IoT Sensor System to Record The Temperature and Location of Ladle Vessels Used In Metallurgy

Craig Martin
School of Computing, Engineering and Digital Technologies
Teesside University
Q5031372@tees.ac.uk

April 2020

Contents

Abstract

The world is changing with more systems becoming automated through the Internet of Things (IoT). By the end of 2020, the number of connected devices is estimated at 38.5 billion, which is an increase of 25.1 billion from 2015. With the digital revolution thoroughly underway companies are now fully aware that statistical data can help their businesses thrive and increase efficiency. In this paper, the design and implementation of an Arduino sensor system used to record the temperature and location of Ladle Vessels for metallurgical purposes is presented. The Arduino microcontroller has multiple different modules connected to it, including a K-Type thermocouple, GPS Shield, Real-Time clock, and a Bluetooth module. To conclude this paper, the results from a live experiment are displayed where the system was deployed for eighteen hours.

1 Introduction

The Internet of Things (IoT) is expected to become the latest revolutionary change in the technology industry, affecting all aspects of everyday life. The IoT has rapidly grown in popularity over the past few years and by the end of 2020, 100 billion connected devices could be part of the IoT [9]. Many different industrial sectors are interested in the potential of the IoT and how the collection of data can transform their businesses by automating tasks. [1]. The data recorded from the devices can be useful to interested parties to help make important decisions. For example, IoT sensors are becoming more widely used in the healthcare sector to record patients blood pressure, sugar levels and weight with the data being accessible by physicians anytime. This can lead to more lives being saved due to the constant access of the data [12].

The IoT has several elements: sensors to obtain the data, data processors, to analyze the data, and finally actuators, that respond to the information by moving and controlling machinery [1]. David Fletcher states that it is "The ability of objects to communicate that delivers the power of the IoT" in his paper on the Evolution of Cyber Technologies [9]. The communication between IoT devices is not just intended for data acquisition, in many cases, they are meant to perform and complete specific tasks that interact in the physical realm improving efficiency in previously manual tasks [1]. One example is used in the transportation sector, were traffic lights are changed based upon real time traffic flow information. The traffic lights can take the information from sensors and alter the flow of traffic based upon the data to ease congestion [13].

1.1 Problem Overview

In metallurgy, the temperature of the molten metal entering the casting process affects the quality of the result. Therefore, the temperature of the ladle vessel must be carefully controlled. Ladles are used to transport molten steel to the continuous casting operation, where the liquid is poured into moulds to create the casting. The ladles must be preheated to decrease thermal shock and damage to the refractory metals[10]. Once the vessel has reached the optimal temperature, any extra overheating is redundant and can be costly.

Material Processing Institute is a research and innovation centre that develop high quality, bespoke steel castings on a commercial basis. They offer a wide range of steel alloys weighting up to six tonnes. As part of the companies Industry 4.0 initiative, they wanted to develop and enhance their current IoT solutions to further examine data from machinery and equipment.
The company required a live monitoring solution to accurately record and analyze the ladle shell temperature and location during the preheat and transportation phases of the metal making process. The ladle shell has eight unique positions around the edge and on top of the vessel that require monitoring to most accurately assess the current temperature and location status.

1.2 Constraints of the Project

Due to the nature of the environment that the solution would be deployed in, the hardware that is recording the live data needs to be cost-effective, easily replaceable and durable. The ladle shell is subject to extreme temperatures and has the potential to damage nearby objects.
The preheat phase of the ladle shell can take up to eighteen hours to complete, therefore, it is imperative that the solution can successfully record the data for the entire process and has fail-safes in place to store the data if the connection falters. For example, if the device's Bluetooth or network connection drops then it must continue to record the data but store it locally in memory and transmit the information once the connection is reestablished. Furthermore, all aspects of the microcontroller's selected hardware, chosen communication method and programming logic needed to be carefully considered to optimize power consumption as the device needed to operate for long periods with minimal user engagement.
Material Processing Institute also develops and tests new systems for external metal producers meaning any system developed would need to be transferable, reusable and scalable to other locations with minimal effort and adjustments. The network connectivity is poor inside the casting area where Material Processing Institute develops the metals, so both an offline and online solution is required. This is to be achieved by changing command line parameters and configuration file variables.

1.3 Proposed Solution

The proposed solution to the problem and the many constraints uses an Arduino microcontroller. Each controller will have an attached thermocouple to gather the current temperature and an external communication module, such as Bluetooth or WiFi, to transmit the data. To investigate the capability of Arduino microcontrollers in a production environment each Arduino would have some unique modules and functionality. The modules include a GPS shield to obtain the current latitude and longitude of the device, a real-time clock to accurately record the exact moment the temperature reading was taken. An infrared communication module to allow the user to stop the data logging via a remote.

The data is recorded by the IoT devices and then displayed through a WPF application (.NET Core, Desktop usage only), with the following capabilities:

1. **Display Live Data** - Each active device's data is displayed on a graph as a live data feed.

2. **Display Previously Recorded Data** - The user can see all the temperature and data for each device between specified date ranges.

3. **Calculate & Present Statistical Information** - Statistical data is calculated for the selected device(s) and displayed to the user. The statistical

data includes the range, mean average and standard deviation of the data set.

The WPF application is also responsible for collecting the data outputted by each nearby Arduino with Bluetooth communication capabilities each second. The received information is then sent to a local SQL database if the application is running in development mode or to an Azure-hosted SQL database if the application is running in live mode.

The Arduino devices that communicate using a network connection send the recorded data to an online Web API, which, in turn submits the data to the SQL database hosted on Azure. The WPF application can be configured to operate using the local or hosted SQL database with command-line arguments and configuration files.

In addition to the WPF application, an Android application has been developed that allows users to view data remotely, via their mobile devices. The Android application only can display previous data to the user and does not include the live or statistical information functionality.

1.4 Background of Problem

Previously, Material Processing Institute recorded the temperature of the ladle shell by using a multi-channel thermocouple data logger which allows twelve channels of temperature readings to be stored on an SD card at a user-specified rate. Even though the device was able to successfully and accurately record the temperature data it had multiple issues and limitations.

1. **Cost** - The price of even the cheapest data loggers can be extremely expensive which is a problem due to the harsh environment the metal is created in.

2. **Damage & Replacement Parts** - As the device is a specialist piece of equipment you cannot obtain replacement parts easily, requiring consumers to purchase a new device each time it breaks.

3. **Data Acquisition** - The device can only log the data to an SD card inserted into the device. Meaning the scalability of such a system is non-existant and requires technologically literate end-users.

4. **Unable to Obtain Other Aspects of Real-Time Data** - The multi-channel device can only record temperature data and cannot record the current date and time and GPS location of the vessel.

Other similar projects and research has previously been conducted regarding the attainment of temperature data using Arduino microcontrollers and thermocouples. Engineers from Varna University in Bulgaria developed an Arduino based device for measuring sixteen channels of live temperatures with

thermocouples in the temperature range from 0°to 1250°. The team used the Arduino Uno microcontroller, which is the most commonly used board, and K-Type thermocouples to receive the latest readings from electrically resistive furnaces. The team's objective was to develop a device that was able to read the latest temperatures and prove the accuracy and precision of the data being recorded [16].

2 Methodology

2.1 Methodology Chosen & Rationale Behind Decision

Due to the structure of the task and the different actors involved the waterfall methodology became the natural fit for the project. The waterfall approach follows a set of steps in a specific order, with the first stage analysing and designing, followed by the implementation, and finally testing and evaluating the final creation [6].

At the start of the project, a qualitative research interview with Material Processing Institute was conducted to understand the requirements and expectations of the project before any development was undertaken. A qualitative technique was well suited for the project as a detailed specification needed to be established before any development could occur.

2.2 Analysis & Evaluation of Methodology

Overall, the waterfall methodology was a good structure to follow for this project as a clear understanding of the problem was needed before any development could be undertaken. Initially, a specification that later developed into a design proposal was agreed upon by all interested parties. Having the initial documentation in place leads to a clear vision and expectation of the final product that was going to be delivered and ensured smooth delivery.
Next, the implementation of the system began until all of the agreed features were completed to a high standard. The waterfall methodology worked well for this phase of development due to the limited time frame the project needed to be completed in. The waterfall approach gave a structured list of features that must be completed for the application to be considered usable and did not allow for new features to be added or changed constantly which would have increased the development time considerably.

2.3 Comparison with Other Methodologies

Another popular structure to use when developing software, that is widely used, is the Agile methodology which focuses on rapid development and teamwork. Agile development is about continuously developing new features at a fast pace and requires strong leadership and communication between all parties. This methodology would have increased the rate at which the final solution was

reached, however, would have been more difficult to achieve because of the inability to communicate regularly with all actors in the implementation phase.

3 Legal, Social & Ethical Considerations

3.1 Information Security

Many experts in the field of The Internet of Things state that the most numerous challenge facing the industry is the protection and security of IoT devices. The greatest power of the IoT is the ability for many small objects to communicate with larger objects, creating a network of devices capable of recording data. The IoT's security is only as strong as the weakest object in the network. If one object is compromised the entire network could become under threat. The ease of access to IoT devices and connecting them to networks, mobile devices and other computers without directly requesting permission creates the possibility of high-risk security breaches [4].
With more IoT devices becoming connected than ever before, it is becoming more vital than ever before that careful security considerations and understanding of the risks are recognised before the deployment of any IoT system [15].

Another underlying problem with IoT systems is that they have environmental data capturing components attached to the microcontroller processor which can be sent to an external server for storage. For example, the Arduino could record the temperature using the thermocouple and communicate the data with a remote server, such as the cloud or a nearby windows device. The issue lies with the non-transparent communication between these devices and the external data servers. These servers can analyze and redistribute the information without the user's direct consent [1]. In 2018, an Amazon Echo recorded a conversation from a user and then shared it with contacts in her address book without direct consent. The situation was deemed to have been a string of unlikely events after the device woke on a keyword, and listening to the conversation resulting in an indirect conversation between the device and the owner with the recorded message being distributed [8].
With this example, it is important to consider and validate that the data is being sent to the correct locations.

3.2 Privacy of Information

When developing an IoT sensor system it is important to consider the ramifications on society that the data can have. Target, a large American retail corporation, extracted and studied users purchasing habits online and produced profiles for each user. With the data, they selected the clients whom they suspected were pregnant and sent them promotional vouchers for baby items. One client, who was a minor, received these vouchers and even though she was pregnant, her family only became aware of the situation because of

the vouchers [11]. The issue with this situation has two key problems. Firstly, Target was collecting significant client information to be able to target specific categories of people. Secondly, they did not consider the repercussions of there actions on society as they sent inappropriate material to a minor that affected her relationships.
With the information Material Processing Institute wish to collect, it is not nearly as sensitive compared to the example, nevertheless, it is still important to remember that the data can only be used for specific situations and should not be misused or shared with external parties with no investment in the data.

3.3 Best Practices when Securing IoT Devices

Before undertaking the IoT project for Material Processing Institute a number of best practises have been identified.

1. **Physical Protection of Devices** - IoT devices should be isolated and protected from unauthorised personnel. If an intruder gained physical access to the device they could add or remove critical components that affect the usability and security of the device. Furthermore, they could upload different code to distribute the data to alternative servers that are not authorised.

2. **Firmware & Patch Updates** - IoT devices require hardware to monitor the environment surrounding them. The hardware attached must be able to be updated and upgraded when required to prevent malicious attacks taking place.

3. **Fail-Safe Design** - As IoT devices connect to other services using some form of communication technology there must be a fail-safe strategy in place. For example, if the communication connectivity dropped for an alarm system a backup must be in place as lives or property could be in danger. For Material Processing Institute, if the Bluetooth connectivity fails temporarily crucial data could be lost that affects their decision making.

4. **Strong Authentication** - To connect to a Bluetooth receiver the encryption key must be known. When manufactures distribute the Bluetooth receivers they come with default weak passwords such as '1234'. Changing these passwords regularly is good practise as it makes it less likely for attackers to gain access.

4 Design of System

When the system is in operation the IoT devices will be located in isolation boxes attached to the ladle vessel. A nearby windows PC will display the data from each IoT device. If the IoT device uses a Bluetooth module to communicate then the PC will open a network stream and extract the data

each second. Otherwise, if the IoT device sends the data to the online Web API, the PC retrieves new data from the API and present it to the user.

The WiFi device that has been developed is a prototype for future usage once the network connection has been extended to the ladle vessel area. Therefore, the default method of communication is Bluetooth. Bluetooth was identified as the ideal method of communication as it uses less battery power compared to WiFi and only a slightly weaker range.

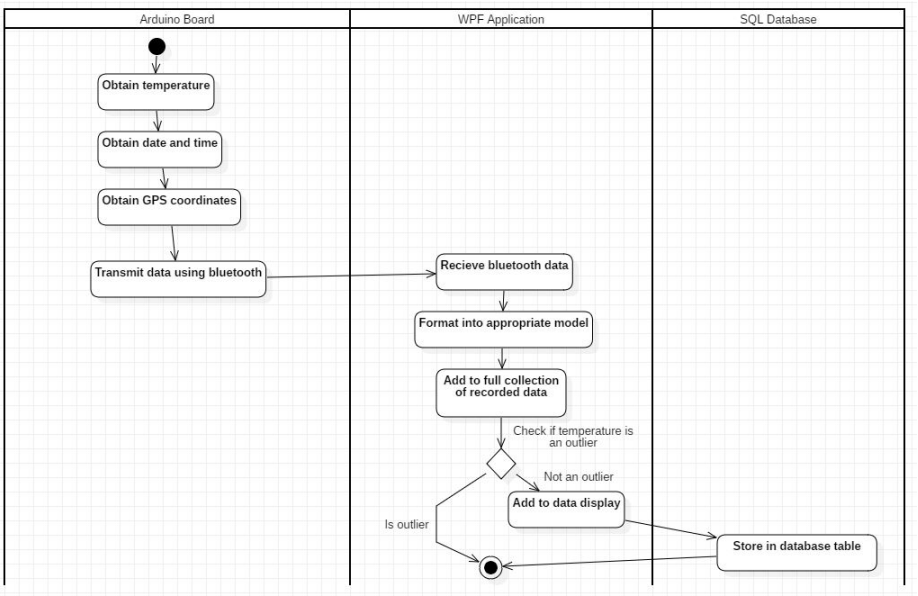

Figure 1 - Shows an activity diagram showing a high level overview of operations between all applications and layers

4.0.1 Hardware Components

Hardware Components	
Hardware Part	**Description**
Arduino Uno	The Arduino Uno is used for the versions of the IoT device that communicate through Bluetooth. Due to the simplicity of the board and the usage throughout other similar projects.
Arduino WiFi Rev 2	The REV 2 is used for builds of the IoT device that communicate through WiFi and transmit the code directly to the Web API.
DSD TECH HC-05	The DSD TECH is the chosen Bluetooth module for the project. The Bluetooth module can run on either 3.3V or 5V, has a large range and can detect if it is connected to other devices.
DS3231 Real Time Clock	The real-time clock allows for accurate date and time recording to the latest second
RoyalTek ITEAD GPS Sheild	The GPS Shield allows for the current latitude and longitude to be recorded.
Type-K Thermocouple	The thermocouple allows for extremely high temperatures to be recorded
MAX31855K Thermocouple Breakout Board	Allows a connection to be established between the Arduino and thermocouple breakout board
Thermocouple Connector PCC-SMP-K	The connector connects the thermocouple to the breakout board
16GB SD Card	Allows for storage of data when a connection fails
LED Lights	To indicate if the device is active or sleeping
Battery Pack	Power supply to power board
KY-022 Infrared IR Sensor Receiver	Receives input requests from the IR remote.
IR Remote	Allows a user to stop the recording of data for the board

4.1 Arduino Design

During the design phase of the project, it was decided that an Arduino would be the best solution due to the plentiful availability and the low cost of the parts. The thermocouple and data communication module (Bluetooth or WiFi) are connected to the Arduino microcontroller and form the basis of the IoT device. Additional components are added to different variations of the core device to evaluate and investigate the IoT's benefits in the steel production environment [5].

The code that is deployed to an Arduino board is called a sketch and is developed in C++. The sketch must include a *setup* and *loop* function which provide the following functionality.

- ***Setup*** - The setup function is executed at the start of the program and is the entry point of the code. The setup function is intended to set global variables and establish connections [3].
- ***Loop*** - The loop function is called directly after the setup function has been completed. The loop function will continuously be recalled by the compiler and the user can insert a delay command to pause the process before repeating again [3].

4.1.1 Base IoT Sensor

A Type-K thermocouple is used for each device in the project because of there ability to record extremely high temperatures and low cost. Additionally, the Type-K has plenty of information online and has been used in other Arduino projects. Other thermocouples were considered such as a Type-J or Type-N but both are more expensive and are less commonly used [7].
The Type-K probe is inserted into a thermocouple connector which is soldered onto the thermocouple breakout board. The breakout board, in this case, is the MAX31855K which is a 14 bit-resolution, SPI-compatible serial interface and acts as a digitizer, converting an analogue reading to digital, and sends the data out using the SPI interface.
The thermocouple breakout board has five pins that must be connected to the Arduino board.

Connecting Breakout Board to Arduino		
Breakout Board Pin	**Arduino Pin**	**Function**
GND	A1	Ground
VCC	A0	Digital pin set to 3.3V (Power)
SCK	D13	Clock
SO	D12	Serial Data Out
CS	D10	Chip Select

The SCK, SO and CS pins are the SPI interface pins and must be connected as specified as the Arduino only has one SPI output.

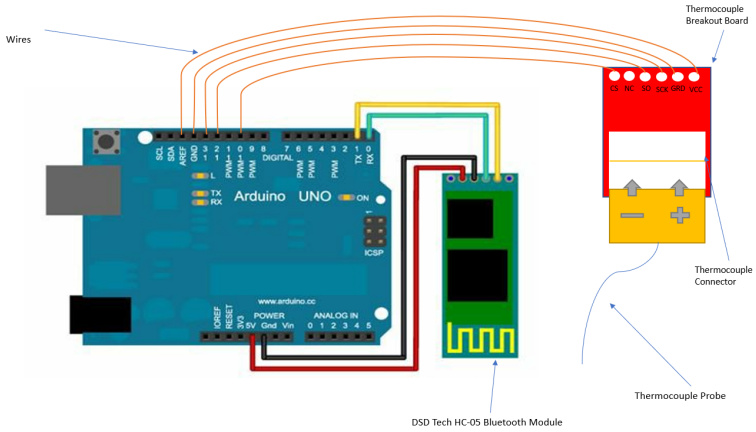

Figure 2 - Shows an illustration of the Arduino Uno board with the thermocouple, breakout board and connector

Each IoT device has a connected DS3231 Real Time Clock Module which allows for the exact date and time the temperature was recorded to be logged. The DS3231 is a power effective solution and has a battery backup for continuous timekeeping. One disadvantage of the module is the time starts from when the code was uploaded. The module takes the time from the uploading device, stores the date and time and continuously increments both. Once the power is restarted, the time is reset to the original time the code was uploaded and repeats.
The Real-Time clock has four pins that require being connected.

Connecting the Real-Time Clock to Arduino		
Real-Time Clock Pin	**Arduino Pin**	**Function**
GND	GND	Ground
VCC	3.3V	Digital pin set to 3.3V (Power)
SDA	SDA	Serial Clock
SCL	SCL	Serial Data

For the IoT devices that require communication using Bluetooth, the DSD Tech HC-05 is attached to the Arduino board. The module comes with a state pin which can be connected to indicate if the microcontroller is communicating with any other devices which is essential for storing the data locally when a drop in connection occurs.
The Bluetooth module has five pins that must be connected to the Arduino module.

Connecting the DSD Tech HC-05 to Arduino		
Real-Time Clock Pin	**Arduino Pin**	**Function**
GND	GND	Ground
VCC	5V	Digital pin set to 3.3V or 5V (Power)
Tx	Rx (D0)	Transmitter to Receiver (Comms)
Rx	Tx (D1)	Receiver to Transmitter (Comms)

4.1.2 Added GPS Shield to Obtain Latitude & Longitude with Additional SD Card Slot

The Arduino Uno only has one set of SPI pins and is already in use by the thermocouple (D12-D13). Many GPS and SD Card modules communicate using the SPI interface. One solution would be to share the SPI interface by changing which SPI device is active on the SPI bus. However, the connection architecture to solve the problem would have drastically extended the life cycle of the project.
To solve the problem, a GPS shield with an SD Card slot was obtained and connected atop of the Arduino. The thermocouple and real-time clock were then reconnected directly to the shield.

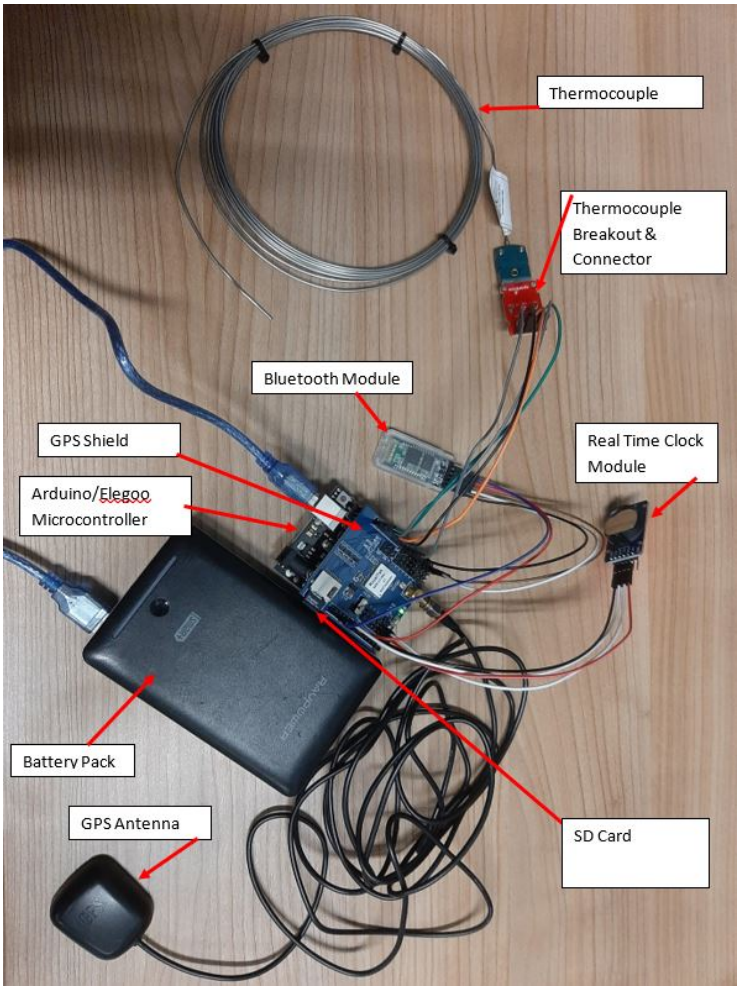

Figure 3 - Shows all of the connected modules and hardware requried to develop the IoT sensor.

4.2 Desktop Application

To allow users to view the data currently and previously recorded a Windows Presentation Foundation (WPF) application was developed. Microsoft originally developed WPF for rendering user interfaces in Windows-based applications [2]. The WPF application was chosen over other available technologies such as ASP.NET MVC, AngularJS & Reach as it can run locally on the desktop of a computer and does not require a network connection to access the application which was a key requirement of the system. Furthermore, the WPF application was able to display the information to the end-users to a high and professional standard.

4.2.1 Application Architecture & Structure

When developing software the solution must be developed to high, professional standards that use the latest technologies and protocols. After researching online I discovered the recommended architecture structure for any WPF application is the Model-View-ViewModel (MVVM) design pattern. The MVVM pattern separates the programming logic into the three different layers [14].

- **Model** - Model's encapsulate application data and represent a domain model. For example, a model could represent a person with properties including the persons name, address, age etc.
- **View** - The view is responsible for defining the structure, appearance and layout of what the user sees on the screen. In WPF applications the view should be defined in the XAML and limited code-behind that does not contain business logic.
- **ViewModel** - ViewModels implement properties and commands that the view can bind to. When the properties are changed the view can be notified to make the change visually. The ViewModel is the bridge between the View and Model layers and is responsible for handling all interactions between the two.

The WPF application follows an N-Tier architecture pattern that has been built around the MVVM design pattern. Other layers have been included to further separate the code to follow a logical, reusable structure. These additional layers are as follows.

- **Repository** - The repository layer is responsible for communicating with the database. The layer allows the business logic to be decoupled from the data access layer.
- **Data** - The data layer contains all the model's needed to communicate with the database. These model classes are Plain Old C# Objects (POCO).

- **BluetoothService** - The BluetoothService is a layer that can search and communicate with nearby Bluetooth devices. The BluetoothService, when reading data from the Bluetooth devices, takes the latest output and looks for certain keys such as "-temp" and extracts the value adjacent to the key, which in this example would be the temperature recorded.
- **Tests** - NUnit tests to validate the correctness of the code.
- **ExcelDataWriter** - The excel data writer takes collections of data and writes the data to excel files on the local computer.
- **CustomDialog** - In WPF the default dialog looks rather old fashioned. The custom dialogue was developed to look more professional and is responsible for alerting the user with information and asking for confirmation of a request before preceding.

4.2.2 Detecting Outliers

While reading the temperature from a Bluetooth device an occasional anomaly occurred were the reading was zero, much lower compared to the previous recordings, and was an outlier. Initially, ignoring any values less than or equal to zero seemed a sensible approach as the metal could not go below zero during the casting process, however, it limited the solution to that specific situation and was not reusable. Eventually, the issue was discovered as periodically the "-temp" key was not found leaving the temperature value as the integer default of zero. By simply making the temperature integer nullable and ignoring any that were null resolved the problem.

Nonetheless, it was decided that an outlier detection system was needed to accurately filter out any readings that could potentially be anomalies (i.e. considerably less or more than previous readings). The first implementation took the last twenty readings of the specified device and calculate the mean average. Then, if the current reading was five degrees lower or greater than the previous average it was considered an outlier and ignored. The problem with this solution was it wasn't fully scientific and allowed for a wide range of error.

After researching online, the descriptive statistics formula, the interquartile range, was identified as the ideal strategy for determining if the latest temperature was an outlier. The new solution took the last twenty temperature readings in order and separated them into three quartiles. Quartile One (Q1) is the median of the first half of the data set, Quartile Two (Q2) is the middle value of the overall set, and Quartile Three (Q3) is the median of the last half of the set. After the median of both Q1 and Q3 are obtained, the median of Q1 is subtracted from Q3 to calculate the interquartile range. Finally, the interquartile range is times by a multiplier and subtracted from Q1's median and added to Q3's median to generate a

lower bound and upper bound. If the current temperature is less than the lower bound or greater than the upper bound it is an outlier and ignored.

Overall, the strategy using the interquartile range was much more effective at removing possible outliers as the bounds could dynamically change based upon the data set it was given instead of being hardcoded to specific values. This dynamic change allowed for more accurate detection with more scientific foundations.

4.2.3 Connection Status Bar

Located at the top of the window is the connection status bar which indicates the connection status of the current Bluetooth device being recorded. There are four main statuses that can occur during the applications life cycle.

1. **Connecting** - The background is orange with a message indicating to wait for the connection to the Bluetooth device to be completed
2. **Connected** - The background is green with a message indicating that the application has connected to the Bluetooth device and is receiving data.
3. **Failed** - The connection to the Bluetooth device has failed and displays and message accordingly. The application will try to re-connect or the user may need to intervene if a more critical error has occurred.
4. **Pause & Stopped** - The user has pressed the stop button in the application or has used the IR Remote to stop the Bluetooth device. This triggers the connection status bar to turn grey.

The connection status bar shows the connection status of the currently selected device on the Live Data screen but is viewable from all different screens in the application to give the user constant updates.

4.2.4 Displaying Live Data

Once the application has loaded the user is presented with the live data screen where the data currently being recorded from each Bluetooth devices is displayed. The display consists of a graph, which plots each of the temperatures on the Y-Axis against the Date & Time on the X-Axis, and a live data feed which shows all of the recorded data as a list output. The data feed for each device can be viewed by selecting the device from the drop-down menu. Furthermore, the devices data feed can be stopped and started by pressing the corresponding buttons below the data feed. The user can add additional information to each record such as comments or a position tag on the ladle shell that the temperature was recorded from. Finally, at run time the user can search for new devices that are currently not being monitored by the system by pressing the "Search For Devices" button.

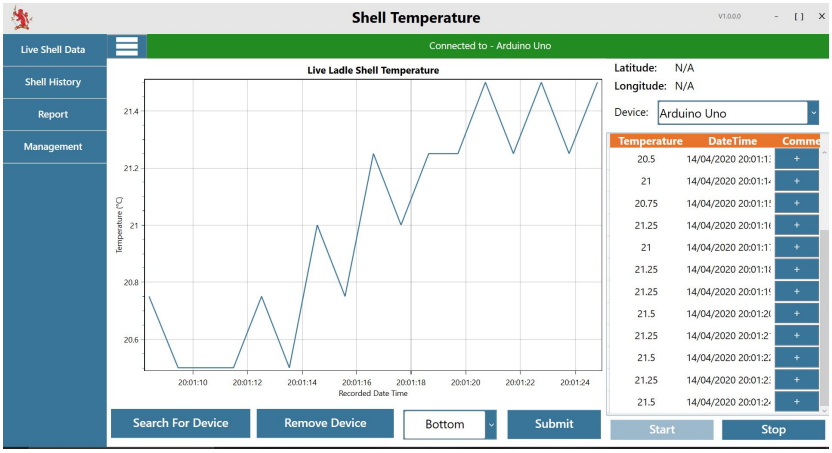

Figure 5 - Shows the live data recording screen of the WPF application

4.2.5 Displaying Historical Data

Another feature that the user can engage with is the Historical Data view were the user can view previously recorded data from different devices. The data is again displayed using a graph output and a data feed. Comments can be added to each of the data readings, updated or deleted and the data currently being displayed can be exported to Excel via a simple button press.

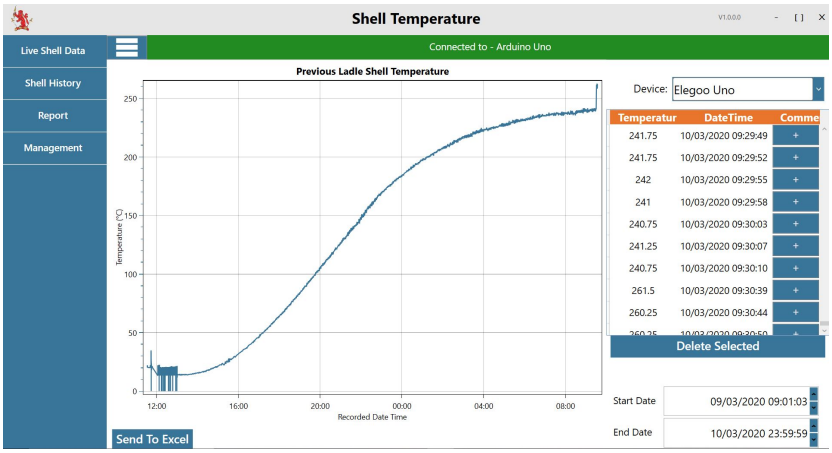

Figure 6 - Shows the data recorded from one Arduino device during the live trial

4.2.6 Calculating Statistical Information

The third and final important view is the statistical report. The statistical report collects all the data readings between a start and end date for a

selected device and calculates statistical information based upon the temperature. The statistics include the mean, median, mode, range, interquartile range, mean deviation and standard deviation of the data. The user can then export the data to excel.

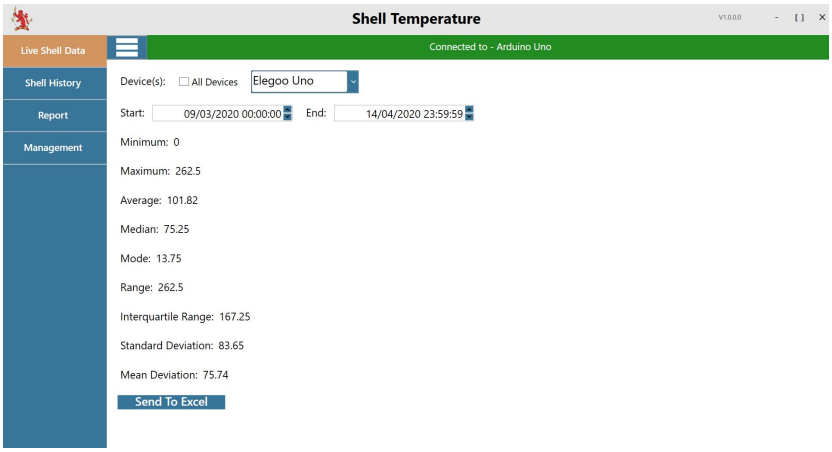

Figure 7 - Shows the report screen where users can see statistical data

4.3 Web API & Android Application

To allow the Arduino boards to submit the data using a network connection an endpoint or service needed to be developed that was hosted on the web. The initial idea was to use Azure IoT or AWS IoT to submit the data to. However, this was not feasible as synchronizing the data between the existing SQL database and these services would have been challenging and too time-consuming. The settled upon solution was a .NET Core Web API hosted on Azure which allowed communication to the existing SQL database. The Web API allows the Arduino devices to submit the data readings and then users can retrieve the data by connecting to another endpoint. The addition of the API added a layer of abstraction between the calling application and database and allowed for more reusability in the future.

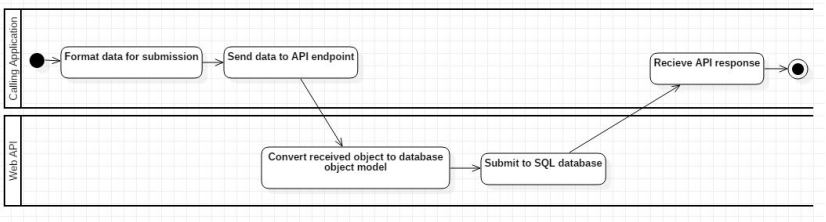

Figure 8 - Shows an activity diagram explaining the submission process from the calling application the Web API, and finally to the SQL database

One of the requirements for the system was to allow access to the data readings on the go. To fulfil this request an Android phone application was developed that retrieves all the latest data readings from the Web API between two date ranges and displays them on-screen to the user.

5 Testing of Solution

5.1 Preliminary & Automated Testing

Throughout the development of the project extensive manual testing was carried out to check and evaluate that the system was working as expected with notes on performance, issues and success status noted down inside a document.

To further verify the correctness of the code and automatically check if the code is behaving as expected, 115 NUnit tests were developed. Each unit test defines a set of mock data, passes the data into the function which returns a result, and finally asserts that result against what is expected. The functionality that has been tested using unit tests includes all calculations, the outlier detector, sorting algorithms and database interaction which covers a large percentage of the total functionality of the application.

5.2 Live Deployment Testing

On the 09/03/2020 at 11:32 AM the Arduino devices and application were deployed and successfully recorded the ladle shell temperature for 22-hour period until 10/03/2020 09:33 AM. One Arduino Uno's attached thermocouple was connected to the lower, bottom outer side of the vessel. In contrast, the other Arduino Uno's thermocouple was attached to the inner edge of the vessel. Over the 22 hours, both Arduino's recorded temperatures began near 13°and gradually rose until the ladle shell was fully heated. The top temperature for the lower thermocouple risen to 262°while the top thermocouple rose to 1217°.
Overall, the live trial was a success as the system managed to record the temperature for the entire period without drops in connection. However, after reviewing the data using the history feature, the early recordings by one of the devices shows that the temperature keeps fluctuating between 13°and 25°for the first hour. It was later realised that old data from the SD card was being transmitted to the application, but it did not have a recorded date & time associated with it, therefore, the application assigned it the computers current date and time and submitted it to the database leading to inaccuracies.

Figure 9 - Shows an image of the system recording the ladle vessel. The technology is positioned on the nearby table with the thermocouples attached to the outer edge of the ladle vessel

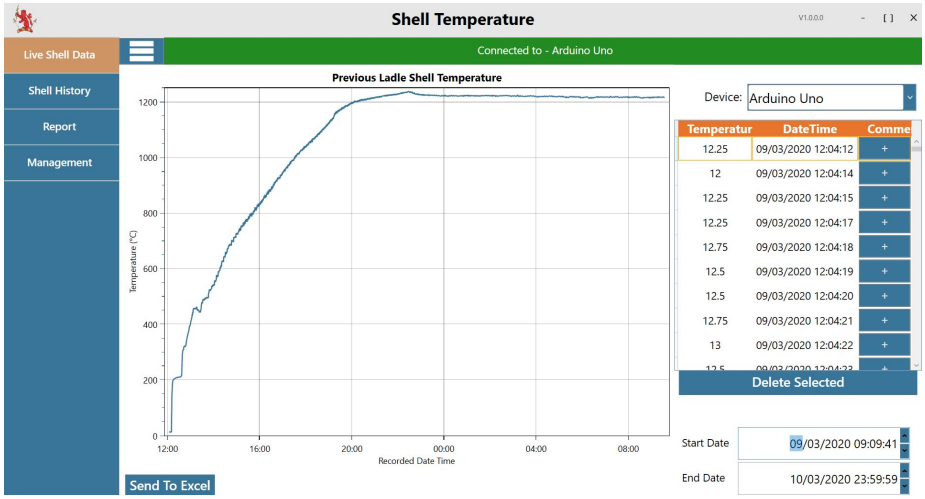

Figure 10 - Shows the data recorded from the thermocouple recording the top section of the ladle vessel

6 Evaluation & Reflection of Project

6.1 Overall Success and Achievements

Overall, the Arduino system and WPF Application were developed to a high standard and provided all the functionality required in the specification. The success of the project is reflected in the live deployment, as the system did not falter during the data logging process.

During the research and design phase of the project, I was unable to find any other direct implementations of Arduino microcontrollers recording live ladle shell temperatures in metallurgy. For me, developing this solution is something rather unique that may not have been completed before and am rather proud of my contribution to the IoT.

The aspect of the project that I am most proud of is the development of the Arduino boards and code. Before the project, I had no experience with microcontrollers and this project has exposed me to the wider world of the IoT. In particular, the part that I enjoyed most, was connecting and soldering the different modules to the Arduino board as it was something I've never experienced before.

6.2 Difficulties & Challenges

One major problem that was overcome during the development process was the reading of data from the SD card. The difficulty with the task was two-fold. Firstly, the WPF application already had a thread for each device to capture

the live data, so it seemed redundant to create a second thread for each device to check if SD card data was being transmitted. Therefore, the SD card data and live data needed to be outputted together. The Arduino library available to read the SD card data only had the inbuilt functionality to read each line individually and could not remove it once read. The second issue was the fact that the Bluetooth connection could drop whilst reading the data from the SD card if this event occurred then the line (position) inside the SD card that had been reached would be forgotten. Once the connection resumed again the data extraction from the SD card would start at the beginning again as there is no functionality available to jump to a specific position. The solution was to output the live and SD card data simultaneously at the same moment. To overcome the problem of the connection dropping during the data reading of the SD card a loop was added inside the function reading the SD card to write to the file if the Bluetooth connection failed. Overall, the solution to the problem took a considerable amount of time to achieve and much longer than I anticipated.

6.3 Improvements, Future Development & Research

One improvement that could be made is to add a service layer to the WPF application to communicate with the database. When the application is running in development mode the service layer would call the repository layer directly to obtain the needed data. If the application was running in live mode then the service layer would call the online Web API to obtain the needed data. This improvement would further increase the scalability of the solution and make more use of the hosted API.

One issue with the Web API is that there are no security protocols in place to prevent unauthorised access using any of the endpoints. Currently, if any user tried to send or retrieve data from the API they could do so without any credentials. Without validating the users identity a major security issue has been exposed to the general public and attackers. To solve this problem, an IndentityServer should be developed that requires users to register, login and pass a token to the endpoint to validate their identity.

As aforementioned, the development of similar Arduino recording systems is almost nonexistent so this could provide an interesting line of future research. Primarily investigating and developing Bluetooth topology in production environments.

7 Acknowledgments

I wish to pay gratitude to the Material Processing Institute for allowing me to use their facilities and continued support throughout the project. I would also like to thank my project supervisor Victor Chang for his support and encouragement throughout the project.

References

[1] F. Allhoff and A. Henschke. The internet of things: Foundational ethical issues. *Internet of Things*, 1:55–66, 2018.

[2] C. Anderson. *Essential Windows Presentation Foundation (WPF)*. Addison-Wesley Professional, 2007.

[3] S. A. Arduino. Arduino. *Arduino LLC*, 2015.

[4] H. F. Atlam and G. B. Wills. Iot security, privacy, safety and ethics. In *Digital Twin Technologies and Smart Cities*, pages 123–149. Springer, 2020.

[5] Y. A. Badamasi. The working principle of an arduino. In *2014 11th international conference on electronics, computer and computation (ICECCO)*, pages 1–4. IEEE, 2014.

[6] S. Balaji and M. S. Murugaiyan. Waterfall vs. v-model vs. agile: A comparative study on sdlc. *International Journal of Information Technology and Business Management*, 2(1):26–30, 2012.

[7] R. Bentley. Irreversible thermoelectric changes in type k and type n thermocouple alloys within nicrosil-sheathed mims cable. *Journal of Physics D: Applied Physics*, 22(12):1908, 1989.

[8] N. Chokshi. Is alexa listening? amazon echo sent out recording of couple's conversation. *The New York Times*, 25, 2018.

[9] D. Fletcher. Internet of things. *Evolution of Cyber Technologies and Operations to 2035*, page 19, 2015.

[10] B. Glaser, M. Görnerup, and D. Sichen. Thermal modelling of the ladle preheating process. *steel research international*, 82(12):1425–1434, 2011.

[11] K. Hill. How target figured out a teen girl was pregnant before her father did. *Forbes, Inc*, 2012.

[12] D. Lu and T. Liu. The application of iot in medical system. In *2011 IEEE International Symposium on IT in Medicine and Education*, volume 1, pages 272–275. IEEE, 2011.

[13] R. Muñoz, R. Vilalta, N. Yoshikane, R. Casellas, R. Martínez, T. Tsuritani, and I. Morita. Integration of iot, transport sdn, and edge/cloud computing for dynamic distribution of iot analytics and efficient use of network resources. *Journal of Lightwave Technology*, 36(7):1420–1428, 2018.

[14] J. Smith. Patterns - wpf apps with the model-view-viewmodel design pattern, Aug 2016.

[15] T. Xu, J. B. Wendt, and M. Potkonjak. Security of iot systems: Design challenges and opportunities. In *2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 417–423. IEEE, 2014.

[16] K. Yordanov, P. Zlateva, I. Hadzhidimov, and A. Stoyanova. Testing and clearing the high temperature module error from 0 to 1250 c for measurement with 16 k-type thermocouples. In *2018 20th International Symposium on Electrical Apparatus and Technologies (SIELA)*, pages 1–4. IEEE, 2018.