Lab 1 – ParkODU Description

Matthew Stevenson

CS 411W

Professor Thomas J. Kennedy

February 26th, 2018

Version 2

Table of Contents

1 Introduction 3
2 Product Description 5
2.1 Key Product Features and Capabilities 5
2.1.1 Vehicle Count and Displays 5-6
2.1.2 Navigation 6
2.1.3 Management 6-7
2.1.4 Other 7
2.2 Major Components (Hardware/Software) 7-9
2.2.1 Language: Java 9
2.2.2 Framework: Spring Framework 9
2.2.3 IDE: IntelliJ Community Edition 10
2.2.4 Build Tools: Gradle 10
2.2.5 Data Stores: Hazelcast, MongoDB 10
2.2.6 Version Control: Git 10
2.2.7 Third-Party API: Google Maps API 11
3 Indentification of Case Study 11
4 Product Prototype Description 11
4.1 Prototype Architecture (Hardware/Software) 12-13
4.2 Prototype Features and Capabilities 13-14
4.2.1 Risks 15-20
4.3 Prototype Development Challenges 21
5 Glossary 21-22
6 References 23

List of Figures

Figure 1 - Major Functional Component Diagram (Complete Product Version) ….……....7

Figure 2 – Major Functional Component Diagram (Prototype Version) ……..……11

List of Tables

Table 1 – Features of ParkODU Prototype and Real-World Product (RWP) ....…….14

Table 2 – Risks (Technical) ……..…15,16

Table 3 – Risks (Customer) ……...17,18

Table 4 – Risks (User) ……..…….19,20

[This section intentionally left blank]

1. Introduction

Old Dominion University (ODU) encounters parking complications based on inadequate parking services and lack of proper management of those resources. Accompanied by a high student population consisting of both resident and commuting drivers, it is difficult for such services to correctly propagate an immediate solution without a tool to increase efficiency of those resources. Unfortunately, the effects of inadequate parking conditions produce a source of stress for students, staff, and visitors in which ParkODU will address and mitigate towards a more efficient solution.

According to ODU's main website, enrollment in 2017 at ODU was approximately 24,828 students. Among this collection, seventy six percent of students live off-campus, which includes students who take classes online, but out of the seventy six percent, the majority does commute. (2) ODU currently offers 5 parking garages specified for faculty, metered, commuters, and other, totaling 3013 spaces. (4) However, roughly 9,400 student commuters need to park at ODU daily while 1511 faculty members (835 Full Time, 676 Part Time) must also park. (3) A student from ODU gave his statement based on the current parking conditions at ODU and states, "Parking at ODU sucks, there are not enough spaces for everyone and if you are a commuter you better get to class an hour early if you want a spot. It is like The Hunger Games for parking spaces. May the odds be ever in your favor." (1)

While encountering the problem related to parking, considerations have been made to create additional parking by constructing more garages. However, due to limited geographic constraints based on property to which ODU currently owns, allocated space to build new parking is limited. Additionally, ODU displays a priority to building

academic buildings over parking structures. Another important factor is the cost of a new garage. The International Parking Institute (IPI) states the national average to build a parking garage is 8.56 million dollars. (6) Based on both relative cost and limited geographic location, ODU should seek a solution to maximize efficiency of their resources given the current state of parking demands. Without improvement, drivers will continue having trouble finding parking spaces, during the hours of 10:00AM - 2:00PM, due to: lack of signage and notifications for available spaces, preferences for specific parking locations, and limited choices during peak hours. Better management of these resources creates an opportunity to promote future planning and allows better knowledge of services how they are being utilized.

ParkODU is a web application catered to drivers that need to park at ODU while offering a management resource that displays detailed usage statistics and analysis based on historical data conditions. The interface will allow users to view all parking information related to ODU, so they may choose the best parking decision based on their needs. The goal is to automate and decrease time spent manually searching for the optimal parking spot while increasing efficiency. The software solution analyzes parking availability in real-time and helps drivers find best vacant parking space closest to their destination. The application will optimize parking as it includes starting location, permit type, and destination to allow the user to find the best parking space available, saving time, resources, and effort. The prototype will use a simulated garage designed by the team founding ParkODU which will demonstrate the functionality of the application so that it is ready for deployment and usage at ODU.

2. Product Description

ParkODU is collection of tools built into a web application for drivers to find available parking spaces on campus that also gathers parking space usage information for ODU Transportation and Parking Services. It will be compatible with all major web browsers and will also be available as a downloadable application for Android and iOS platforms. ParkODU will compile data gathered in real time by various vehicle counting systems installed in garages and parking lots and make the information available to drivers. ParkODU will be designed in a matter that allows flexibility in the customer to choosing their desirable vehicle counting hardware and offers interfacing compatible to whichever method is used. The main objective of ParkODU is to inform drivers of vehicle counts in real-time, so drivers can allocate their time more efficiently without dealing with the uncertainty of parking conditions. The secondary objective of ParkODU is to provide ODU Transportation and Parking Services with usage data to facilitate efficient future strategic planning and allocation of resources.

2.1. Key Product Features and Capabilities

ParkODU offers multiple options in how data is presented to a user. The functionality will give the user ultimate control to whichever functionality they choose to utilize and will present information calculated in real-time offering the most up-to-date conditions while promoting the best possible decision.

2.1.1. Vehicle Counts and Displays

ParkODU will populate displays for all parking showing both graphs populated with daily capacity percentages and floor plans depicting what spaces are currently available. The

color red will indicate that a space is occupied, and the color green will indicate that a space is vacant.

- Tracks and monitors garage parking availability in real-time
- Counts vehicles by garage, floor, and space will be the specified view type to which a user can view data based on which parking resource they wish to view
- Displays a detailed floor plan of each garage floor and the available spaces
- Displays average vehicle counts at each location by time of day
- Allows users to filter garages, floors, and space by their parking permit type
- Allows users to sort garages by walking travel time to another building

2.1.2. Navigation

ParkODU will provide the functionality for users to use navigation services to direct them to their desired destination.

- Utilizes Google Maps navigation features to access geolocation while calculating arrival time and displaying travel distance given in the search feature
- Predicts future vehicle counts based on calculation of historical data compared to vehicle exit and entry rates
- Allows faculty and students to import their schedule into the application for closest parking space recommendations

2.1.3. Management

ParkODU will provide a management interface through which will alter the configuration of a specific implementation.

- Allows the addition, editing, or removal of a garage, floor, or space and is only available to the management portion of ParkODU

- Allows the addition, editing, or removal of user roles and is only available to the management portion of ParkODU
- Provides easy configuration of occupancy signage commonly displayed on the outside of a garage for drivers to view parking conditions not utilizing ParkODU
- Sends notifications of the events based on ODU event schedules that affect parking
- Interfaces with any vehicle detection system which gives the customer ultimate control based on their needs

2.1.4. Other

ParkODU will offer additional features to accommodate as new technology emerges.

- Utilizes open source software
- Mobile support planned for future releases

2.2. Major Components (Hardware/Software)

The flow of information starts from the vehicle detection technology of customer's choice; it could be IR sensors, inductive loops, IP cameras, or other means of detection. Devices, such as IR sensors, are installed at each parking space and are capable of counting by space, which correlates to the system detecting vehicles by individual parking space. However, devices such as inductive loops are constrained to count-by-floor mechanisms that do not provide an accurate individual representation of the availability of each parking space because they are commonly installed at the entry and entry points. Inductive loops also do not offer the flexibility the determine differences in permit type allocation because of suggested installation methods. Installation of inductive loops at each individual parking space is infeasible based on cost and

considered a negligible option for most customers. Vendors of vehicle detection technology gather data from their devices and store them on their server to be used to interface with ParkODU. Figure 1 offers a visual representation between hardware and software interactions while giving the user a fast, efficient web application to navigate and browse parking conditions.

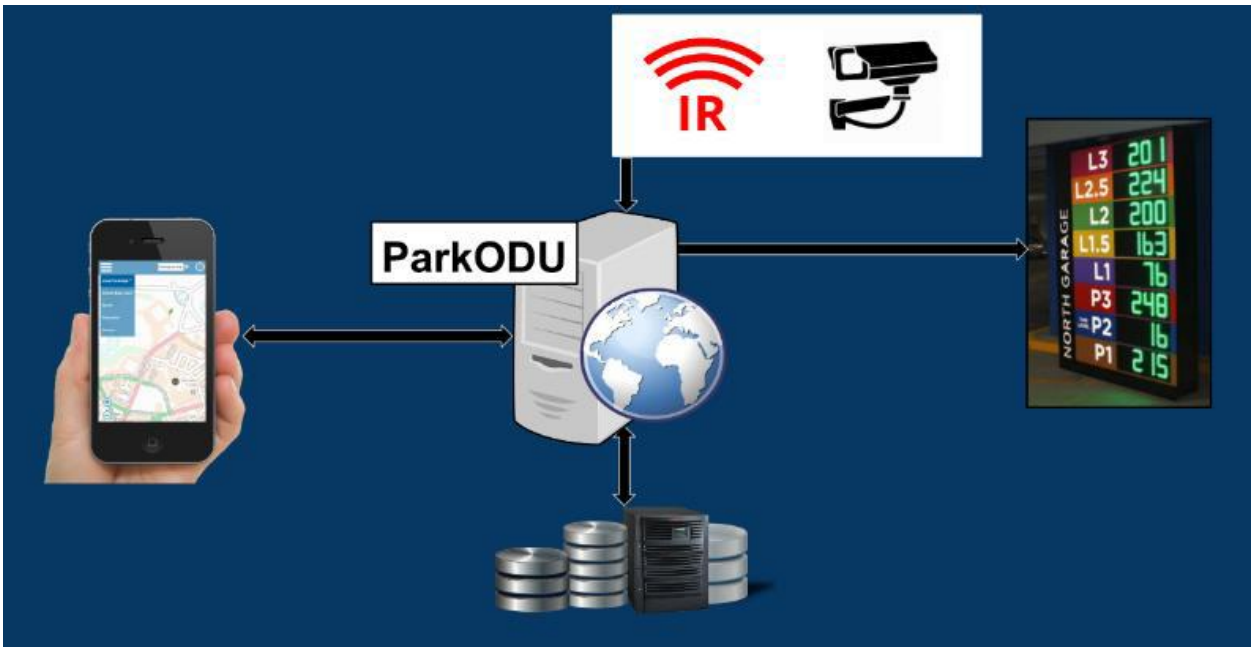

Figure 1. Major Functional Components Diagram (Complete Product Version)

ODU would normally access the vendor's web portal to access the occupancy data. However, vendors also provide APIs, which allows ODU's software developers to create their own app or interfaces to handle formatting and keep data consistent. The data obtained from the vendor's APIs will be sent to ParkODU via requests to REST endpoint. The data will be stored in Hazelcast's in-memory data grid for fast queries and MongoDB will be used as the secondary and backup data storage. The database and the web application will be installed on physical or virtual machines provided by the customer. In situations where the customer fails to provide physical machines, the ODU

CS department virtual machines will be used as the servers. ParkODU database and application servers will service the web app, Android/iOS app, and the digital signs also uses REST endpoints.

ParkODU will be hosted on a server or cluster of servers. These servers can be physical or virtual. ParkODU uses the installed vehicle detection technology to update garage counts and floor plans. ParkODU stores data in MongoDB, an open source database. This information can then be displayed on signage or accessed via the web on a smartphone or computer.

2.2.1. Language: Java

Java is a coding language that builds into a Java Virtual Machine (JVM) once compiled. This allows for flexibility and portability so that the code can run anywhere and is self-contained once built. Java is a popular coding language based on its similar correlation to C++ which most Computer Science students learn as a core coding language.

2.2.2. Framework: Spring Framework

Java Spring Framework is best described as, “The Spring Framework is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE (Enterprise Edition) platform. Although the framework does not impose any specific programming model, it has become popular in the Java community as an addition to, or even replacement for the Enterprise JavaBeans (EJB) model. The Spring Framework is open source.” (12)

[This section intentionally left blank]

2.2.3. IDE: IntelliJ Community Edition

IntelliJ Community Edition is the open-source version of IntelliJ's product line that mainly focuses with Java development. This utility allows the developer an environment to write, build, and run code with many integrations to Git and other plugins.

2.2.4. Build Tools: Gradle

Gradle is a build automation system that derives its roots from Apache Ant and Apache Maven while using Domain Specific Language used by Groovy. Gradle can recognize which parts of a build are already up-to-date and offers the flexibility for incremental and multi-scale builds.

2.2.5. Data Stores: Hazelcast, MongoDB

Hazelcast is an open source in-memory data grid based on Java. Hazelcast provides high-availability access to frequently used data while offering scalable solutions in a multi-node environment. The data grid created by Hazelcast offers redundancy and increases performance.

MongoDB is an open source database solution that uses JSON documents. Document objects can vary, however, are able to adapt automatically. Mapping is distributed within the MongoDB framework which allows the opportunity for flexible and horizontal scaling.

2.2.6. Version Control: Git

Git an open source version control content solution that provides an interface for collaboration when a team is working together on a project. Git offers the availability to control changes and keep each member up-to-date with the most recent revision based on source code uploaded to the hub.

2.2.7. Third-Party API: Google Maps API

Google Maps API interfaces with web programming and Google Maps. This tool maps the functions usually involved in Google Maps and offers a direct method to interacts with such feature.

3. Identification of Case Study

ParkODU is being developed for the department of Transportation and Parking Services at Old Dominion University. Due to factors such as geographical constraint and academic buildings having priority over parking lots, ParkODU offers a more effective and less costly solution to parking. Signage accompanied outside each parking garage will offer easier navigation and decisions for any driver, especially those visiting, to know about vacancies that choose to not utilize the web application. Occupancy signage offers an alternative solution to display parking conditions based on garage. ParkODU offers the potential to be adopted by other universities or organizations who desire a more efficient solution involving utilization and management of parking resources.

4. Product Prototype Description

The prototype of ParkODU display all defined features and capabilities of the web application, the native Android application, the native iOS application. Digital signage will not be displayed within the prototype; however, the simulator will test the REST endpoint written to display current conditions per garage. The data that would normally be obtained from vehicle detection systems such as inductive loops, IR sensors, and IP cameras, will be simulated also by the simulator. The ParkODU prototype will compile

the simulated data to demonstrate all the features and capabilities of the web application, the native Android application, and the native iOS application.

4.1. Prototype Architecture (Hardware/Software)

The prototype will build a correlating conception to how the Real-World Product (RDP) will interact once it is deployed into an implemented environment. The simulator will be a key tool in accessing correct product implementation and that all features work as they are intended. Endpoints will manage each individual parking space displayed for the user view through a REST controller that interacts with objects inlayed within Hazelcast in-memory data grid. At each input event, this will correlate a REST update that will reflect within the object involved at each endpoint. Once an input event is updated, that object can then be used for calculations and other queries done while a user interacts with the ParkODU application. In Figure 2, interaction placement of the simulation interacting with the prototype at a top-level view.

[This section intentionally left blank]

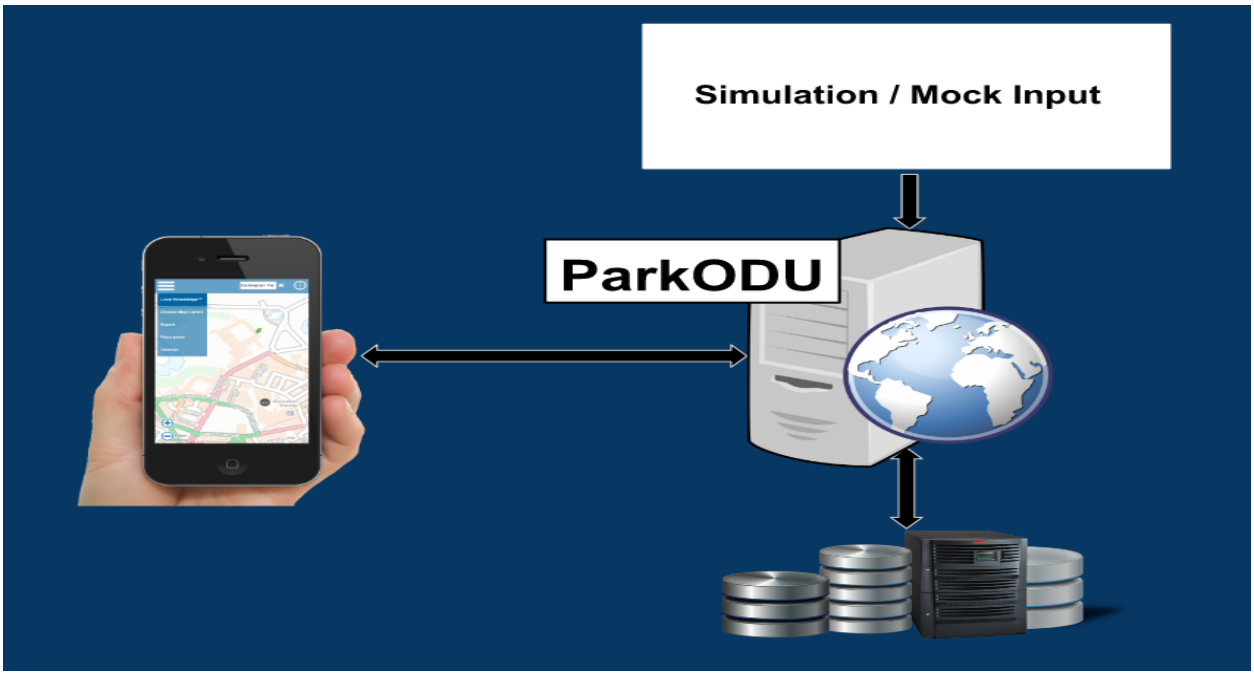

Figure 2 – Major Functional Component Diagram (Prototype Version)

The hardware for vehicle detection will be simulated by a REST client application. The application will send requests to the ParkODU REST endpoints and update the vehicle counts. The application will closely mirror the actual ODU parking traffic during weekday normal hours and simulate special events on weekends. The ParkODU web app and native Android/iOS App will perform operations on the simulated data to demonstrate the features and capabilities.

4.2. Prototype Features and Capabilities

The ParkODU application prototype will simulate an input and display a real-time vehicle count by floor in every garage. The prototype will provide the detailed floor plan along with navigation to the vacant space. The user will be able to import his/her schedule and the application will generate the nearest parking options. It will also analyze the users previous parking data in for improved recommendations.

The functionality and purpose are vital in promoting an improved parking experience at ODU. By providing the user with a live display of parking spaces available by garage they can quickly and effectively locate parking. Reduced traffic flow will be a by-product due to users efficiently finding their desired parking space from the knowledge provided by ParkODU. Table 1 offers are representation of the functionalities of ParkODU between the prototype and real-word product.

Feature	RWP	Prototype
Real-time vehicle counts on every level of each garage	*	*
Display floor plan to show counts by space on each floor	*	*
Display average vehicle count at each location by time of the day	*	*
Allow users to sort garages by walking travel time	*	*
Allow users to filter garages, floors, and space by their parking permit type and space types	*	*
Allow ODU parking staff to configure parking garages, floors, and spaces.	*	*
Provide directions to each garage from user's current location	*	*
Predict future vehicle counts based on the current and historical traffic pattern	*	*
Upload special event schedules and allow the apps to display notification to end users	*	*
Send data to digital signs at the entrance of every garage	*	

Table 1 – Features of ParkODU between the Prototype and the Real-World Product

The ParkODU prototype will support all features and capabilities of the real working system. All the features of ParkODU are designed to provide transparency to ODU parking availability; this will help drivers avoid full parking garages and go directly to parking garages that have available parking spaces. The prototype will demonstrate ParkODU's ability to accurately process the data received from the vehicle detection devices and display the results to end users.

4.2.1. Risks

Depicted in the Tables 2, 3, and 4 is a detailed overview of the assessment of risks and mitigations. In determining these risks, factorization between the impact and the probability of the risk occurring are fundamental drivers for the suggested mitigations to resolve a risk that may potentially occur.

4.2.1.1. Technical Risks

Technical risks need to be considered for the prototype of ParkODU to guarantee functionality is not altered by mitigations given as possible solutions. These factors include functionalities pertaining to the hardware and software used to implement ParkODU. Table 2 details elements considered technical risks.

[This section intentionally left blank]

4.2.1.1. Technical Risks(Continued)

Risk	Description	Mitigation	Impact	Probability
(T1) Web Connection Failure	Application server fails to connect to the Web.	• Test the connection and ensure communication is regained.	Very High	Low
(T2) Database/Web Application Failure	Database/Web App failure may occur due to network settings being offline or unavailable.	• Verify the database and software communication through testing. • Establish dedicated clustered server environments for both database and web application server clusters to reduce possible downtime of ParkODU.	Very High	Low
(T3) Software Bugs	Software development opens up the possibility for bugs that may reduce functionality of the Web application.	• Software updates and debugging techniques will be administered routinely. • User Interface/User Experience (UI/UX) Testing • Regression testing and continuous integration	Very High	Very Low

Table 2 – Risks (Technical)

[This section intentionally left blank]

4.2.1.1. Technical Risks (Continued)

(T4) Hardware Failure	Hardware including IR sensors, Garage Signage (optional), may not be functioning or require repair.	• Mark spot as hardware malfunction and send a service request to maintenance.	High	Medium
(T5) Failure to Notify User of an Event	ParkODU is not updated with event schedules that may affect garage availability.	• Ensure ParkODU is updated with upcoming events. • Create a scheduled task to poll the event calendar through rest endpoints.	Very High	Low
(T6) Lack of Technical Knowledge	Minimal technical experience and/or programming familiarity needed to develop the application.	• Individually improve programming knowledge and provide training to less experienced members in area of deficiency.	Medium	Medium
(T7) Incompatible Input Format	Formatting of input is not compatible with ParkODU.	• Verify input formatting compatibility through testing.	Medium	Medium
(T8) Inability to Scale Under Load	As volume or customer count increases the database becomes slow or may fail.	• Expanding computing resources to handle the exponential growth of work with the use of database scalability. • Establish dedicated clustered server environments for both database and web application server clusters.	Very High	Very Low

Table 2 – Risks (Technical)

4.2.1.2. Customer Risks

Customer risks include possibilities to which the customer will choose a course of action depending on their situation, however, ParkODU will be able to adapt to the customers needs based on certain conditions listed in Table 3.

Risk	Description	Mitigation	Impact	Probability
(C1) University Implements a Better Solution	The university implements a solution other than ParkODU.	• Show the customer how ParkODU's benefits and features are superior to competing solutions. • Offer ParkODU as an open-source solution.	Very High	Very Low
(C2) University Does Not Allow Access to Network	ODU ITS does not allow ParkODU to run on the university's network.	• Some departments within the university run things on their own networks. • Customer determines hosting location.	Low	Very Low
(C3) Customer Unable to Maintain Servers/Hardware	Customer will be unable to maintain the hardware utilized by ParkODU.	• Customer determines the most effective hardware solution with their implementation for ParkODU.	High	Medium

Table 3 – Risks (Customer)

[This section intentionally left blank]

4.2.1.2. Customer Risks (Continued)

(C4) University Replaces All Garages and Lots with Other Buildings	All parking garages and lots are replaced by buildings.	• Parking will still be essential. The software will allow for reconfiguration as the university changes parking allocations.	Very High	Very Low
(C5) Customer Unwilling to Purchase Hardware	Customer may not agree to purchase hardware.	• Software will allow for multiple hardware implementations. • ParkODU will allow for manual toggling of parking space availability.	Very High	Medium
(C6) Parking Lot Replaced by Parking Garage	The University builds parking more parking garages in place of parking lots.	• Ability to add/edit/delete parking objects.	Low	Low
(C7) Customer Purchases Partially Compatible Technology	Customer purchases detection hardware that does not support a certain functionality, such as count by space.	• The software can be reconfigured to support specific customer implementation.	Medium	Medium

Table 3 – Risks (Customer)

[This section intentionally left blank]

4.2.1.3 User Risks

Risks associated to users that may utilize ParkODU is detailed in Table 4.

Risk	Description	Mitigation	Impact	Probability
(U1) User is Distracted While Using the Application	User is distracted while using ParkODU.	• Provide safety notification. • Allow for ParkODU to auto-refresh in order to display current data without any additional user interaction.	High	High
(U2) No Internet Device	The end user does not have access to an internet device, such as a Smartphone or a computer, to use the mobile or web application.	• User is able to view occupancy signage while on campus. • User can use public resources such as a public library computer to access ParkODU. • User can utilize ParkODU's historical prediction feature to print future projections.	Very High	Very Low
(U3) User Cannot Find a Parking Spot	All parking that is being monitored by the application is full.	• Inform the user parking is full and application will notify ODU Parking Services. • Provide ODU Parking Services contact information.	Very High	Very Low

Table 4 – Risks (User)

[This section intentionally left blank]

4.3. Prototype Development Challenges

The challenges while completing the objectives of the prototype will come from obtaining knowledge for various development tools, services and technology. The development of the ParkODU prototype will require substantial knowledge in MongoDB, Hazelcast, Java programming, RESTful web services, web programming (HTML5 and JavaScript), and native App development for Android and iOS. It is expected that every team member will devote a significant amount of time in personal research and mentoring to understand all the tools and services required to complete the objectives of the ParkODU prototype. Members will be expected to conduct such research within their own timeframes as time is not allotted within the normal class time. Research of these topics will require much dedication to determine successful completion of the product. Another challenge of the prototype development is time constraint due to the complicated nature of the solution and many requirements that must to be met. Given the knowledge needed for development, research of the subject matter is fundamental toward the success and completion of the prototype.

5. Glossary

Administrator - a special user with access to additional tools for user account and space management

Agile - a methodology that anticipates the need for flexibility and applies a level of pragmatism into the delivery of the finished product

Best Garage - the closest garage to the destination building with the specified minimum number of available spaces

Driver - anyone who drives and parks at ODU

Driver Entry Rate - the number of vehicles entering the garage each minute

Driver Exit Rate - the number of vehicles exiting the garage each minute

Event - an occasion which affects garage and/or space availability

Garage Rate - Driver Entry Rate - Driver Exit Rate (a positive number denotes that the garage is filling up)

Operating Hours - 7:00AM - 10:00PM

Permit - a physical decal that specifies in which spaces the vehicle is allowed to park

Predictions - a guess based on current and historical data about garage space availability

Real-time - current time

Reconfigurable - software-based creation, deletion, or editing of spaces, floors, and garages

Rush Hours - 7:45AM - 9:00AM, 12:00PM - 1:00PM, 3:00PM - 4:30PM

Sensor - any device which indicates to the software whether a space is occupied or not

Signage - signs that indicate the number of available spaces

Statistical Analysis - the ability to use sample data to form predictions

User - an entity using Park ODU

Vehicle Detection Technology - any device which indicates to the software that a vehicle has entered a specified area

[This section intentionally left blank]

6. References

"Access Automation Car Park Count Systems". (n.d.). Retrieved October 10, 2017, from http://www.access-automation.co.uk/car-park-count-systems.

Agile [Digital image]. (2017, May 8). Retrieved November 29, 2017, from https://www.codingmart.com/uploads/post/image/57e0c0488ca7853c76dd986e/Agile_Development_Process.pngvehicle-coun. (F.4.)t

Burr, David W. "Is University Parking a Common Grievance?". Parking Today Media. September 2011. http://www.parkingtoday.com/articledetails.php?id=1072. September 2017. (8)

"Car counting solutions". (n.d.). Retrieved October 10, 2017, from http://www.puretechsystems.com/solutions-car-counting.html. (9)

"Hazelcast the Leading In-Memory Data Grid" Retrieved January 23rd, 2018 from https://hazelcast.com

"How Much Does a Parking Garage Cost?" Retrieved November 02, 2017, from http://www.parking.org/2016/01/19/tpp-2013-09-how-much-does-a-structure-cost/. (6)

"IntelliJ IDEA: The Java IDE for Professional Developers by JetBrains." *IntelliJ IDEA*, Jet Brains, Retrieved January 18th, 2018, fromwww.jetbrains.com/idea/.

"What Is MongoDB?" Retrieved on January 23rd, 2018 from *MongoDB*, www.mongodb.com/what-is-mongodb.

"Operating Budget and Plan. Old Dominion University". Retrieved November 02, 2017, from https://www.odu.edu/content/dam/odu/offices/budget-office/docs/opplan2017.pdf. (5)

"ODU Campus Parking Map". Retrieved October 23, 2017, from https://www.odu.edu/content/dam/odu/offices/parking-and-transportation-services/docs/odu-student-parking-map-mm.pdf. (F.1.)

"Parking and Traffic Procedures". Old Dominion University. Retrieved November 02, 2017, from https://www.odu.edu/content/dam/odu/offices/parking-and-transportation-services/docs/parking-transportation-rules-and-regulations.pdf. (4)

"Providence Place mall enhances parking garage with $20M in improvements". (2016, December 15). Retrieved October 30, 2017, from https://pbn.com/providence-place-mall-enhances-parking-garage-adds-more-pay-stations-improves-signage119194/. (F.3.)

Rogers, Emily. “Dear Future ODU Students”. (2017, August 28). Retrieved November 02, 2017, from https://www.theodysseyonline.com/dear-future-odu-students. (1)

“Solutions: vehicle counting”. (n.d.). Retrieved October 10, 2017, from http://www.t2systems.com/solutions/vehicle-counting. (10)

“Spring: the source of modern java by Pivotal” Retrieved January 23rd, 2018 from http://spring.io

Team Gold. “ParkODU.” December 2017. PowerPoint presentation.

“The Problem at Hand - The Expansion of Parking At Old Dominion University”. (n.d.). Retrieved November 02, 2017, from https://sites.google.com/a/odu.edu/the-expansion-of-parking-at-old-dominion-university/home/the-problem-at-hand. (2)

“University Facts & Figures. Old Dominion University”. Retrieved November 02, 2017, from https://www.odu.edu/about/facts-and-figures. Accessed November 1, 2017. (3)

“Vehicle Counter”. (2016, February 12). Retrieved October 10, 2017, from https://www.kiwisecurity.com/.

“Vehicle counting & detection systems”. (n.d.). Retrieved October 10, 2017, from https://www.swarco.com/stl/Products-Services/Parking-Solutions/Parking-guidance/Vehicle-counting-detection-systems. (11)