs and analytics.

Additionally, Weibull distribution has historically been one of the tools for describing the probability of pipeline failures over time. While this technique is very accurate at describing failure distributions for large populations of components, it works very poorly at predicting the time until failure of an individual component. The mean time until failure is often used to predict times until failure of individual components, but this value may vary greatly with actual times until failure.

Proposed Architecture

As mentioned in the above segment, we propose to use the advantages of cloud computing and Integrated platforms (for instance Google Cloud Platform, Microsoft Azure Cloud Platform etc.) to create the surveillance and failure prediction application. We suggest using the inherent aspects of cloud computing like Cloud IDE (Integrated Development Environment), Pay and consume model services(Machine Learning, Internet of Things, Data Management, Authorization & Authentication, User Management etc.).

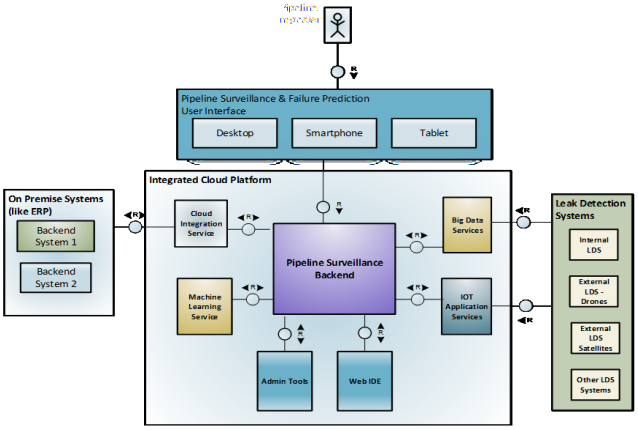

Fig. 1. Proposed Architecture of the Use Case

Figure.1, shows the overall reference architecture of the considered case study with the following details:

A. Surveillance Image Streaming via Satellites

The satellite images are continuously being sent to the cloud platform using stream processing capability of big data services.

B. Classification of Surveillance Images to find Anomaly

Then the big data management services help in cleansing and storing the data. As there will be always incoming feed of satellite images, the need to understand the incoming images and then classify it as useful or not is very critical.

C. Drone based Image and Data Capture

Once an anomaly has been detected, usage of drone is proposed to get the precise details of the leakage location and the particulars of the pipeline condition from the information received from the sensors. We propose to use IoT services that would be used to getting the information from the drones. The details are discussed in the subsequent segments of the paper.

D. Integrated Development Environment (IDE)

Web IDE will give us an inbuilt development environment, which helps the developer to build the surveillance and prediction app without any installations at the local machines (For instance, if JAVA is used as the language for implementation then Eclipse installation is not required at the local machine).

E. Backend Enterprise Resource Planning (ERP) Integration

As in most of the cases, there are backend ERP implementations that handle the maintenance order creation once there is a problem identified so that corresponding

processes are triggered in the system. The integration services from the cloud cockpit will be used to create the inbound and outbound scenarios from the pipeline surveillance and prediction failure application.

F. Security Concepts Handling

The Identity and Access Management (IAM) services on cloud provide the authorization and authentication concepts, also with single sign-on features so that the pipeline surveillance application can plug and use these features without much development effort.

G. Miscellaneous Admin Tasks

The Cloud Administrator will be using admin cockpit to handle general cloud activities like user onboarding, technical service handling, tenant setup etc.

IV. Converting Streamed Images to Meaningful Data Using Big Data Management Service

The images of the pipeline region sent via the satellites are continuously streamed into the cloud cockpit. The Big Data as a Service incorporates ways to capture image transformations and ensure that they are consistent and support coherent data interpretations. This service supports the following features:

- Stream processing engines with scanning capability on content
- Filters to select the meaningful information for capture
- Users must be able to define filter to check data relevant to identify alerts of spillage/maintenance
- Storage and subsequent access

This service aids in identifying the leakage patterns of pipeline and maintenance orders raised in the past for a company so that they can apply them in their machine learning model to improve pipeline health check cycle.

Using the power of big data, we intend to perform "predictive pipeline maintenance" to not only better manage pipelines, sensors, and drones, but also to help mitigate time down on maintenance of pipelines and sometimes even improve safety. Organizations can also use such a service to track the health of pipelines and improve asset utilization. It can also allow them to more quickly respond to sudden leakages.

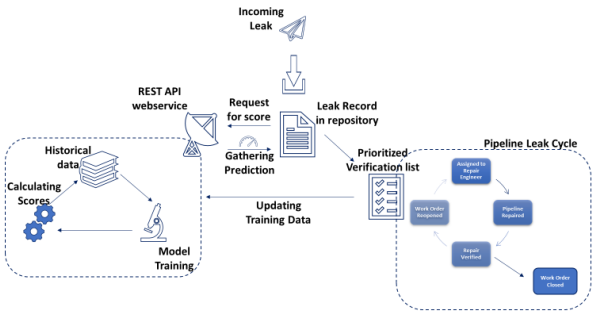

Fig. 2. Pipeline Leak Cycle; Depicting various elements of the ecosystem monitoring health and fixing leaks of a pipeline

The goal of such continuous learning system is to ensure the highest possible quality of exposed model.

In our research, we have used the Google Cloud Storage to store data collected from the IoT sensors. The data from IoT sensors were uploaded via files to the Google Cloud Storage bucket [21].

V. Integration to IoT Services for Data Analytics

With the advancement in technology, Advanced RISC Machines (ARM) core processors, Global Positioning System (GPS) sensors and batteries are being used in drones today to collect data. These devices are fully autonomous, connected to the Internet and providing a perspective from the remotest area that's not easily accessible for man. With Internet of Things, these drones do not need a separate device to operate them; our smart phones can be used to monitor the drones and the data collected by these drones can be stored on cloud. It is well-known that drones have the ability to gather up to half a terabyte per hour. Thus, we can conclude that drones can be used to collect big data, stored on cloud and then be properly analyzed using the latest advancements in data analytics to predict leakage in pipelines.

An intelligent cloud platform can be the one-stop place to enable a drone to function effectively. Various smartphone services deployed on cloud can be integrated with the data collection service placed on the drone. These smartphone services are used to capture various operating parameters of the pipelines mentioned in [11] are stored using the Google Cloud Storage bucket as explained in the Data Management Section. Subsequently the data is used by the CloudML Engine which is explained in the Machine Learning section. The operating parameters of the pipeline that are captured include Temperature, CO_2 partial pressure, Water Content, Flow Regime and Internal Pressure.

As this data is stored on the cloud, services for data analytics hosted on the same cloud platform are used to derive meaningful analysis. Since these services are tightly integrated, identification of leakage in pipelines can be real-time. This will also enable lesser loss of material and

thereby, lesser pollution to the environment caused due to pipeline leakages.

In [13], the authors affirm that telemetry data obtained from the drone's sensors can be provided as a source of input to fault detection services. In our research, we have simulated such an integration on the cloud using data integration services with reverse proxy mechanisms.

VI. Integration to Enterprise Resource Planning (ERP)Systems

The platform should provide integration services between disparate data sources and applications accessing them. A cloud platform service must connect to an enterprise and collaborate with its internal IT systems and applications. In our use case we plan to integrate data outsourced from the cloud platform to a target On Premise ERP installation to trigger a maintenance order based on the result of prediction/AI. The integration service primarily provides similar data and software integration as to an in-house integration application/service but uses the cloud for delivery or to enable integration.

An integration scenario can be achieved in an Inbound and Outbound manner. Inbound integration involves data coming into to the cloud and Outbound integration refers to data/information exposed from the cloud. Any integration service delivered out of the cloud platform is generic and directly relatable to the default data model setup in the cloud. However, these services are generally consumed by applications and transformed according to the consuming application needs. In an inverse manner the inbound service is consumed by the cloud application.

In our Pipeline surveillance use case the following Inbound and Outbound Services are proposed to be provided from the cloud platform.

Proposed Inbound Integration Services

- Generic Sensor Data(Information received from sensors placed on pipelines)
- Image Information(Images streamed from any form of surveillance)
- Drone Broadcasting Data(Drone signal feed)
- Alert Information(Alerts received from any third party)

Proposed Outbound Integration Services

- Pipeline Health Check(General information specifying the health of a pipeline)
- Leakage Alert(Notification of an existing or a possible pipeline breakdown)
- Request Maintenance Order(To send all information required to raise a Plant Maintenance Order)

VII. Handling of Identity and Access Management:

Identity and Access Management (IAM) is one of most critical aspects for any software application. For enterprise applications, the complexity increases as there are also heterogenous types of systems that communicate to each other. In this case study, Security and IAM aspects become utmost critical as these pipelines may be owned by various stakeholders (varying from region to region). Also, there might be various vendors involved (again varying geographically) who are responsible for maintenance of the pipeline.

Additionally, there might be a necessity to immediately create a Maintenance Order in the backend Enterprise Resource Planning system. Thus, authorization and authentication not only require the concepts for running an isolated app but to also need a secure and safe communication channel between cloud app and backend system. Thus, in a complex use case like this, multiple levels of authorization and authentications are required.

Browser-based end-user single sign-on has become a commodity in the cloud, as basically all cloud applications support the Security Assertion Markup Language (SAML). There are SAML based authorization/authentication services provided on Cloud platforms (e.g. AWS IAM Services, Google Cloud Platform IAM Services) that help the mobile application development easier to implement from development perspective. These services provide mechanism for authentication, single sign-on, user management, and backend integration. Many of them also provide user self-services such as password management etc. From the Identity perspective, these services provide security features for protecting access to applications - authentication rules, two-factor authentication and delegated authentication to on-premise systems.

VIII. Using Machine Learning Services for Predictive Maintenance:

Existing Machine Learning Algorithms can help achieve Pipeline Failure Prediction

With the advent of machine learning techniques, the ability to learn from past trends to predict future behavior makes it possible to predict an individual component's time until failure much more accurately. The conclusion drawn from the paper [17] shows that traditional data mining and Machine Learning techniques are unsuitable for handling big Data however, deep learning has the potential in dealing with such challenges.

With a focus on the heterogenous characteristics of crude being supplied through pipelines [9], the proposed model helps us predict the risk of an equipment failure. In this use case, the dimensions used to machine learning algorithms are Pipeline Identifier, Quantity of crude, Under Water Corrosion Index, Thickness, Pressure, Temperature, Density and Timestamp.

Details of Machine Learning Implementation

In this research we used the R interface to Google CloudML [18] to deploy a model (based on deep learning) in the Google CloudML Engine which was built using the TensorFlow Package in R. The Google CloudML Engine helps us to monitor the scheduled jobs on the models deployed [19].

A. Load Data

All the input variables that describe the health of the pipeline are numerical which can be directly used for neural network in Keras.

Note, the dataset has 9 columns which consist of eight input variables and one output variable. Once data is loaded we split the dataset into input variables (X) and output variable (Y)

```
#loading the data from a file
traindata <-read.csv("operating_parameters.csv", sep = ',', header = TRUE)
# split into input (X) and output (Y) variables
X = array(data = traindata, dim = 0:8)
Y = array(data = traindata, dim = 8)
```

Fig. 3. R code to load data and split input and output variables

B. Define Model

Models in Keras are defined as sequential layers. First, we ensure the input layer has the right amount of inputs. This can be specified when creating the first layer and setting it to 8 for the 8 input variables. Second, we define a hidden layer with dropout rate of 0.4 and 1 output layer has 1 neuron to predict the class (pipeline needs maintenance or not)

```
#defining a keras sequential model
model <- keras_model_sequential()

#defining the model with 1 input layer[8 neurons], 1 hidden layer[8 neurons] with dropout rate 0.4 and 1 output layer
model %>%
  layer_dense(units = 8, input_shape = 8) %>%
  layer_dropout(rate=0.4)%>%
  layer_activation(activation = 'relu') %>%
  layer_dense(units = 1) %>%
  layer_activation(activation = 'sigmoid')
```

Fig. 4. R code to define a Keras Model and create input and output layers

C. Compile Model

After defining the model, we compile it. We must specify the loss function to use to evaluate a set of weights, the optimizer used to search through different weights for the network and any optional metrics we would like to collect and report during training.

In our case study, we have used logarithmic loss, which for a binary classification problem is defined in Keras as "binary_crossentropy". We have also used the efficient gradient descent algorithm "adam" as it is an efficient default. Finally, because it is a classification problem, we collect and report the classification accuracy as the metric.

```
#compiling the defined model with metric = accuracy and optimiser as adam.
model %>% compile(
  loss = 'binary_crossentropy',
  optimizer = 'adam',
  metrics = c('accuracy')
)
```

Fig. 5. R Code to compile the Keras Model

D. Fit Model

Now we have evaluated the model on the simulated data. We used the fit function in R to fit our model on our loaded data. We have run this for a small number of iterations (100) and used a small batch size of 10. These results can be further be optimistically chosen by trial and error.

```
#fitting the model on the training dataset
model %>% fit(X, Y, epochs = 100, batch_size = 12)
```

Fig. 6. R code to Fit the Model on Training Data

During the run of the R program, a message is seen for the 100 epochs along with the loss and accuracy for each epoch as shown in the figure below.

```
Epoch 1/100
60000/60000 [==============================] - 11s 192us/step - loss: 0.2467 - acc: 0.9296
Epoch 2/100
60000/60000 [==============================] - 11s 179us/step - loss: 0.0973 - acc: 0.9715
Epoch 3/100
60000/60000 [==============================] - 10s 165us/step - loss: 0.0619 - acc: 0.9813
Epoch 4/100
60000/60000 [==============================] - 10s 166us/step - loss: 0.0434 - acc: 0.9871
Epoch 5/100
60000/60000 [==============================] - 10s 175us/step - loss: 0.0309 - acc: 0.9910
Epoch 6/100
60000/60000 [==============================] - 10s 165us/step - loss: 0.0208 - acc: 0.9945
Epoch 7/100
60000/60000 [==============================] - 10s 166us/step - loss: 0.0154 - acc: 0.9960
Epoch 8/100
60000/60000 [==============================] - 10s 166us/step - loss: 0.0117 - acc: 0.9968
Epoch 9/100
60000/60000 [==============================] - 10s 167us/step - loss: 0.0098 - acc: 0.9975
Epoch 10/100
60000/60000 [==============================] - 12s 204us/step - loss: 0.0099 - acc: 0.9974
Epoch 11/100
60000/60000 [==============================] - 12s 208us/step - loss: 0.0086 - acc: 0.9975
```

Fig. 7. Evaluation of epochs during Training Run

E. Evaluate Model

We had split our simulated data into train and test datasets. We then evaluated our model on our test dataset by using the evaluate function in R.

```
#Evaluating model on the cross validation dataset
loss_and_metrics <- model %>% evaluate(test_x, test_y, batch_size = 128)
```

Fig. 8. R Code to evaluate a model on a test dataset

The simulated dataset from the IoT sensors used to train and test our model reached the accuracy up to 0.98 and losses up to 0.006 when run for 100 epochs as seen in the figure below.

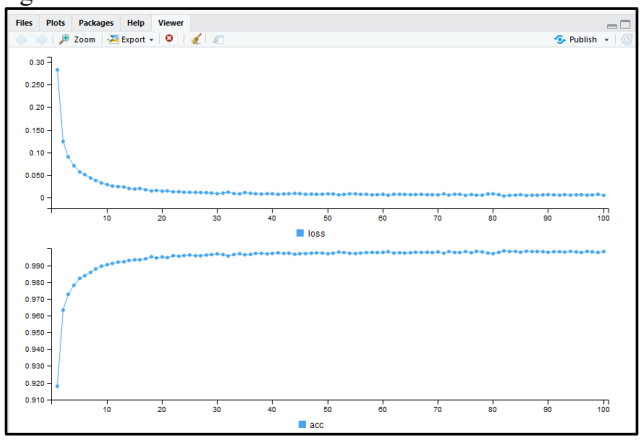

Fig. 9. Loss and Accuracy of the Model built using R when run for 100 epochs.

IX. SYSTEM PERFORMANCE COMPARISON

The comparison report below in TABLE I is our view of the summarized data established by several attributes [21] used to evaluate the performance of legacy systems used to maintain Oil Pipelines with the proposed Integrated Cloud Cockpit. For this, we compare the execution performance of the methods established in the prior work using the leak detection systems [20], with results observed by usage of the proposed solution.

TABLE I. COMPARISON OF LEAK DETECTION SYSTEM WITH INTEGRATED CLOUD COCCPIT

METHOD	FEATURES					
	Easy Usage		False Alarm		Cost	
	LDS	ICC	LDS	ICC	LDS	ICC
Inspection by helicopter	Yes	Yes	No	No	High	Low
Drone Monitoring	No	Yes	Yes	No	Med	Low
IoT Pressure Point	No	Yes	No	Yes	High	Med

X. CONCLUSION

The proposed architecture of an Integrated Cloud Cockpit where every independent module exists as a service is the need of the hour for organizations striving to build end-to-end applications to provide their consumers with seamless navigations. With such an Integrated Cloud Cockpit, a complex scenario of pipeline surveillance and prediction of failures can be easily resolved by creating software applications powered by cloud computing.

REFERENCES

[1] Babcock, C.: Management Strategies for the Cloud Revolution.McGraw-Hill, New York (2010)

[2] Oltsik, J.: What's Needed for Cloud Computing. Enterprise Strategy Group, Inc. (2010)

[3] Oszczuk–Januszewska J. (2011) The Advantages of the Use of Cloud Computing in Intelligent Transport Systems. In: Mikulski J. (eds) Modern Transport Telematics. TST 2011. Communications in Computer and Information Science, vol 239. Springer, Berlin, Heidelberg

[4] Reilly, S. & Dow, G. Comput Game J (2013) 2: 14. https://doi.org/10.1007/BF03392341

[5] Partha Pratim Ray, "A survey of IoT cloud platforms", Future Computing and Informatics Journal, vol. 1, pp. 35, 2016.

[6] J. Zhou, T. Leppanen, E. Harjula, M. Ylianttila, T. Ojala, C. Yu,and H. Jin. Cloud things: A common architecture for integrating the internet of things with Cloud computing. In CSCWD, 2013.IEEE.

[7] Peter Mell, Timothy Grance, "The NIST Definition of Cloud Computing", NIST Special Publication 800-145.

[8] Babu, S.M., Lakshmi, A.J., Rao, B.T.: A study on cloud based Internet of things: CloudIoT. In: Global Conference on Communication Technologies (GCCT), India, pp. 60–65 (2015)

[9] Chern-Tong, H., Aziz, I.B.: A Corrosion Prediction Model for Oil and Gas Pipeline Using CMARPGA, pp. 403–407. IEEE (2016)

[10] Real-Life Big Data Examples. Retrieved from http://www.evolvingsol.com/2014/05/16/real-life-big-data-examples/

[11] Pipeline Risk Based Inspection. Retrieved from http://www.oilfieldwiki.com/wiki/Pipeline_Risk_Based_Inspection

[12] Predictive pipeline maintenance. Retrieved from https://www.hydrocarbons-technology.com/features/feature130338/

[13] Hansen, S., Blanke, M., & Adrian, J. (2012). Fault Diagnosis and Fault Handling for Autonomous Aircraft. Technical University of Denmark, Department of Electrical Engineering

[14] Geiger, G., T. Hazel, and D. Vogt. "Integrated SCADA-based approach for pipeline security and operation." Record of Conference Papers Industry Applications Society 57th Annual Petroleum and Chemical Industry Conference (PCIC). 2010.

[15] Sriram, I. and Khajeh-Hosseini, A., 2010. Research agenda in cloud technologies. arXiv preprint arXiv:1001.3259.

[16] SEDAYAO, J. 2008. Implementing and operating an internet scale distributed application using service-oriented architecture principles and cloud computing infrastructure. In iiWAS '08: Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services, 417-421.

[17] Lidong Wang. Data Mining, Machine Learning and Big Data Analytics. International Transaction of Electrical and Computer Engineers System. Vol. 4, No. 2, 2017, pp 55-61

[18] R Interface to Google CloudML. Retrieved from https://tensorflow.rstudio.com/tools/cloudml/articles/getting_started.html

[19] Developing a TensorFlow Training Application. Retrieved from https://cloud.google.com/ml-engine/docs/tensorflow/trainer-considerations

[20] Lawrence Boaz, Shubi Kaijage and Ramadhani Sinde: An overview of pipeline leak detection and location systems. In: Pan African International Conference on Science, Computing and Telecommunications (2014)

[21] Staord, M., Williams, N., 1996. Pipeline leak detection study. Tech. rep., Bechtel Limited.