Software Requirements Specification

For

Chariot

Submitted by

Chariot Dev

|  |  |
| --- | --- |
| **Instructor:** | Dr. Gregory Hislop |
| **Team Members:** | Medard Azandegbe, Kamalludin Colaire, Juan Garcia, Ryan Hassing, Christopher Mak, Enioluwa Segun |
| **Cycle:** | 2 |
| **Date Submitted:** | 1/26/2020 |

Grading Rubric - Requirements Specification

This rubric outlines the grading criteria for this document. Note that the criteria represent a plan for grading. Change is possible, especially given the dynamic nature of this course. Any change will be applied consistently for the entire class.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Achievement** | **Minimal** | **Exemplary** | **Pts** | **Score** |
| **Content** | Section(s) missing, not useful, inconsistent, or wrong. | Provides all relevant information correctly and with appropriate detail |  |  |
| Introduction |  |  | 10 |  |
| Overall Description |  |  | 10 |  |
| Functions and Constraints |  |  | 40 |  |
| Logical Database |  |  | 20 |  |
| **Grammar and Spelling** | Many serious mistakes in grammar or spelling | Grammar, punctuation, and spelling all correct | 10 |  |
| **Expression** | Hard to follow or poor word choices | Clear and concise. A pleasure to read | 5 |  |
| **Tone** | Tone not appropriate for technical writing | Tone is consistently professional |  |  |
| **Organization** | Information difficult to locate | All information is easy to find, and important points stand out | 5 |  |
| **Layout** | Layout is inconsistent, visually distracting, or hinders use | Layout is attractive, consistent, and helps guide the reader |  |  |
| **Late Submission** |  |  |  |  |
| **Total** |  |  | 100 |  |

# Contents

[Contents 2](#_Toc29818190)

[1 Introduction 4](#_Toc29818191)

[1.1 Purpose 4](#_Toc29818192)

[1.2 Definitions 4](#_Toc29818193)

[2 Overall Description 5](#_Toc29818194)

[2.1 Product Functions 5](#_Toc29818195)

[2.2 User Characteristics 5](#_Toc29818196)

[3 Specific Requirements 7](#_Toc29818199)

[3.1 Functional Requirements 7](#_Toc29818200)

[FR1 – Installation 7](#_Toc29818201)

[FR2 – Running Chariot 7](#_Toc29818202)

[FR3 – Authentication 8](#_Toc29818203)

[Network Management 8](#_Toc29818204)

[FR4 – Networks 8](#_Toc29818205)

[FR5 – Manage Networks 8](#_Toc29818206)

[FR6 – Network Initialization 8](#_Toc29818207)

[FR7 – Network Deletion 9](#_Toc29818208)

[FR8 – Manage Network Permissions 9](#_Toc29818209)

[IoT Device Management 9](#_Toc29818210)

[FR9 – IoT Device Configuration 9](#_Toc29818211)

[FR10 – Adding IoT Devices 10](#_Toc29818212)

[FR11 – IoT Device Addition During a Data Collection Episode 10](#_Toc29818213)

[FR12 – IoT Device Removal 10](#_Toc29818214)

[FR13 – Disconnecting an IoT Device During a Data Collection Episode 11](#_Toc29818215)

[FR14 – IoT Device Removal During a Data Collection Episode 11](#_Toc29818216)

[FR15 – IoT Device Status 11](#_Toc29818217)

[Data Collection 11](#_Toc29818218)

[FR16 – Data Collection Episode 11](#_Toc29818219)

[FR17 – IoT Device Data Collection 13](#_Toc29818220)

[FR18 – Data Collection Episode Configuration 13](#_Toc29818221)

[FR19 – Concurrent Network Data Collection 14](#_Toc29818222)

[FR20 – Concurrent Network Data Collection of Network Sharing IoT Devices 14](#_Toc29818223)

[Data Access 14](#_Toc29818224)

[FR21 – Data Storage 14](#_Toc29818225)

[FR22 – Live Data Access 15](#_Toc29818226)

[FR23 – Timestamped Received Data 15](#_Toc29818227)

[FR24 – Timestamped Collected Data 15](#_Toc29818228)

[FR25 – Load Collected Data from a File 15](#_Toc29818229)

[Data Output 16](#_Toc29818230)

[FR26 – Data Output 16](#_Toc29818231)

[FR27 – Data Output during a Data Collection Episode 16](#_Toc29818232)

[FR28 – Visualizer 16](#_Toc29818233)

[Module Components 17](#_Toc29818234)

[FR29 – Modular Components 17](#_Toc29818235)

[FR30 – IoT Device Module 18](#_Toc29818236)

[FR31 ~~–~~ Storage Unit Module 18](#_Toc29818237)

[FR32 – Data Output Module 18](#_Toc29818238)

[FR33 – Module Loader 19](#_Toc29818239)

[3.2 Nonfunctional Requirements 19](#_Toc29818240)

[NFR1 – Extensibility 19](#_Toc29818241)

[NFR2 – Deployment 19](#_Toc29818242)

[NFR3 – Data Integrity 19](#_Toc29818243)

[NFR4 – Real-Time Data Collection 20](#_Toc29818244)

[NFR5 – Error Handling 20](#_Toc29818245)

[NFR6 – Security 20](#_Toc29818246)

[NFR7 – Documentation 20](#_Toc29818247)

[3.3 Data Requirements 21](#_Toc29818248)

[3.4 Design Constraints 22](#_Toc29818249)

[DC1 – Graphic User Interface 22](#_Toc29818250)

[DC2 – External Library Usage 22](#_Toc29818251)

[DC3 – LGPL Licensing 22](#_Toc29818252)

[DC4 – Networking Standard Compliance 22](#_Toc29818253)

[DC6 – Adaptation of Drexel Wireless’ IoT Sensor Framework 22](#_Toc29818254)

[4 References 23](#_Toc29818255)

# Introduction

## Purpose

This document shall provide a description and the requirement specification for Chariot, an IoT sensor framework for the development and operation of heterogeneous sensor networks. Chariot is built upon the foundations provided by Dr. Mongan’s IoT Sensor Framework, but Chariot will expand and adapt from it and other IoT Sensor Frameworks. This document shall outline the requirements that will be used by stakeholders and the product development team for development and review of the final product.

## Definitions

This section describes technical terms commonly used in the Software Requirements Specification.

1. **Internet of Things (IoT)** - the interconnection via the Internet of computing devices embedded in everyday objects, enabling them to send and receive data.
2. **Core Component** – components of Chariot that provide core functionality for Chariot to perform its tasks.
3. **IoT device** – a hardware device that records data about its environment and has the capability to connect to a network and communicate with other devices.
4. **Data Collection Episode** – the timeframe during which Chariot is actively collecting and storing data received from IoT devices
5. **Received Data** – data transmitted, or in transmission from an IoT device that has not yet been stored
6. **Collected Data** – data received from an IoT device that has been saved to a storage unit
7. **Storage Unit** – where collected data is saved, be it a database, csv file, or other format or data structure
8. **Network** - a collection of configured, connected IoT devices
9. **Data Output Receiver** – an external program or service that connects to Chariot and receives outputted data
10. **Module** – an addon to Chariot that adds functionality or compatibility tools to the system

# Overall Description

The Internet of Things (IoT) is a system of connected devices capable of collecting and transferring data. Different users of IoT devices may be interested in different types of data. A gardener might want to track the pH value of their soil and the air humidity. A city planner may look for a way to track pedestrian traffic in common spaces. A factory manager might look for a way to track the status of machinery and production rates.

IoT infrastructure has exploded in the past decade, so there are devices on the market to collect data for a wide variety of applications. But the wide variety of devices require special considerations. A Wi-Fi 4K camera connects to a network much differently than a home thermometer. A motion detector may collect and transfer data in a different form than a pressure sensor.

So, despite progress in the IoT industry, it is still difficult to integrate different types of devices into a unified system. Chariot solves this problem. Chariot is a platform for IoT device management, data storage, and analysis. Users can add, remove, and configure a variety of devices on a single network. With Chariot, a gardener could purchase off the shelf pH and humidity IoT devices and gather data from all from one convenient interface

## Product Functions

Chariot will provide the following functions:

* Easy addition and removal of IoT devices on a network
* Configuration of devices
* Simultaneous usage of heterogeneous devices on a single network
* Real-time data collection
* Device status reporting
* Flexible data export, including a web API
* Data analysis tools for data visualization and processing
* Modular expansion

## User Characteristics

This section identifies and describes the user types Chariot is designed for.

### Technical – A user with technical familiarity with the system and its underlying code. They will have the knowledge to add or improve data models and IoT drivers for use by Chariot.

### Non-technical – This user does not need to interact with the system on a deep technical level. They use existing modules rather than creating their own with a focus on the final collected data.

# Specific Requirements

The below table describes the priorities of each requirement.

|  |  |
| --- | --- |
| **Priority** | **Description** |
| **Essential** | A requirement that is necessary for the final product to perform to stakeholder expectations. |
| **High** | A requirement that is not required but will significantly improve the quality of the final product. Will be added after Essential requirements, time permitting. |
| **Low** | A requirement that offers minor improvements to the final product. Will be added after High requirements, time permitting. |

## Functional Requirements

### FR1 – Installation

**Priority: Essential**

**Target User: All**

Chariot shall have a single program install all components of the system to a computer.

### FR2 – Running Chariot

**Priority: Essential**

**Target User: All**

Upon system startup, Chariot shall prompt for the user to login, **See FR3**, and then display a welcome screen.

### FR3 – Authentication

**Priority: High**

**Target User: All**

Chariot shall have an authentication system for the management of user access.

### Network Management

The following section describes the core component of Chariot’s IoT Network Management, and its administration of collection of IoT devices.

### FR4 – Networks

**Priority: Essential**

**Target User: All**

Chariot shall group IoT devices into heterogeneous collections called networks. Each network will have configuration information for its devices and data collection.

### FR5 – Manage Networks

**Priority: Essential**

**Target User: All**

Users shall be able to manage connected networks, allowing them to view devices in a network, and edit the configurations of said devices and the collection parameters of the network.

### FR6 – Network Initialization

**Priority: Essential**

**Target User: All**

A user shall be able to create new networks of IoT devices, adding it to the list of known networks. When adding devices to the network, a user will have to either manually configure or load a pre-existing configuration file.

### FR7 – Network Deletion

**Priority: Essential**

**Target User: All**

A user shall be able to delete a network from the list of available networks, removing its device configurations and data collection parameters from the system.

### FR8 – Manage Network Permissions

**Priority: High**

**Target User: Technical**

A technical user shall be able to control other users’ access to their owned networks. They can allow or deny other users from editing or using their networks.

### IoT Device Management

The following section describes the core component of IoT Device Management, and how Chariot will interact with external IoT Devices.

### FR9 – IoT Device Configuration

**Priority: Essential**

**Target User: All**

Chariot shall use user defined configurations for the interaction with IoT devices. Each device configuration will include identifiers for the system and users as well as the setting of any device’s defined parameters.

#### FR9.1 – Save IoT Device Configuration

**Priority: Essential**

**Target User: All**

A user can save the configuration of an IoT device to an external file for use with other system IoT devices, or with another, external Chariot system.

#### FR9.2 – Load IoT Device Configuration

**Priority: Essential**

**Target User: All**

Users shall be able to load in previously saved IoT device configurations in place of the manual input of configuration parameters.

### FR10 – Adding IoT Devices

**Priority: Essential**

**Target User: All**

A user shall be able to add IoT devices to new or existing networks.

### FR11 – IoT Device Addition During a Data Collection Episode

**Priority: High**

**Target User: All**

A user shall be able to add IoT devices to a network that is currently in a Data Collection Episode, **See FR15**.

### FR12 – IoT Device Removal

**Priority: Essential**

**Target User: All**

A user shall be able to remove a connected IoT device from an associated network. Removal will prevent the further receiving and collection of data from that IoT device onto the target network.

### FR13 – Disconnecting an IoT Device During a Data Collection Episode

**Priority: High**

**Target User: All**

~~An IoT device can be removed from a Network, either purposefully or through a failure, without affecting the DCE.~~

In the event that an IoT Device is unintentionally disconnected from a network during its DCE Chariot will attempt to reconnect, before setting its status to disconnected and ignoring received data from the device.

### FR14 – IoT Device Removal During a Data Collection Episode

**Priority: Low**

**Target User: All**

A user can remove an IoT Device from a network during a DCE. Chariot will first ignore data received from this device and then allow the user to continue with the removal process.

### FR15 – IoT Device Status

**Priority: High**

**Target User: All**

Chariot shall record and make accessible the status of all IoT devices on a network every second. A device status will include the device ID, uptime, resource utilization, and other information pertinent to the model of device.

### Data Collection

The following section describes the core component of Data Collection and the Data Collection Episode.

### FR16 – Data Collection Episode

**Priority: Essential**

**Target User: All**

Chariot shall take received data from a network’s IoT devices and save it as collected data into a storage unit during a data collection episode.

#### FR16.1 – Initialize Data Collection Episode

**Priority: Essential**

**Target User: All**

Users will initialize a data collection episode to receive data from a network’s connected devices and save it to a data storage unit.

#### FR16.2 – Pause Data Collection Episode

**Priority: Essential**

**Target User: All**

A user shall be able to pause an ongoing data collection episode, suspending the receiving of data from a network’s IoT devices. The system shall also prevent the collection of additional data to a data storage unit.

#### FR16.3 – Resume Data Collection Episode

**Priority: Essential**

**Target User: All**

A user shall be able to resume a paused data collection episode. This will allow for the further collection of data to the storage unit and the receiving of data from the network’s IoT devices.

#### FR16.4 – Terminate Data Collection Episode

**Priority: Essential**

**Target User: All**

A user shall be able to terminate data collection episodes that have gone through their full duration or allow for the user to preemptively terminate a data collection episode. This will suspend the further receiving of data from the network’s IoT devices and terminate the collection of data to the storage unit.

### FR17 – IoT Device Data Collection

**Priority: Essential**

**Target User: All**

Chariot shall receive and collect data from at least one IoT device from a single network during a data collection episode.

#### FR17.1 – Concurrent IoT Device Data Collection

**Priority: Essential**

**Target User: All**

Chariot shall be able to receive and collect data from at least two IoT devices from a single network simultaneously during a data collection episode.

#### FR17.2 – Concurrent Heterogenous IoT Device Data Collection

**Priority: Essential**

**Target User: All**

Chariot shall be able to receive and collect data from at least two heterogeneous IoT devices from a single network simultaneously during a data collection episode.

### FR18 – Data Collection Episode Configuration

**Priority: Essential**

**Target User: All**

A user must create a DCE configuration to define what data is to be collected from the network’s IoT devices, the storage unit to use, formatting of the collected data, and duration of the data collection episode.

#### FR18.1 – Save Data Collection Episode Configuration

**Priority: High**

**Target User: All**

A user shall be able to save a DCE configuration to a file for use with other networks or on external chariot systems

#### FR18.2 – Load Configuration for Data Collection

**Priority: High**

**Target User: All**

A user shall be able to load previously saved data collection episode configurations in place of the manual input of configurations.

### FR19 – Concurrent Network Data Collection

**Priority: Low**

**Target User: All**

Chariot shall be able to receive and collect data from at least two separate networks simultaneously.

### FR20 – Concurrent Network Data Collection of Network Sharing IoT Devices

**Priority: Low**

**Target User: All**

Chariot shall allow for at least two separate networks that share at least one IoT device amongst their collections to perform concurrent data collection episodes if all shared devices have the same IoT device configuration, **See FR12**.

### Data Access

The following requirements indicate how Chariot manages user access to data.

### FR21 – Data Storage

**Priority: Essential**

**Target User: All**

Chariot shall have interfaces to external storage units for the saving and accessing of collected data.

### FR22 – Live Data Access

**Priority: High**

**Target User: All**

~~Users shall be able to access and use stored data during a data collection episode.~~

Chariot shall have interfaces for the access and manipulation of stored data during a data collection episode. **See FR25.**

### FR23 – Timestamped Received Data

**Priority: Essential**

**Target User: All**

Chariot shall timestamp all received data from an IoT device to the storage unit with the time it was received.

### FR24 – Timestamped Collected Data

**Priority: Essential**

**Target User: All**

Chariot shall timestamp all collected data to the storage unit with the time it was collected.

### FR25 – Load Collected Data from a File

**Priority: High**

**Target User: All**

A user can load in an existing collected data file for use within Chariot.

#### FR25.1 – Append Data

**Priority: High**

**Target User: All**

A user shall be able to append new collected data to an existing storage unit with collected data.

#### FR25.1 – Edit Data

**Priority: Low**

**Target User: All**

A user shall be able to edit collected data from a file, this will include changing the formatting, stored values, or deleting collected data.

### Data Output

The following requirements specify requirements for Chariot’s data output capabilities, and how data will be accessible by other external programs and systems. Examples of a data output tool include visualizers, data processing tools and algorithms.

### FR26 – Data Output

**Priority: Essential**

**Target User: All**

~~Chariot shall output all collected data to external programs services.~~

Chariot shall output collected data to external programs and services in parallel to the storage of said data.

### FR27 – Data Output during a Data Collection Episode

**Priority: Essential**

**Target User: All**

A user can connect and disconnect outputs to Chariot, and this will not affect the DCE or the connection of other outputs.

### FR28 – Visualizer

**Priority: Essential**

**Target User: All**

Chariot shall have a visual graphing tool that will display data transfers from IoT devices to Chariot as labelled line graphs during a DCE.

#### FR28.1 – Visualizer Replay

**Priority: High**

**Target User: All**

A user shall be able to simulate a live visualizer graphing by using existing collected data.

### Module Components

The following requirements will introduce and specify the addition and removal of new components to an existing Chariot system to increase in functionality.

### FR29 – Modular Components

**Priority: Essential**

**Target User: All**

Chariot shall utilize module components to allow for the expansion potential devices, programs and services it can interact with. Modules will also be able to create other functional tools provided by Chariot. A module will fulfill its intended purpose without interfering with the function of existing core components or other modules.

#### FR29.1 – Module Addition

**Priority: Essential**

**Target User: All**

A user shall be able to add modules to Chariot. This will involve the user providing source files to Chariot and then installing them to the system. When installed the user can begin using module with Chariot.

#### FR29.2 – Module Removal

**Priority: Essential**

**Target User: All**

A user shall be able to remove modules from Chariot. The user will uninstall the module from Chariot but will not delete or change any of the source files that may be saved on the system associated with the module. Networks, or DCEs that utilized the module will remove them.

### FR30 – IoT Device Module

**Priority: Essential**

**Target User: All**

IoT Device Modules shall define what information will be received from an IoT device and how it will be formatted as received data. The configurable variables for the IoT device can be changed by the user using this module.

### FR31 ~~–~~ Storage Unit Module

**Priority: Essential**

**Target User: All**

Storage Unit Modules shall format received data to a collected data format accepted by its associated storage unit. The module will also assist in the initialization and configuration of the storage unit and translate commands to the appropriate syntax of the storage unit.

### FR32 – Data Output Module

**Priority: Essential**

**Target User: All**

Data Output Modules shall allow for further formatting and filtering of the data sent to different data outputs. This new formatted data will be accessible as a new output source that can be connected to by an external program or service.

### FR33 – Module Loader

**Priority: Low**

**Target User: Non-Technical User**

A module loader will install or uninstall modules for a non-technical user in a user-friendly environment. The loader will check for compatibility issues and inform the users of these during installation and uninstallation of modules.

## Nonfunctional Requirements

The statements below define the non-functional requirements for the system.

### NFR1 – Extensibility

**Priority: Essential**

**Target User: Technical**

A technical user shall be able to increase Chariot’s functionality by creating and adding IoT devices, data storage units, and data output receivers to a Chariot system.

### NFR2 – Deployment

**Priority: Essential**

**Target User: Non-Technical**

Chariot’s deployment process shall be streamlined to the point that a Non-technical user described in Section **2.2.2** can install and begin data collection without external assistance.

### NFR3 – Data Integrity

**Priority: Essential**

**Target User: All**

Chariot shall maintain integrity of collected data in such a way that Chariot does not alter values during the collection process.

### NFR4 – Real-Time Data Collection

**Priority: High**

**Target User: All**

For every second of data sent to Chariot, Chariot shall fully capture the data sent in that one second interval to allow for real time analysis.

### NFR5 – Error Handling

**Priority: High**

**Target User: All**

Chariot must not crash due to the unexpected failure of an IoT Device or network. The system shall have error handling measures to ensure the continued, uninterrupted collection of data.

### NFR6 – Security

**Priority: High**

**Target User: Technician**

A technical user shall be able to encrypt the collected data and data streams used and generated by Chariot too their own encryption standards.

### NFR7 – Documentation

**Priority: High**

**Target User: All**

Chariot shall be packaged with extensive internal and external documentation. Internal documentation will clearly describe how Chariot’s code functions. External documentation will describe how to operate the system.

## Data Requirements

The following table describes the various kinds of data expected to be utilized by Chariot

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Size** | **Comment** |
| User information | Dictionary | Variable | Username/password information for each user of the system, **See FR3** |
| Access list | List | Variable | For each network, there is a list of users with access to network, **See FR8** |
| Networks | List | Variable | List of a connected networks. **See FR4** |
| Devices on network | List | Variable | List of devices connected to a network. **See FR5** |
| Device information | Dictionary | Variable | ID, name, and other information specific to a device. |
| Device Configuration | JSON | Variable | Configurable settings of a device and their values. **See FR9** |
| Device status | JSON | Variable | Uptime, resource utilization, and other status information pertinent to device. Different from Device information, as Device status will frequently change. **See FR14** |
| Device data stream | Stream | Variable | Data received from device sensors. |
| Network Configuration | JSON |  | Configurable settings for a network. |
| Data collection episode status | Integer |  | Indicates if data collection episode is in progress, paused, or terminated. Also indicates errors in data collection. |
| Collected data | Various | Variable | Incoming data from data analysis episode is written to a database. Previously collected data can also be loaded from a file. The type is dependent on the storage unit used by the data collection episode. |
| Output Source | Stream | Variable | An output stream that can be connected to by an external service or program to receive data from Chariot. |

## Design Constraints

The following section specifies the design constraints restricting design of the system.

### DC1 – Graphic User Interface

The user shall be able to use the system via a graphic user interface. Stakeholders have expressed frustration using the command line for tasks this system would accomplish, so a graphic user interface is preferred.

### DC2 – External Library Usage

Chariot will make use of external libraries to interface with IoT devices. The system will use these libraries as specified.

### DC3 – LGPL Licensing

Chariot shall be licensed with the GNU Lesser General Public License (LGPL) (Free Software Foundation, Inc., 2007).

### DC4 – Networking Standard Compliance

Chariot shall comply will need to be able to communicate with IoT devices using various networking standards such as, IEEE 802.3, 802.11 and continue to develop for newer networking technologies such as LoRa. (Institute of Electrical and Electronics Engineers, 2018).

### DC6 – Adaptation of Drexel Wireless’ IoT Sensor Framework

Chariot will use and adapt related work in the field of IoT data collection. The system will use several key features of Dr. Bill Mongan’s IoT Sensor Framework (Mongan, 2019). Chariot will adhere to the conventions of original work and will document where work is used or adapted.

# References

Free Software Foundation, Inc. (2007, June 29). *GNU Lesser General Public License*. Retrieved from GNU Operating System: https://www.gnu.org/licenses/lgpl-3.0.en.html

Institute of Electrical and Electronics Engineers. (2018, November 14). *IEEE Standards Activities in the Internet of Things.* Retrieved from IEEE Standards Association Web site: https://standards.ieee.org/content/dam/ieee-standards/standards/web/documents/other/iot.pdf

Mongan, W. (2019). *IoT Sensor Framework*. Retrieved from William Mongan portfolio: http://www.billmongan.com/portfolio/iotframework/