Requirements Specification

for

User-Configurable Ocean Data Dashboards

Version 1.0 approved

Prepared by:
Tyler Harnadek, V00818721
Joel Kerfoot, V00855314
Christina Rembold, V00924549
Solitei Tutui, V00810533

Group Number: 22

November 2nd, 2018

Table of Contents

Revision History

Name	**Date**	**Reason For Changes**	**Version**
Christina Rembold	31/10/18	Original Draft	1.0.0
Joel Kerfoot	02/11/18	TA Feedback	1.1.0
Tyler Harnadek	02/11/18	Final Draft	1.2.0

Table of Figures

1. Introduction

As part of the ECE399 – Design Project – at the University of Victoria, it is our task to improve the existing user interface of Ocean Networks Canada's (ONC) Oceans 2.0 website. This document specifies the requirements of the project "User-Configurable Ocean Data Dashboards".

1.1 Purpose

This document outlines the requirements specification for a user configurable Ocean Data Dashboard, as requested by ONC in the ECE 399 project proposals document [1]. This document mainly describes the optimal orientation of a user-friendly user interface (UI).

The Ocean Networks Canada (ONC) data archive presently houses 0.5 petabytes of data from hundreds of sensors. The data is archived and made freely available through a web-based interface and web services using the Oceans 2.0 data management system. The data can be accessed through a GUI or an application programming interface (API).

ONC data is freely available over the Internet for any non-commercial use and ONC encourages multi-disciplinary and inter-disciplinary work. For this reason, the tools and applications ONC develops are already employed by users across a diversity of disciplines and work settings.

This project focuses on building a set of easy-to-use dashboard components on top of the Oceans 2.0 API, which will facilitate the creation of customized web- and mobile-deployed dashboards for oceanographic data access. The dashboards will be built from a set of customizable widgets, which can be standard or user-defined [1].

1.2 Document Conventions

Due to limited communication with ONC regarding their requirements, there are some details missing for the complete creation of the requirements. To illustrate this, the acronym TBD is used if some important information is unavailable but has to be considered. As the project progresses, all instances of TBD are to be replaced with the relevant information.

1.3 Product Scope

Central to ONC's Smart Ocean™ Systems digital infrastructure offering is Oceans 2.0 – a unique and critical component developed to connect the subsea instruments systems, providing the capability for the 24/7 acquisition of extremely diverse and vast amounts of data, quality control and calibration, storage, visualization and access by a potentially global audience, as well as providing a convenient interface to handle otherwise complex tasks associated with the remote monitoring and control of the observatory infrastructure itself.

The core value of Oceans 2.0 lies not only in its ability to create greater and easier access to complex information, but also in the fact that the fundamental design was based on a principle of flexibility and scalability. This allows for not only disparate instrumentation to be accommodated, but also facilitates ease of customization to accommodate diverse end-user requirements, span core research, industrial monitoring, emergency preparedness, etc., in small- or large-scale observatory implementations.

In recognition of an emerging market sector for ocean analytics, ONC continues to enhance its Oceans 2.0 offering with state-of-the-art tools to improve, accelerate and expand operational marine decision-making capabilities [2].

1.4 References

[1] ECE Project Descriptions – revised. Victoria, BC, Canada: Ocean Networks Canada, 2018, p. 19. [Online]. Available: http://bit.ly/399pd18

[2] Oceans 2.0, Ocean Networks Canada, 2018. [Online]. Available: http://www.oceannetworks.ca/innovation-centre/smart-ocean-systems/ocean-observing-systems/oceans-20 [Accessed: 02 - Nov - 2018].

[3] Oceans 2.0 - API, Ocean Networks Canada, 2018. [Online]. Available: https://wiki.oceannetworks.ca/display/O2A/Oceans+2.0+API+Home [Accessed: 02 - Nov - 2018].

[4] *B. Biffard, M. Valenzuela, P. Conley, M. MacArthur, S. Tredger, E. Guillemot and B. Pirenne, "Oceans 2.0: Interactive tools for the Visualization of Multi-dimensional Ocean Sensor Data," 12 2016. [Online]. Available: http://adsabs.harvard.edu/abs/2016AGUFMIN12A..05B. [Accessed 20 10 2018].*

2. Overall Description

This section provides a more in-depth outline of the main problem of the existing user interface (UI) and why it is important to solve it. Furthermore, this section outlines the constraints, assumptions, and dependencies of the User-Configurable Ocean Data Dashboard.

2.1 Product Perspective

This product is an extension on an existing solution. Oceans 2.0 offers state-of-the art tools and capabilities to improve, accelerate and expand ocean research and marine decision-making capabilities.

The Oceans 2.0 Dashboards will make use of the following existing components. The main connection to these components will be via the Oceans 2.0 API.

- Oceans 2.0 API: The Oceans 2.0 API provides programmatic access to ONC data products and services which are used as input to the widgets.

- Oceans 2.0 Core Services: The Oceans 2.0 Core Services support all data acquisition, archival, and processing, in addition to performing user management and service hosting.

- Oceans 2.0 Storage: All data and code are hosted on servers at the University of Victoria and in a Compute Canada replicate data center to ensure data integrity. [1]

Users can browse through the data from ONC's underwater and land-based sensors, select the information they want, and then confirm their order, which is downloaded to their computer.

Developing robust software systems like Oceans 2.0 that can reliably collect, process, analyze, and archive huge volumes of data is probably the single biggest challenge facing the Internet of Things. Therefore, it is very important to develop a system that minimizes data loss.

However, in order to achieve the greatest possible success to make the website more popular and useful to users, it is important to make the operation of the website as easy as possible for the user. Therefore, the task is to improve the User Interface (UI) of Ocean Networks Canadas Data Preview to make it easier and more comfortable for users to search and download data. [3]

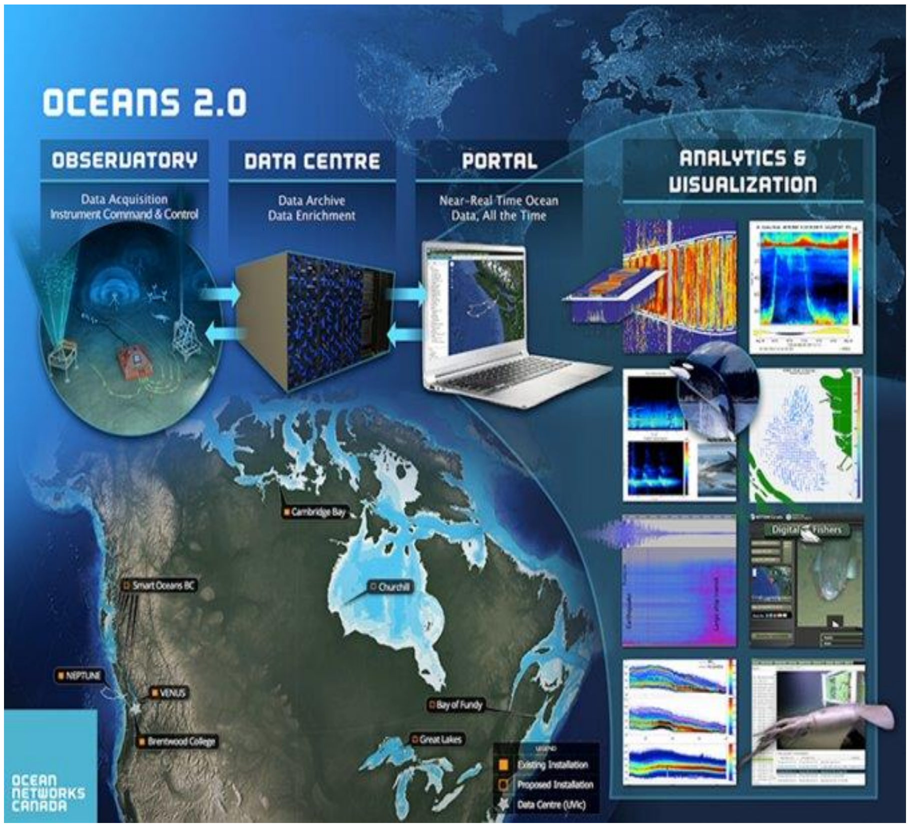


Figure 1: Overview Oceans 2.0

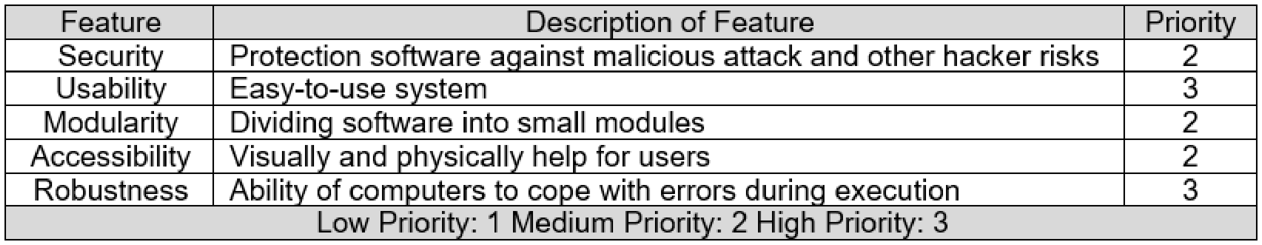
2.2 Product Functions

The major functions of this product are to provide a user customizable web dashboard that displays data from ONC 2.0 API in module widgets. The following list describes the primary functions of the web application:

- Customizable Dashboard
    - Create public or private templates
- Widget Library
    - Small modular UI elements that connect to OCEAN 2.0 API

2.3 Design Priorities

The following table shows how we have prioritized various parts of the design.



Feature	Description of Feature	Priority
Security	Protection software against malicious attack and other hacker risks	2
Usability	Easy-to-use system	3
Modularity	Dividing software into small modules	2
Accessibility	Visually and physically help for users	2
Robustness	Ability of computers to cope with errors during execution	3
Low Priority: 1 Medium Priority: 2 High Priority: 3		

Figure 2: Prioritization of Features

2.4 Operating Environment

The ONC Web Dashboard application will be hosted on ONC's existing servers, and their existing data access services will be used to provide all of the data that the application uses.

The application will be built for a modern web environment. This includes mobile and desktop support on all major browsers and operating systems. Specifically, support for Edge, Chrome, Firefox, Opera, and all major mobile browsers should be supported. The application will be operational on any browser with HTML5, CSS3, and JavaScript support.

2.5 Design and Implementation Constraints

The ONC dashboard application must be able to run smoothly on browsers, including less powerful computers. This includes alternate computing devices such as smartphones and tablets. Calculations must be optimized for this reason.

The application must also be modular, so it can be expanded as ONC adds additional underwater monitoring stations. For this reason, the application must either dynamically expand based on data changes or be easy to update with these changes.

2.6 Assumptions and Dependencies

A-1: The existing ONC servers will be able to service data requests in a reasonable period of time	
Rationale	In order to build a responsive web application, data needs to be responded to relatively quickly, usually less than 10 seconds. If responses take too long, users will become frustrated and probably close the application.

A-2: The data provided by the ONC database is free of errors	
Rationale	In order to provide reliable data through the Oceans Dashboards, the data sent to the application from the ONC database needs to be accurate so that the information acquired from the processed data is dependable.

D-1: ONC’s data access interface and API (Application Programming Interface) are robust and stable over time	
Rationale	The application needs to access the ONC data platform to get all of its data. For this reason, it’s critical that the API is available at all times, and that the interface is stable and doesn’t change.

D-2: The user’s browser has access to HTML5, CSS3, and 2014 or newer JavaScript	
Rationale	All modern web frameworks require the use of these resources, and they’re standard on most browsers. To save developer time, building without a web framework is not an option.

D-3: The Oceans Dashboard depends on the Oceans 2.0 API responding in JSON, or any other comma separated values	
Rationale	The Oceans Dashboards application expects the data sent from the Oceans 2.0 API to be in a comma separated value format. Any other format will mean that the data sent to the application is unable to be processed and presented as usable information.

3. External Interface Requirements

3.1 User Interfaces

Users will interact with the application through a web interface. The main user interface will be a dashboard where ONC data is presented to the user through customizable widgets, including charts, graphs, and other visualizations. These components will need both menus and settings to customize their properties. The dashboard itself should also be customizable, allowing users to choose which data they want to track. There are several discrete components that will require user interfaces.

3.1.1 Login and User Management

In order to save user settings and data, user profiles will be used in this application. To this end, the application should include a basic user interface to allow the user to create and delete profiles, change their access information, and add or delete dashboard configurations.

TBD 1: Management of user accounts is not yet defined. User account management may make use of existing ONC systems, or a standalone system may be developed for this application.

3.1.2 Dashboard

The goal of the dashboard is to provide users with an ‘at-a-glance' view of their chosen data sources and visualizations. For example, a scientist may choose to view data from a specific sensor type that suits their research. Users will be able to customize their dashboards to show only data that they are interested in.

Visualizations are presented inside of Widget components, which are detailed in section 3.1.3.

3.1.3 Widgets

Widgets are modular components that display a specific data point or group of data points. They can take the form of a statistic, graph, chart, or abstract visualization like a map or Venn diagram. Users will be able to customize the content of widgets and configure them so as to create new types of visualization.

3.1.4 Community Page

The community page is intended to provide a ‘discovery’ mechanism for new types of data and visualization. To this end, it should be easy for a user to view other visualizations and assess their usefulness.

Specifically, the user should be able to preview visualizations and dashboards that have been shared in the community page or share their own content.

3.1.5 Accessibility

Providing accessibility for all users is an important part of this project. The application will support internationalization in order to allow users who do not speak English to use it. Support for disability is a requirement.

The application should support the visually impaired by including high contrast colors, fonts, and visualizations, and offer text-based alternatives for graphical representations. To support screen readers, the application should take advantage of HTML5’s built in accessibility support.

3.2 Hardware Interfaces

Not applicable for this project.

3.3 Software Interfaces

The user configurable Dashboard will function using data retrieved from the Oceans 2.0 API. Oceans 2.0 is backed by a NoSQL database [4] and exposes an HTTP interface. The API responds to HTTP requests in a variety of formats; it supports JSON, CSV, and plain text formats.

ONC provides wrappers for the API in both MATLAB and Python programming languages, in addition to supporting HTTP requests. The Dashboard application will make use of one of these three interfaces.

Section 2.5 covers the software interfaces that are expected on the client browser; in short, support of 2014+ JavaScript, CSS3, and HTML5 browser interfaces is expected. These are provided in all of Edge, Chrome, Firefox, and Opera.

3.4 Communications Interfaces

The Oceans 2.0 API provides a collection of web services for programmatically unlocking access to the vast ONC data archive, whether it is scalar data, complex data or video and imagery. Use Discovery Services to find the data and Delivery Services to retrieve data. [3]

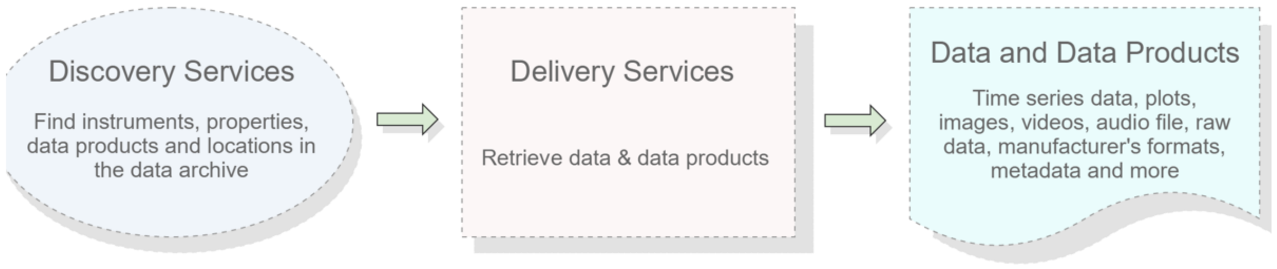

Discovery Services
Find instruments, properties, data products and locations in the data archive
Delivery Services
Retrieve data & data products
Data and Data Products
Time series data, plots, images, videos, audio file, raw data, manufacturer's formats, metadata and more

Figure 3: Overview of Ocean 2.0 Web Services

Communication with the Oceans 2.0 occurs over the HyperText Transfer Protocol (HTTP). (Appendix B: Figure 1) Because of the large amount of data to be transferred, data transfer rates are entirely dependent upon the speed of the Oceans 2.0 API. Optimizing this transfer rate is outside the scope of this project.

As the data is not sensitive, no specific encryption is required. The application should default to HTTP Secure, but if the user does not have an HTTP Secure compliant connection, HTTP can be used.

4. System Features

The system has two major features; a widget library and corresponding dashboard are the main requirements of this project. The application should provide a user configurable dashboard, composed of a variety of informative widgets. The user will be able to customize which widgets are displayed, and the content of individual widgets as they see fit. Additionally, users should be able to save their dashboard configurations, and share them with other users.

4.1 Widget Library

This section describes the functionality of the widget library. This feature is not existing; the following describes the correct behavior when implemented.

4.1.1 Description and Priority

This library must contain a collection of small and modular user interface elements that connect to ONC API to display scalar data, complex data, audio, imagery, and video. The individual widgets must be reusable and easy to implement should ONC choose to provide new or different data to users. The user should be able to customize the properties of individual widgets. They must be designed to work on mobile (phone/tablet) and desktop systems and support changing the displayed language either automatically or manually. This feature is critical to the web application and has the highest priority.

4.1.2 Stimulus/Response Sequences

The following list describes the various stimuli and response sequences:

- If the user changes the data to be displayed, the system will fetch the new data, do processing as necessary then update the widget to display the new data. The system will also display the current status of the process while the widget is being updated.

- If the system gets an instruction to change the display language the system will reload the widget library with the new language.

4.1.3 Functional Requirements

REQ 1: The widget must be customizable	
Details	The widget must have a menu of settings and respond to changes The widget should dynamically respond to user changes
Error Behaviour	If the user inputs a request that isn’t valid it will display an appropriate message to notify the user

REQ 2: Widgets must support different types of data	
Details	Widgets must be able to support data of different units and formats The widget should scale graphs and charts automatically, to suit the received data
Error Behaviour	The widget must fail gracefully if the data can’t be displayed, and provide insight to the user about the failure

REQ 3: Widgets must support user specified level of precision	
Details	Widgets should be able to set the level of precision of the data being acquired and processed The information presented should be based on the user specified precision level
Error Behaviour	The widget should display an error message when the level of precision selected is unavailable

4.2 Customizable Dashboard

This section describes the functionality of the customizable dashboard. This feature is not existing; the following describes the correct behavior when implemented.

4.1.1 Description and Priority

The customizable dashboard must be a system that is capable of displaying selected widgets from the larger library. The layout of the dashboard must be customizable to the user's needs. These dashboard layouts can then be saved as templates to be used again later. These templates can be kept private to the user or shared to the community. Users can then change the layout of their dashboard by loading one of their personal templates of selecting one from the community.

4.1.2 Stimulus/Response Sequences

The following list describes the various stimuli and response sequences:

- The user selects a template to use from the dashboard settings, the template will be loaded from the server and the layout of the widgets will be adjusted accordingly. If any new widgets are loaded its data must be fetched.
- The user chooses to save a dashboard template with private access control, the current layout will be saved, and the use provides a unique name and description. The template will then be saved to the server.
- The user chooses to save a dashboard template with public access control, the current layout will be saved, and the use provides a unique name, description, and keywords that others can search to find the template. The template will then be saved to the server and can be accessed by the community.
- The user chooses to add a new widget(s) to the dashboard. They will then be presented with a menu from which they select all the data products they want to add. The new widgets will then be added to the end of the dashboard and the widgets will fetch the needed data.
- The user chooses to change the current layout, they will then be able to manipulate the positions of the widgets against grid of snap points, similar to rearranging apps on a smart phone.

4.1.3 Functional Requirements

REQ 4: The dashboard must be customizable	
Details	The user should be able to define what widgets they want to see This includes adding, removing, and changing widgets The dashboard configuration should be saved to the user's profile
Error Behaviour	If a user enters a configuration that cannot be displayed, the application should inform the user and fail gracefully.

REQ 5:	The dashboard must be reusable
Details	The user should be able to save a layout as a template to be used later. When the template is loaded the layout is identical to when it was last used
Error Behaviour	If the template fails to load there will be no change to the user's current layout and the user will be informed of failure If the template fails to save the user will be informed of the fail and potential cause and be prompted to try again if possible

REQ 6:	The dashboard must be mobile compatible
Details	Web design should be responsive on screens of all sizes Major web browsers such as iOS Safari, Android Chrome, and mobile Firefox must be supported
Error Behaviour	If the web application is accessed in an incompatible browser, it should inform the user of browsers with which it is compatible

REQ 7:	Private dashboards must not be accessible by the public
Details	For the security of individual users any dashboard template that has been saved without being made publicly available will not be available for the public to view or otherwise use.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Application loading time and data access time are performance requirements in this application. Application loading time should be very quick – the application should be kept small, under 1MB before any images or media is included, and the application should load quickly once downloaded. It's reasonable to expect that the application could load in less than 1 second on a modern desktop computer.

Data access times are mostly limited by the ONC servers. In other words, how long it takes the ONC database to receive a query, access data, and respond with the requested data. Because ONC has a huge amount of data, this may be very slow. However, we expect that once the data is received by our application, we should process it in less than 5 seconds.

5.2 Safety Requirements

Safety requirements in this project include protecting user anonymity and security. The NIST (National Institute of Standards and Technology) Cybersecurity Framework provides guidelines that should be followed to protect security and integrity of users.

5.3 Quality Attributes

Ease of use is a critical quality attribute on this project. The current implementation is neither easy to learn, or easy to use once you've learned it. Our goal is to improve both learning and ease of use, with the higher priority placed on ease of use.

Correctness of data is vital for this project. Research may be based upon ONC data and if it is displayed incorrectly by the dashboard, there may be legal or ethical recourse. We define correctness as accurately displaying data as it is retrieved from the ONC backend and making a reasonable effort to confirm that the data has not been damaged in transit. Verifying the correctness of the data provided is not part of the scope of this project.

5.4 Business Rules

ONC Data is freely available for non-commercial use. The web dashboard application should maintain this openly available data and continue to work towards the goal of easy access to ocean data for all researchers. To this end, the web application should be available to all users.

The regulations that ONC use for their data will extend to the terms and conditions of this application; commercial use of the application will be prohibited, but use by researchers, educators, students, and the public is encouraged.

6. Other Requirements

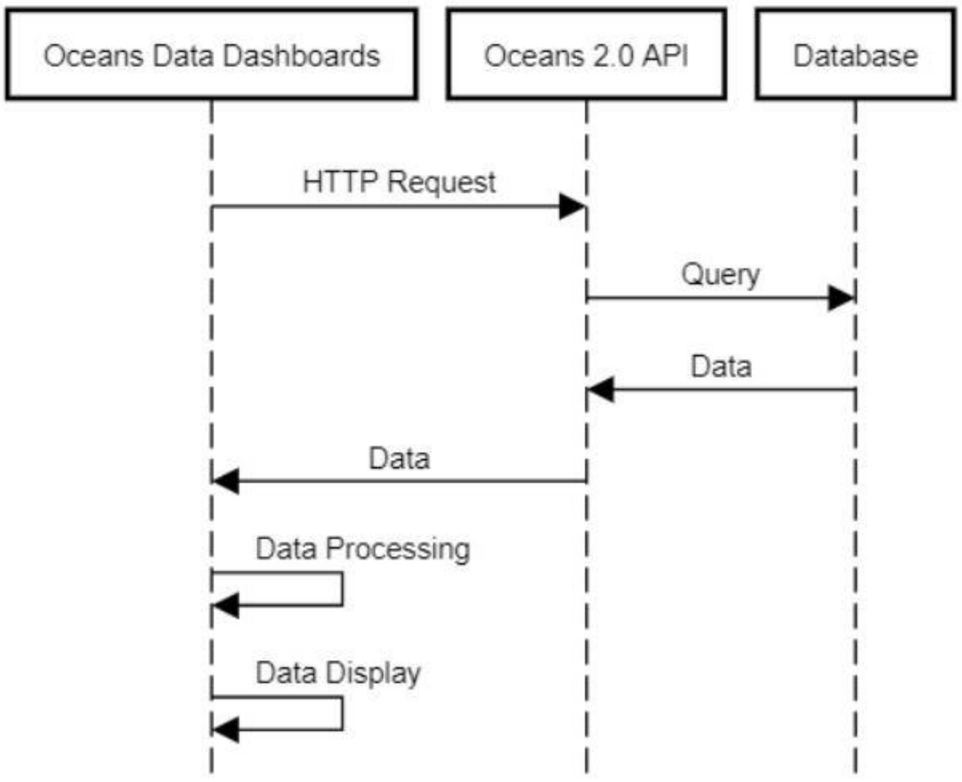
Extensibility is an important requirement for this project. As ONC adds additional monitoring stations and collects different types of data, they'll need different types of data visualization to keep track of them. Because of this, the application will be modular and well tested, in order to allow future updates to be as smooth as possible.

Appendix A: Glossary

API	Application Programming Interface, a type of interface that allows two pieces of unlike software to communicate
HTTP	Hypertext Transfer Protocol, a communication protocol over the world wide web
ONC	Ocean Networks Canada
TBD	To be decided
UI	User Interface

Appendix B: Analysis Models

HTTP Request Flow Diagram



Appendix C: To Be Determined List

TBD 1: Management of user accounts is not yet defined. User account management may make use of existing ONC systems, or a standalone system may be developed for this application.