Lab 2 - ParkODU Prototype Product Specification

Gerard S. Silverio

CS 411

Professor Thomas J. Kennedy

26 Feb 2018

Version 1

Table of Contents

List of Figures

List of Tables

1 Introduction

Like many colleges and universities in the United States, Old Dominion University students, faculty, staff, and visitors face a challenging dilemma regarding the inefficiency of finding available parking spaces on campus. In 2017, Old Dominion University boasts a combined total of 24,375 undergraduate and graduate enrollments and an 18:1 student-to-faculty ratio. (3) With approximately 18,769 of students living off-campus, 11,692 students enrolled in online courses, 1,850 of on-campus students with vehicles, and 1511 faculty members (835 Full-time and 676 Part-time), approximately 10,438 commuters, and resident students are forced to share the 7,500 parking spaces available at Old Dominion University. (12, 4, 3, 13, 14)

ParkODU is a next generation parking solution platform designed to reduce the time and effort of identifying available parking spaces while increasing the efficiency of filling vacancies in parking lots, garages, and meters. By leveraging open-source technologies and with the ability to interface with new or existing vehicle detection systems, configuration and deployment is simple and cost-effective. In addition, as the popularity and the technology of distance learning advances, enrollment in online courses at ODUOnline will mitigate the need for additional parking spaces thus further reducing traffic flow within parking lots and garages.

1.1 Purpose

ParkODU is a web-based software solution designed to automate the search for an ideal parking space and to reduce the amount of time drivers normally take to identify vacant spaces through manual search. The platform uses continuous monitoring to detect parking space availability in real-time while collecting trend data to be used to generate analytical reports and forecasting. ParkODU boasts a suite of innovative modules with goals to simplify the process of identifying available parking locations, parking management, user management, and events

management. While ParkODU offers modules for parking management, it does not offer a built-in vehicle detection system. As ParkODU's only constraint, without an existing vehicle detection system to integrate with, its capabilities are very limited.

1.2 Scope

ParkODU was designed and developed to be implemented and integrated onto any system with an existing vehicle detection system. Although its initial purpose is to serve as a parking management solution for Old Dominion University, it is highly capable of being integrated onto any external system with a vehicle detection system. As an open source project, it is freely available to the public allowing new and existing vehicle detection systems to build tools to integrate with ParkODU.

The ParkODU prototype's main objective is to demonstrate the capabilities of its major functional components which consist of the Web Application, Data Persistence, Application Programming Interface (API), and the Mock Vehicle Detection System. Utilizing Hazelcast's distributed computing platform, clusters of ParkODU can be easily configured and deployed on any operating system. Due to the constraint of ParkODU requiring a vehicle detection system, the prototype provides a simulated approach to generating realistic parking space availability data.

(This section intentionally left blank)

1.3 Definitions, Acronyms, and Abbreviations

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

Agile - a software development 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 can park without receiving a violation

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

Real-time - actual 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

1.4 References

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

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

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

Dear Future ODU Students. Retrieved on November 02, 2017 from https://www.theodysseyonline.com/dear-future-odu-students. (1)

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

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

Inductive Loops. Retrieved on January 24, 2018, from http://diamondtraffic.com/technicaldescription/124 (17)

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

Is University Parking a Common Grievance? Parking Today Media. September 2011. Retrieved on September 2017 from http://www.parkingtoday.com/articledetails.php?id=1072. (8)

Silverio, Gerard. “Lab 1 - ParkODU Product Description.” February 2018. Portable Document Format (PDF).

ODU Campus Parking Map. Retrieved on 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.)

ODUOnline at a Glance. ODU Online. Retrieved on January 24, 2018 from https://online.odu.edu/about/at-a-glance (14)

Old Dominion University | Student Life. U.S. News Best Colleges. Retrieved on January 24, 2018, from https://www.usnews.com/best-colleges/old-dominion-3728/student-life (12)

Old Dominion University | Campus-Info. U.S. New Best Colleges. Retrieved on January 24, 2018, from https://www.usnews.com/best-colleges/old-dominion-3728/campus-info (13)

Operating Budget and Plan. Old Dominion University. Retrieved on November 02, 2017 from https://www.odu.edu/content/dam/odu/offices/budget-office/docs/opplan2017.pdf. (65)

Parking and Traffic Procedures. Old Dominion University. Retrieved on November 02, 2017 from https://www.odu.edu/content/dam/odu/offices/parking-and-transportation-services/docs/parking-transportation-rules-and-regulations.pdf. (64)

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

Robust Vehicle Detection under Various Environments to Realize Road Traffic Flow Surveillance Using an Infrared Thermal Camera. Retrieved on January 24, 2018 from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4339787/ (16)

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

Spring: the source of modern java by Pivotal. Retrieved on January 23, 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)

Ultrasonic Sensors vs Infrared (IR) Sensors. Retrieved January 24, 2018, from https://www.maxbotix.com/articles/ultrasonic-or-infrared-sensors.htm (15)

University Facts & Figures. Old Dominion University. Retrieved on November 2, 2017, from https://www.odu.edu/about/facts-and-figures. (3)

Vehicle Counter. Retrieved on October 10, 2017 from https://www.kiwisecurity.com/.

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

What Is MongoDB? Retrieved on January 23, 2018 from https://www.mongodb.com/what-is-mongodb.

Why Hazelcast IMDG? Retrieved on January 24, 2018, from https://hazelcast.com/why-hazelcast/imdg/ (18)

(This section intentionally left blank)

1.5 Overview

This product specification provides a high-level overview of the ParkODU system architecture, major functional components, functional capabilities, software configurations, and communication protocols. The prototype architectural description portion of the product specification provides a detailed description of the overall architecture of the prototype along with its major functional components and their underlying functional capabilities.

2 General Description

ParkODU offers a web application and API that demonstrates the capability of increasing the efficiency of fulfilling parking space vacancies while reducing the time it takes for drivers to identify ideal parking locations.

2.1 Prototype Architecture Description

The major functional components of the ParkODU prototype include the following:

- **Web Application:** This component offers a web-based user interface providing users with the capability to interact with ParkODU.
- **Data Persistence:** This component provides the capability of storing ParkODU objects in cache utilizing Hazelcast's In-Memory Data Grid (IMDG). MongoDB is utilized as a redundant data store in the event of a failover.
- **Application Programming Interface (API):** This component provides the capability for external users and applications to send and receive data to or from ParkODU.
- **Mock Vehicle Detection System:** This component provides the capability of simulating a vehicle detection system.

As shown on Figure 2, the prototype will consume simulated data from an external service mocking a vehicle detection system. Once consumed, the data will be evaluated by the ParkODU analytics engine and be readily available for consumers to view through the web application and APIs.

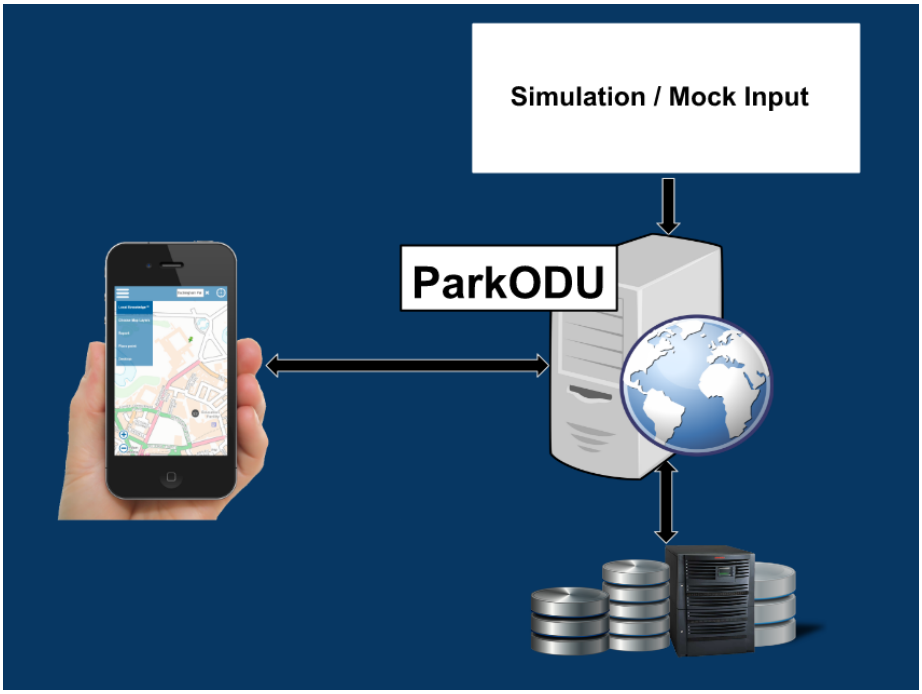

Figure 1- Prototype Major Functional Components Diagram

2.1.1 Web Application Component

The Web Application Component utilizes the Spring Framework's Model View Controller (MVC) design pattern. With a web front-end constructed using Thymeleaf's server-side template engine, pages are received by the end user as an html document thus allowing for faster page rendering. This component encapsulates most of the features of ParkODU which include Historical Trends, Report Generator, Availability Forecasting, Continuous Monitoring,

Dynamic Floor Plans, Events Management, Advanced Querying, and the Administrative Console.

2.1.2 Data Persistence Component

ParkODU's Data Persistence Component takes advantage of the Hazelcast In-Memory Data Grid (IMDG) that allows applications to store objects within heap in order allowing faster data persistence and faster query execution. A redundant data store is persisted on MongoDB acting as a failover for the system.

By utilizing Hazelcast IMDG, ParkODU significantly decreases the time it takes from receiving a web request to sending a response payload back to the user. By storing objects within the same JVM as the ParkODU itself, this allows for faster data reads and writes in comparison to a traditional data store that reads and writes to disk. In addition, Hazelcast Map Indexing gives the added performance boost of indexing key object attributes used by queries.

Due to the extreme volatility of storing objects within the java heap, a failover plan to utilize MongoDB as a secondary data store is put in place by ParkODU. In the event that all of ParkODU's Hazelcast nodes were to unexpectedly shut down, data stored within the java heap would be deallocated and lost. In order to mitigate this potential data loss, objects are redundantly stored into Mongo collections as they are persisted within Hazelcast maps. Thus, after the Hazelcast servers are rebooted, the Hazelcast maps are pre-populated with the redundant data stored within Mongo.

2.1.3 Application Programming Interface (API) Component

The Application Programming Interface (API) Component provides an interface for external applications to communicate with ParkODU. Interactions between ParkODU are

possible by defining RESTful web services and object specifications. External applications such as native Android and iOS applications, external signage, vehicle detection systems, and external web applications can be built to send and receive data to and from ParkODU.

2.2 Prototype Functional Description

The prototype of ParkODU will contain all the functional capabilities from the real-world product except for the ParkODU Android and iOS Mobile Applications. In addition, the generation of dynamic floor plans with real-time updates of parking space availability will be modeled. The vehicle detection system integration and external signage will be simulated by an external web application. The full functional capability list comparing the real-world product and the prototype is shown on Table 1.

Functional Capabilities	Real-World Product	Prototype
Historical Trends	Yes	Yes
Report Generator	Yes	Yes
Availability Forecasting	Yes	Yes
Continuous Monitoring	Yes	Yes
Dynamic Floor Plans	Yes	Modeled
Events Management	Yes	Yes
Advanced Querying	Yes	Yes
Administrative Console	Yes	Yes
Android Mobile Application	Yes	No
iOS Mobile Application	Yes	No
Vehicle Detection System Integration	Yes	Simulated
External Signage	Yes	Simulated

Table 1 - Real World Product and Prototype Capabilities

The ParkODU main components are comprised of the following functional capabilities:

- **Historical Trends:** Provides users the capability to generate historical reports depicting past garage usages and trends.

- **Report Generator:** Provides the capability for ParkODU to generate reports such as garage usage reports, trend analysis, and future projections.
- **Availability Forecasting:** Provides users with a line graph for each parking structure depicting the predicted total garage availability for the next 24 hours.
- **Continuous Monitoring:** Provides the mechanism that allows ParkODU administrative users to configure a scheduled task to collect parking space statuses and availabilities within a configurable time interval.
- **Dynamic Floor Plans:** Provides administrative users the capability to draw floor plans using a web interface portraying the floor plan for an actual parking garage.
- **Events Management:** Allows administrative users to import and schedule events to occur such as temporarily changing garage statuses and parking space permit types.
- **Advanced Querying:** Provides users with the capability to search for available parking spaces utilizing a combination of query filters such as the user's beginning address, permit types, and destination building.
- **Administrative Console:** Provides administrative users with the capability of adding, updating, and deleting garages, floors, parking spaces, permit types, users, and events.
- **Android Mobile Application:** Provides a native Android application for users to interface with ParkODU.
- **iOS Mobile Application:** Provides a native iOS application for users to interface with ParkODU.
- **Vehicle Detection System Integration:** A software-based adapter that allows interconnectivity between ParkODU and an external Vehicle Detection System.

- **External Signage:** A physical sign with the capability of querying ParkODU garage counts and statuses in real-time and presenting the results on an external display.

2.2.1 Software Interfaces

Mock Vehicle Detection System Interface: This simulated vehicle detection system interface establishes a connection between an external software application that generates realistic parking space availability data and ParkODU. The data transfer is conducted through Representational State Transfer (REST) utilizing HTTP methods and mapping CRUD (create, read, update, and delete) operations to HTTP requests.

2.2.2 User Interfaces

Web Application Interface: ParkODU's web application interface allows its users and administrations to conduct CRUD (create, read, update, and delete) operations for objects within ParkODU utilizing an HTML based web front-end.

2.2.3 Communications Protocols and Interfaces

Hazelcast TCP/IP Cluster Configuration: The ParkODU Hazelcast nodes are configured to be a full TCP/IP cluster. A list of member IP addresses are declared by stating an IP address range or individually IP addresses delimited by a comma. These configurations are read and established on runtime. During startup, a Hazelcast node checks to see if any other nodes declared within the member list are running. If none of the nodes within the member list are running, the new node declares itself as the 'master node' and establishes a new Hazelcast cluster. Otherwise, if a node within the member list is already running, a join request will be sent to that node and if accepted, the new

node will join the existing cluster. Once connected, the Hazelcast nodes communicate directly through TCP/IP.

Hypertext Transfer Protocol (HTTP): Communication between users and external applications to ParkODU are established using Hypertext Transfer Protocol (HTTP).

(This section intentionally left blank)