Senior Design Final Report
Bad Area Detector

Version 1.0 - 05/10/2019

Team Members:
Alex Rodriguez
Eduardo Gutierrez
Ka-Yee Yeung
Robert Martinez
Tanner Hoeflicker

Faculty Advisor:
Professor John Hurley

Liaisons:
N/A
# Table of Contents

1. Introduction  
   1.1. Background  2  
   1.2. Design Principles  2  
   1.3. Design Benefits  3  
   1.4. Achievements  3  

2. Related Technologies  4  
   2.1. Existing Solutions  4  
   2.2. Reused Products  4  

3. System Architecture  5  
   3.1. Overview  5  
   3.2. Data Flow  6  
      3.2.1 / 3.2.4 User Interface/Web Application  6  
      3.2.2 IBM Cloud Analytics  6  
      3.2.3 User Account Authentication  7  
      3.2.5 Cloud Object Storage  7  
      3.2.6 Simulated Police Dispatch  7  
   3.3 Implementation  7  
      3.3.1 San Diego Police Dispatch Data  7  
      3.3.2 IIoT Cloud Setup  7  
      3.3.3 User Interface/User Experience  8  
      3.3.4 Web Application Backend  8  
      3.3.5 Data Science Experience  8  

4. Conclusions  10  
   4.1. Results  10  
   4.2. Future  10  

5. References  12
1. Introduction

1.1. Background
The Industrial Internet of Things (IIoT) generally refers to smart sensors and actuators (edge devices) placed on industrial equipment. These components allow industry users to monitor the health and energy use of equipment by sending metadata to a cloud platform. This in-turn allows for preventative maintenance measures and data analytics to be run which can reduce costly unplanned repairs.

The Bad Area Detector (BAD) has transformed this concept to fit its own purpose: to increase public safety. The system utilizes microcomputers as edge devices to relay police dispatch call information to the IBM Cloud. Once in the cloud, the data is placed into a database where it can be accessed by the BAD Web Application to be displayed in various forms to authorized Department of Transportation (DoT) employees as well as first responders. The data is also analyzed in the cloud by machine learning algorithms that can create predictive models which can then be utilized to plan logistics such as bus routes or daily police patrols.

The BAD System was created as a practice exercise to prepare for a competition held by the partnership of General Electric (GE) and the City of San Diego. The idea was to create an IIoT project that would improve the city in some way by utilizing GE’s IIoT platform, Predix.io. However, the competition was canceled due to unforeseen circumstances. For this reason, the BAD System now runs on the IBM Cloud but is designed to run on any IIoT platform.

1.2. Design Principles
The web application is the face of the BAD System and is used to show information from police dispatch calls. The application user interface (UI) is designed with minimal feature selection in order to maintain simplicity while maximizing information. The analytics created by machine learning algorithms are not displayed in the UI, but are created utilizing the Data Science Experience as defined by IBM through the Watson Studio. These statistics must be created and delivered to the proper authorized authorities by engineers.

The data flow behind the system is designed in such a way that it should be transferable between different cloud platforms with ease. The Simulated Police Dispatch (SPD) is in place to show the functionality of the BAD System by utilizing publicly available police dispatch data because recent and live data is only available to authorized personnel. The SPD consists of an edge device that would otherwise easily be implemented in any live system.
1.3. Design Benefits
Having an architecture designed around the IIoT caters to future scalability. Multiple edge devices placed at various authorized locations can easily be added to the BAD System with security credentials provided by the IBM Cloud Platform. All incoming dispatch information from these devices can be pushed into a single, scalable database. The amount of edge devices does not change the way the data is utilized by the web application or Watson Studio. This means that a user can add multiple cities to the system by simple adding edge devices.

The UI is designed around simplicity allowing for users to interact with all its components. There are multiple tooltip buttons that open information panes which instruct the user on how to use the feature selections.

1.4. Achievements
Throughout the past academic semester, our team has been able to develop a web application that provides first responders with information in several different forms. The BAD System implements services such as the Cloud Object Storage (COS), a storage service that is designed for high durability and security. Our web application utilizes the COS to pull more than 10,000 data points into the backend of our application. BAD is also successful in displaying all the information from the back end to the front end using a javascript library called React.js. React.js is a library used for building user interfaces with an emphasis on fast rendering via the Virtual DOM, therefore allowing simulations to happen with ease on the BAD System.

App ID is another service from IBM cloud that is implemented into BAD. App ID provides user sign-in/sign-up functionality with easy profile management on the developer side. This service ultimately provides the security for BAD System, allowing us to focus on other developments. BAD also uses Carbon Design System, an open source design system that consists of a list of UI components. Carbon UI components have great detail and each have their own functionality. The goal of using Carbon Components is to maintain a modern theme with a consistent UI components across the front end of the BAD System.

Furthermore, the application also uses a web application framework called, Express.js. Express.js provides the tools and functionality for structuring a web application to handle various HyperText Transfer Protocol (HTTP) requests. An Application Programming Interface (API) was developed using Express.js, in order to allow the front end of the application to request its necessary data.
2. Related Technologies

2.1. Existing Solutions
Currently, most police stations and officers in the field use what is called the Computer Aided Dispatch (CAD) System. This system is used to relay information from the dispatcher to the police officers on duty in the field. The user-interface seems dated but serves its purpose well. The incoming dispatch data has fields for date, time, call type and a summary of the call. There is a map component showing where the current user is as well as the location of a reported incident. This system has many similar functions as the BAD System but the BAD is meant to do more than just show current dispatch information.

There are many ambulance companies that exist and each one has different methods to alert their responders. AmbuServe is one of those companies and it uses a pager system to accomplish this. When the company receives a call from a dispatch center they contact their Emergency Medical Technicians (EMTs) by radio to send location information via a pager that each employee carries. The employees then enter the information into a Global Positioning System (GPS) device or their personal cell phone.

PublicEye is a private company that has a product similar to the BAD System. Their product is marketed toward city departments and civilians to create a collaborative geo-alert system. They use IoT and give the following as examples of how the system can be utilized: “Early detection of floods or wildfires, reaching heart attack victims in time and protecting infrastructure from hazardous materials.” The difference between the BAD System and PublicEye is that BAD is geared towards first responders and public transportation drivers. BAD is meant to relay information to first responders that is not available to the public. Another difference is that the BAD can deliver valuable crime statistics that can help these departments decide where to allocate future resources.

2.2. Reused Products
The BAD System is reliant on previously produced IBM services and libraries. As mentioned earlier, App ID is an IBM Cloud service that provides sign-in and sign-up functionality as well as the overall security for BAD. Carbon is IBM’s official design system that is developed and maintained by IBM designers, developers, writers, etc. Carbon is used in the front end of our application, utilizing multiple components to display a modern UI. ReactJs is a javascript library for building user interfaces and produces efficient rendering based on data changes. Node-RED is a flow based programming tool that allows data and or commands to be sent from an edge device to a cloud platform. In our case, a Raspberry Pi 3 Model B is running Node-RED which sends the data to the Watson IoT Platform.
3. System Architecture

3.1. Overview
The Context Diagram (DFD level 0) provides the overall structure of the software modules and all its inputs and outputs. The BAD architecture is summarized in the Context Diagram given below:

BAD Data Flow Diagram: Level 0

- Police dispatch calls are simulated and injected into the BAD System by utilizing IBM’s Industrial IoT platform.
- Dispatch data is stored in the COS where it is readily available for analytics.
- Data is pulled and displayed to a front end web application’s user interface.
3.2. Data Flow
The overview of data flow can best be described by our DFD level 1 as follows:

BAD Data Flow Diagram: Level 1

There are six modules in the BAD System which are described below.

3.2.1 / 3.2.4 User Interface/Web Application
The UI and the Web application are essentially one in the same but are separated as modules due to functionality. The UI is the face of the BAD System which is viewed by first responders. It sends request to the Web Application which pulls data from the COS via an API.

3.2.2 IBM Cloud Analytics
This module is called the “Data Science Experience” in the IBM Cloud. It is a combination of Apache Spark and the Watson Studio being used in tandem and is not intended to be available by the average user but rather data analyst and software engineers. The module is intended to analyze data and provide information that can aid in determining where future resources should be applied to be most effective in stopping crimes.
3.2.3 User Account Authentication
This module was implemented to provide security to the application. The data used to show the functionality of the BAD System is available for public use. However, when the system is deployed for use by first responders, the data is considered sensitive until the proper authorities deem it releasable to the public. This is the reason the BAD System needs users to register before using the product.

3.2.5 Cloud Object Storage
The COS is a long term, encrypted database that is globally distributed. This module receives data from the Cloudant NoSQL database via an event stream. It is meant to store all the police dispatch data so that it is readily available for use in the web application and analytics modules.

3.2.6 Simulated Police Dispatch
This module consists of the Raspberry Pi and is the edge device in this IIoT structure. The SPD was created to show the functionality of the BAD System due to the fact that we did not have access to live police dispatch data. For future use, this module would not simulate data but would be connected to a database in any police station and stream live dispatch data to the cloud as the calls were made.

3.3 Implementation
There are five individual sections for the development of the BAD System: San Diego Police Dispatch Data, IIoT Cloud Setup, User Interface/User Experience, Web Application Backend and the Data Science Experience.

3.3.1 San Diego Police Dispatch Data
The data used is available through the City of San Diego Open Data Portal website. The database used has a total of 16 features which are not all used (more information in section 3.3.5). The dataset originally only used street addresses for the locations of reported calls which had to be converted to latitude and longitude. The call types were originally represented as numbers which can be seen on the UI as they are displayed above each gauge. However, when one clicks the information button above the gauge area, the string definition of each call type is shown. This information is not found in the original dataset and has to be found from Police Calls for Service Dictionary file which is available on the City of San Diego Open Data Portal as well.

3.3.2 IIoT Cloud Setup
Starting with the IBM Cloud Platform, user needs to create a IBM Cloud Account. This account will allow users to manage all the services that IBM offers. Follow the user guides to set up the
After setup, the user will be able to configure the Cloudant NoSQL DB as a historical database for the Watson IoT Platform. Once completed, the user will need to go through the procedure found in the guides to set up their edge device. Once completed, the user needs to connect the historical database to the COS via the Event Stream service. Instructions for setting up the Watson Studio along with Apache Spark are detailed in the IBM Community guides. These guides should be followed after all previous steps have been completed.

3.3.3 User Interface/User Experience
The user interface is comprised of two libraries, ReactJS and the Carbon Design System. ReactJS is the core library used to build the interface. The Carbon Design System is a series of individual components and styles that was used to make an intuitive design. The implementation of the Carbon components helped us as developers to create a modern interface as well as memorable user experience.

3.3.4 Web Application Backend
The backend of the application was developed using Node.js, allowing us to write the server-side application in JavaScript. Moreover, the use of Node.js allowed us to leverage its native package manager, npm, providing us with many popular frameworks and modules to aid us in the development process. An API was created using Express.js, a web application framework that provides the functionality and tools to structure a web application so it is able to handle different HTTP requests.

Passport.js and IBM Cloud App ID are two other important components that work hand in hand along with Express.js, and are responsible for protecting our resources and handling the authentication of users. Passport.js is an Express.js middleware that you can add to certain API endpoints which require authentication before you are able to access them. Passport.js along with a strategy provided by IBM Cloud App ID will redirect unauthenticated users to an IBM login page where they must sign in before they are able to access the BAD System.

3.3.5 Data Science Experience
We made several predictions with the San Diego Police Dispatch dataset. The dataset contains the location, time, crime type and priority level of each crime.

We first predicted the priority level of crimes that happen during a specific hour on a given day of the week. The date feature in our dataset is given as follows: 1/2/2017 23:51:1. We decided not to use the seconds or minutes of the crime. For this reason, we made two new features in our dataset, the first being the day a dispatch call was made by giving it a numeric value with 1 being
Monday and 7 as Sunday. The other column that was created was “hour” which we again assign an integer value, in the range of 1 to 24, to represent hours of the day.

Next, we created a new column to categorize location data by area. This was done by labeling locations as the same if they were in a 1 to 3 mile radius of each other, creating crime area clusters. This in-turn reduced the location feature from the total number of entries in the dataset down to about 150 different clusters, which vastly improved the prediction accuracy of our machine learning models.

The values we are trying to predict is the priority level which is a measurement of the severity of a crime with 4 being the most serious. Because there are only 4 possible results, classification algorithms were used.

The first classification algorithm used to create a model was the Decision Tree, which had an accuracy score of 92%. The second algorithm used was Random Forest and this yielded an accuracy of 94.5%. The last algorithm, an artificial neural network (ANN) defined by further by multilayer perceptron (MLP), scored an accuracy result of 46%.

The last prediction attempted was to determine the crime rate of the area clusters. The same features were used as in the previous prediction stated above. The formula used to find the crime rate in the data is as follows:

\[
\frac{\text{Crimes Committed at given time in crime area cluster} \times 100,000}{\text{San Diego Population}}
\]

Because crime rate value is continuous with an infinite range a Linear Regression algorithm was utilized. The accuracy of the model’s predictions was determined by evaluating the root mean square error (RMSE), which came out to be 15.7. This number gives the standard deviation of the model’s prediction errors. This results in a satisfactory model given that it is trying to predict future crime rates with a small amount of features.
4. Conclusions

4.1. Results

The UI exists within the BAD Web Application and consists of 5 components. Each component displays different statistics in a unique way based off the features selected. They are listed as follows:

- Map Component
  - The first feature displays the locations of dispatch calls using markers. The second feature displays a heatmap that indicates the concentration of crimes in a given area.
- Call Table
  - Displays all incoming dispatch data with statistics allowing users to see the full history of calls and it provides a filter and search option.
- Gauges
  - Displays the top five most called in priority types.
- “Priority Call Stats” Section
  - Displays mini bar graphs that represent the number of priority calls per month that fall into the categories priority 1, 2, 3 and 4.
- Line Graph
  - Displays the total calls per month.

To access the web application the user must register his/her credentials through a web authentication page. Once this step is completed the user will have full access to the UI.

The Watson Studio utilizes Apache Spark to pull data from the COS and run machine learning algorithms to analyze the data. We used both supervised and unsupervised learning techniques for our predictions: Random Forest, Decision Tree and AAN for classification and Linear Regression for the regression prediction.

4.2. Future

Future improvements for the BAD application will be the following:

- Display machine learning results in real time in the front end web application:
  - Currently, our data analytics and machine learning models are being processed in the back-end.
  - The machine learning model will train itself on new data as new calls are made and update its weights accordingly.
• A mobile application:
  ○ A mobile application would allow easier access to the BAD.
  ○ Making BAD mobile would assist first responders in navigation while on the go.
  ○ The application will create “push” notifications to the users’ mobile phone or tablet.
  ○ The mobile application would be available for both Android and IOS users.
5. References

- App ID
  - https://cloud.ibm.com/catalog/services/app-id

- Carbon Design System
  - https://www.carbondesignsystem.com/
  - http://react.carbondesignsystem.com

- Cloud Object Storage
  - https://www.ibm.com/cloud/object-storage

- ExpressJs
  - https://expressjs.com/

- Google Maps React
  - http://google-map-react.github.io/google-map-react/map/main/
  - https://sites.google.com/site/gmapsdevelopment/

- IBM Watson
  - https://www.ibm.com/watson

- Node-RED
  - https://nodered.org/

- PassportJs
  - http://www.passportjs.org/

- Raspberry Pi

- ReactJs
  - https://reactjs.org/

- ReduxJs
  - https://redux.js.org/

- City of San Diego Open Data Portal
  - https://data.sandiego.gov/datasets/police-calls-for-service/