Vinaya Bhattarai

Developing weather forecast delivery system to vessels operating in the Arctic region

Metropolia University of Applied Sciences

Bachelor of Engineering

Information Technology

Bachelor’s Thesis

20 May 2019

Author Title Number of Pages Date	Vinaya Bhattarai Developing weather forecast delivery system to vessels operating in the Arctic region 31 pages + 1 Appendix 20 May 2019
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Professional Major	Software Engineering
Instructors	Janne Salonen, Head of Department ICT

Despite the increase in marine traffic in the Polar region, high-quality forecasting has been lackluster in comparison to other areas of the planet. In an effort to improve this situation, the Polar Prediction Project started the multi-year long project named Year of Polar Project (YOPP). FINYOPP is a research project managed by the Finnish Meteorological Institute (FMI) following the objectives of the parent YOPP project.

The main aim of this thesis is to research the feasibility of providing forecasts and observations data to vessels operating in the Arctic region. This is carried out by cooperation among Navidium Plc, FMI and KNL Networks all of whom operate in Finland. The forecast delivery network is provided by KNL, software development is handled by Navidium and FMI provides forecasts as well as satellite observations in addition to supervising this project.

This thesis then describes various development stages of the project development which started with analyzing the current product of Navidium and measures that would be required to make it compatible with the project. Moreover, methodologies such as the use of Scrum for agile development are briefly discussed. Then, this thesis provides requirements of the project which is based on the location of components installed which is Vessel and Shore modules.

The project implementation fulfilling most of the requirements is described with proper workflows to explain the process. Objectives that could not be achieved are provided with reasoning for that to occur. At the end of the implementation, the thesis summarizes the achievement of most of the requirements.

At the end section of this thesis, the results are briefly discussed. The problem that arose after the deployment is mentioned along with the suspected cause of it. Finally, this thesis addresses the need for further improvement in the quality of an already quite complete project.

Keywords	YOPP, Weather Forecasting, Arctic, SAR, FMI

Contents

List of Abbreviations

BBOX	Bounding Box
ECMWF	The European Centre for Medium-Range Weather Forecasts
EPSG	European Petroleum Survey Group
FMI	Finnish Meteorological Institute
GPS	Global Positioning System
HTTP	Hypertext Transport Protocol
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
NMEA	National Marine Electronics Association
PPP	Polar Prediction Project
REST	Representational State Transfer
RMC	Recommended Minimum Specific GPS/TRANSIT Data
SAR	Synthetic-aperture radar
SMTP	Simple Mail Transfer Protocol
TCP	Transmission Control Protocol
URL	Uniform Resource Locator
UTC	Coordinated Universal Time
YOPP	Year of Polar Prediction

WMS	Web Map Service
WMO	World Meteorological Organization
WGS	World Geodetic System

1 Introduction

In a time where we are facing the biggest challenge of climate change, the polar regions of the planet Earth have seen a lot of interests. Those attentions are evident by the increased marine traffic carrying out both commercial and non-commercial activities in the polar region. As such, there has been demand high quality of good forecasting and observation coverage. These improvements can only be achieved with proper research and investment regarding the current issues and known solutions to such issues.

This thesis in itself is an activity under the Polar Prediction Project (PPP) undertaken by the World Meteorological Organization’s (WMO). Through the YOPP initiative, WMO is promoting international research and cooperation among various profits and non-profit organizations to collectively enhance the weather and environmental forecasting in the polar region. The FINYOPP project is a collaborative project between Finnish Meteorological Institute (FMI), Navidium Plc and KNL Networks.

The main aim of this thesis is to carry out research on the feasibility of delivering weather forecasts and observations to vessel mainly operating in the Arctic polar region. In order to accomplish this research, FMI is supervising the project. KNL networks, a Finnish company based in Oulu, would be providing the data delivery solutions [1] and another Finnish company, Navidium Plc employs the writer of this thesis and is responsible for the handling of the end-to-end software development necessary for the completion of this project.

There are 6 sections in this thesis. After the Introduction, Section 2 enlists main project requirements. Section 3 provides the readers with information about software tools and terminologies that are used during the project implementation. Project implementation is described in Section 4. And lastly, the two sections wrap up the thesis with the results and finding as well as the overall conclusion of the project.

2 Project Specifications

2.1 Materials and Methods

The thesis followed the Agile method of software development, specifically the Scrum framework. The main reason for choosing Scrum was its simplicity. Also, according to Schwaber and Sutherland [2, p. 5], co-creators of the scrum, the optimal team size suitable for using Scrum framework is between 4 and 8 members. As our team of four members fell in that category, Scrum was chosen to be an agile framework of choice for the project.

Due to the requirement for the initial version of the product to be ready in two and half months' time, each Sprint cycle was thus selected to be one week long unlike the usual cycles of two weeks or one month. Each Monday, the sprint planning was carried out where relevant features and bug fixes were to be included along with responsible team members to implement them. In order to increase the effectiveness of the sprint, a rotating scrum master technique was set in place. This provided each team member the opportunity to be a scrum master for a sprint duration. The proactive participation of each member was clearly visible after such practice was established.

The Kanban board created in a whiteboard was favored over web services like Trello and JIRA. This decision was made collectively by the team taking into consideration the fact that all except one team member, were working from the same office. As a result of which, team discussions and understanding of other team member's work were clearer in comparison to other projects using web services as Agile board.

The software and the hardware used by team members varied as most of the sub-projects were independently developed but keeping each other well informed about one's progress. A member who was working on the .net framework was primarily using a windows machine with the visual studio as the preferred Integrated development environment (IDE) where other members preferred working in Visual Studio Code (VS Code) text editor. Regardless of the IDE's used, the team members used git version control to host the source code. Bitbucket.org, as with the other projects in the company, was the git hosting provider.

2.2 Current state analysis

After selecting KNL Networks as the data delivery provider for FINYOPP research project [3, p. 9], FMI and Navidium Plc came into an agreement for overseeing software development aspect. As per the agreement, Navidium would develop software for shore as well as vessel modules called Back Office and Remote System respectively. The plan was to use the pre-existing product of Navidium, Marecast which along with many other capabilities, had a feature of providing forecast data to Remote Systems installed in its customer's vessels.

One of the major changes required by Navidium to adapt the Marecast platform in FINYOPP was the technology used for data transfer. Marecast primarily used HTTP connections over TCP but in FINYOPP, the KNL data delivery system only supported SMTP. In addition to the change in transfer protocol, the project also required to support both Windows as well as Linux operating systems for the Remote system.

After the initial study by teams at Navidium and FMI, based on the requirements for two separate modules, a list was created with detailed information about the requirements.

2.3 Remote system requirements:

The section below provides the list of requirements determined for the Vessel Module called Remote system.

2.3.1 Integration of Remote System with email server in KNL device

The requirement of SMTP, IMAP, and POP3 protocols being supported by KNL device means that the Remote System should be able to support either of these protocols to receive forecast images. In order to fulfill that purpose, there is a need for email client in Remote System which subscribes to the KNL device's mail server As per KNL's briefing, locations with access to terrestrial networks like 3G, the device supports full 3G speed which is about maximum of 21 Mbps whereas in other places the bandwidth range is limited from 2 to 150 Kbps and individual email size maxing 200KB in size.

Due to this limitation, the forecast image file size has to be reduced significantly. Along with that, forecasts have to be synchronized automatically since users cannot manually refresh forecast fetching.

2.3.2 Basic Weather Station View with Animation player

The landing page in the web application is the Weather Station View. The design of the weather station view has to adhere to the design specification provide by the company's designer. Users should be able to select specific weather forecast property such as wind, fog, pressure, etc. from the right sidebar resembling the one in figure 1.

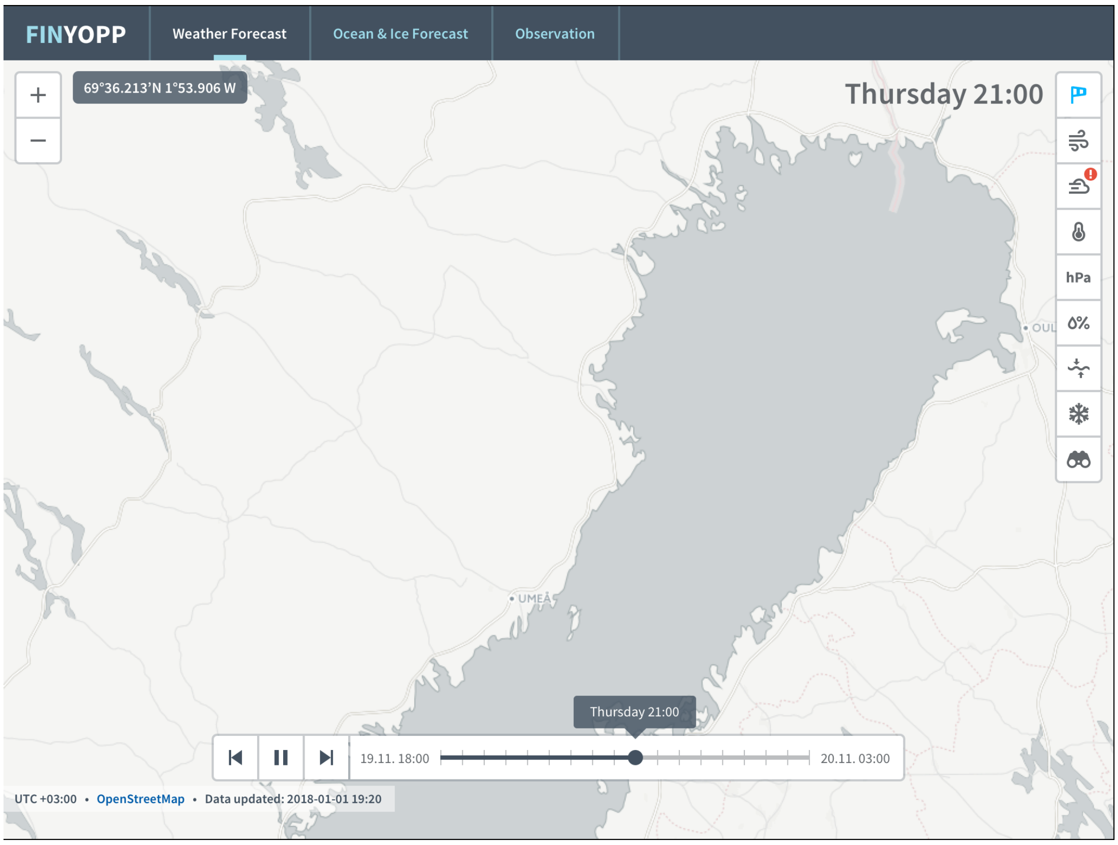


Figure 1. The initial design of Weather station view including an animation player

Based on the number of forecast images available for a particular weather property, there has to an animation player with a number of steps equaling the number of images. This allows the user to cycle through each image on the press of a button. The player should show the date and if applicable also the time of forecast. Most importantly, animation player should not be shown for Observation view. And lastly, it should follow design specification similar to one shown in Figure 1. Along with these, relevant information like mouse coordinates projected in a map has to be visible on the top left corner.

2.3.3 Notification to Remote System user about new content

The user should be notified of a new forecast that has arrived. There could be a red notification icon next to a weather property as shown in Figure 1. It must be possible to control when the user can see a new animation. All images or sometimes even a single image might not arrive in the same email. Thus, the user should either be shown notification as soon as a single image arrives or only after a batch of images arrive.

2.3.4 Getting Remote System position from KNL's radio

The position of the vessel has to be read from KNL's radio device, which provides GPS coordinates in the form of NMEA 0183 sentences. The Remote system should connect to the KNL device via ethernet cable and receive the NMEA sentences over TCP connection. Position, speed and heading information provided in the sentences. The location data received from the KNL device’s radio would have to be shown on the map.

2.3.5 Populating forecast or observation data from files

Once a forecast or observation has been received as an attachment in an email, it must be ingested into Remote System so it can be rendered in the map for the user to see. The content would be PNGs, so it could be stored in the file system and plotted to the correct position on the map. The metadata required such as bounding box coordinates would have to be embedded in the forecast image name.

2.3.6 Forecast images and offline Maps in Mercator and Polar projections

As the vessel would be operating near or at the North pole, the map projection would be useful to be switchable to something else than Web Mercator. Some variant of Polar projection could be used.

Alongside providing forecast images in the Polar projection, the system also needs to work without an Internet connection. Thus, the map accessible in the Remote System must be available offline. The offline map installed must have to be supported by the Mapping library used in the web application.

2.3.7 Day / Night mode for UI components

The users of the vessel where the FINYOPP is getting would be operating from a dark cockpit. As such, it could be beneficial for the user to have the option of switching between light and dark themes.

2.4 Back Office requirements

The following section enlists requirements identified for the Shore module known as Back Office.

2.4.1 Getting vessel location

KNL networks provide API access to the fetch location of its devices. Back Office (BO) should connect to this API frequently (roughly once every couple of hours) and send forecasts accordingly.

2.4.2 Getting vessel specific data from FMI

Based on the location of the vessel, BO would be required to get forecasts from FMI. The reason for getting them based on the location is to limit the amount of information in the forecast to the absolute minimum needed by that vessel.

The existing API access to FMI's services would have to be utilized for pushing ECMWF weather forecasts. Any additional mapping provided in FMI's Weather Map Service (WMS) server could also be utilized.

2.4.3 Sending forecast files to Remote System

The main responsibility of BO would be to send forecast emails and observations to Remote System at regular intervals. The intervals would be set by according to the previous and current location of the vessel calculated in a time period of a couple of hours. Forecast image file name should include unique feature id with bounding box and date and time of forecast.

3 Theoretical Background

3.1 Weather Forecasting

Weather forecasting refers to the art of using science and technology of today to predict the atmospheric conditions sometime in the future. In order to make the weather forecasting possible, enough data related to current conditions such as humidity, wind, and temperature has to be gathered first. These gathered data then serve as a basis for using meteorology in order to predict the way atmospheric changes may occur. Thus, the accuracy of any forecasting system depends on the amount of such data collected about a certain atmosphere. [4]

Technological advancement has made the forecasting more accurate than how it used to a decade before [5, p. 1]. But even with all the sophisticated technologies, the same fundamental methods apply for forecasting which is making observations of the conditions. Such observations over a period of time act as a foundation for predicting the atmospheric state such as wind, fog, pressure, etc. in the future. Being important in basic prediction such as rain, wind, and temperature, weather forecasting is specifically important for the aviation industry.

There are two types of forecasting based on the duration of the forecast, short-range and long-range. In Short range forecasting, meteorologists provide a forecast for an atmosphere in the range of up to 48 hours. This method uses observations from radar and satellite over a certain atmosphere and then those observations are combined and processed by computers. Then, computers generate forecast data. Alternatively, long-range deals with forecasting over a very long period such as weeks and months. The significance of such a forecast is that they provide a snapshot of weather or climate changes over a long period of time.

As a generation facing the real threat of climate change, it is more important than ever to invest in technologies capable of quickly and accurately predicting severe weather conditions. Those technologies, in the long run, will be responsible for saving countless lives of humans as well as other animals. So, there is a need for cooperation among countries, companies, and individuals regardless of their socio-economic background to tackle this threat. [6]

3.2 Software Technologies

The tools and technologies for software development were selected based on the initial requirements set for the Remote system. Among those requirements, cross-platform capability and nature of application meant to run as a web application shrunk the choices to few options. Moreover, tools providing better developer experience and easy maintainability were given preference. As a result of all these considerations, the technologies described below were chosen.

3.2.1 React

React is a JavaScript library developed by Facebook to build user interfaces [7]. The React project is of opensource nature and is maintained by Facebook and opensource contributors. It follows a declarative pattern to allow developers to build web applications of small or big in size. React can be used in a small portion of the already existing multi-page application or used to develop full-fledged Single Page Applications (SPA). Although, in order to develop SPA using React, a separate routing library has to be used as React does not provide itself. A third-party library like React router can be used in such a case. [8]

Applications developed using React usually have multiple components. These components work together, with their own individual styling and behavior, similar to how Lego bricks work [9, p. 476]. As per the results from [10], the two most admired facet of React is its programming patterns and rich ecosystem with packages available for varying use cases.

3.2.2 Node.js, Express.js, and npm

Node.js is a cross-platform JavaScript runtime which is based on Google’s V8 JavaScript engine present in its Chrome browser. It is an asynchronous, event-driven platform providing the capability to JavaScript applications in server environments. Both V8 and Node.js are opensource projects maintained by a group of designated people with taking feedback and suggestions from the open source community. [11] [12]

Express.js is a web application framework which runs in Node.js. It provides a thin layer of abstraction and does not affect core Node.js features. Features such as Routing, middleware, templating and database integration comes along with Express. It is also an open source project and with the help of the community, express has a large number of third-party tools and services integrations. [13]

Npm, commonly called Node Package Manager is the Node.js modules manager that comes alongside every Node.js installation. It is a tool used for sharing JavaScript code among developers and contributors either privately or publicly. The company behind npm tool, Npm Inc., maintains a public registry called npm from where developers and contributors alike can fetch and publish Node modules. Npm is claimed to be the world's biggest software repository. [14]

3.2.3 Email server and client

An Email is an electronic form of mail used for exchanging messages. These messages are primarily in plain text format. Although, nowadays email is extended for sending and receiving different types of files such as images, audio, documents, etc. Regardless of the content of an email, the same architecture model of Client-server is used today as well.

In a client-server, a device fulfills both the roles of being a client as well as a server. The device acts as a server by keeping data and providing the data to clients who connect with it. Similarly, it is able to connect to other devices to receive data thus operating both as server and client. Email servers are the most popular devices adhering client-server architecture. The clients are able to connect and fetch emails over Simple Mail Transfer Protocol (SMTP) from the device. Along with that, the server itself can connect to another server using the SMTP protocol. [15]

3.2.4 Web Map Service and Tiled web map

Web Map Service (WMS) is a standardized mapping protocol developed by the Open Geospatial Consortium (OGC) [16]. The standard describes the HTTP API structure in order to request geo-referenced map pictures from one or more distributed geospatial databases. The response of such requests is common image formats such as PNG and JPEG. [17]

Tiled Web Map is also known as Slippy Map service, is a form of web map usually displayed in a browser after fetching multiple small tiled images from the internet and joining them together. The common image tile dimension is 256 x 256 pixels in PNG format. A Slippy map client assumes tiles are delivered by a web server in the URL format provided in Listing 1.

```
url/zoom-level/x-coordinate/y-coordinate.png
```

Listing 1. A slippy map service URL format

The above-mentioned format is referred to as XYZ Tilenames and describes the base URL of the web server, zoom level, x, and y coordinates value of requested map tile.

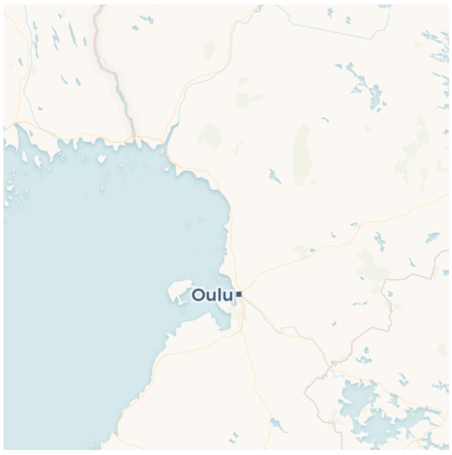


Figure 2. An image response from a Tile server

The HTTP response of the Listing 2 returns a 256 x 256 pixels PNG tile image as shown in Figure 2.

```
https://b.basemaps.cartocdn.com/rastertiles/voyager/6/36/16.png
```

Listing 2. A map tile URL requesting tile image for zoom level 6 with x-coordinate of 36 and y-coordinate of 16

3.2.5 Cartographic projections and NMEA sentence

Cartographic projection or a map projection describes the transformation of longitudes and latitudes coordinates from a sphere or ellipsoid's surface into a plane [18]. Today, one of the most famous map projections, Spherical Mercator, is viewed by millions of people on the internet from sources like Google maps, Open Street Maps [19]. Recently, Google maps replaced Mercator projection with spherical Earth in lower zoom levels [20].

North Pole LAEA Europe with EPSG:3575 also known as Lambert Equal-Area Azimuthal projection is another form of map projection. It covers an area north of 45°N and thus includes the Arctic region.

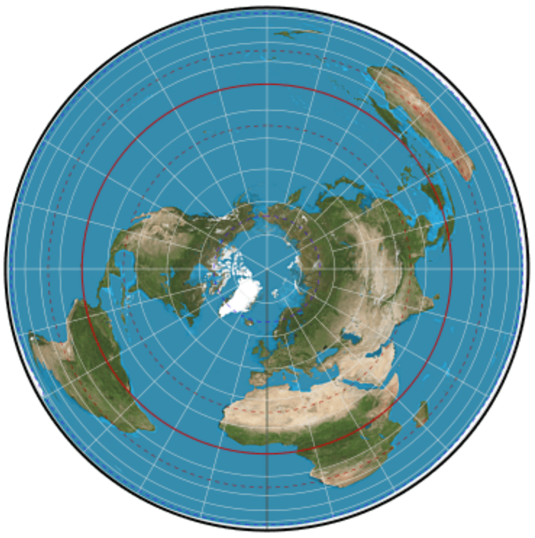


Figure 3. Lambert Equal-Area Azimuthal projection with the Polar center. Copied from [21]

A view of EPSG 3575 placing North pole in the center is shown in Figure 3. It is commonly used for research in the Arctic region.

With the library called Proj4, it is possible to transform coordinates from one map projection into another. Proj4 has mappings for different programming languages, frameworks, and libraries making it very useful whenever there is a requirement for such re-projections of images or maptiles. [22]

NMEA 0183 is a standard defined by The National Marine Electronics Association (NMEA) for data transfer interface in Marine Instruments. The “0183” in NMEA 0183 denotes the release year of the first version which was 1983. An example of an NMEA sentence is provided in Listing 3. [23]

```
$GPRMC,190306.828,A,6010.458,N,02456.997,E,,,100419,000.0,W*74
```

Listing 3. An example of NMEA 0183 sentence

The NMEA sentence in Listing 3, upon decoded by a valid decoder provides the information listed below.

Table 1. An example of NMEA 0183 decoded of Listing 3

Field	**Value**
Position	60.174300°N 24.949950°E
Timestamp	Wed, 10 Apr 2019 19:03:06 UTC
Close to	Helsinki, Finland
Local time	Wed, 10 Apr 2019 22:03:06 EEST
Time zone	Europe/Helsinki (UTC +0300)

Some special diagnostic messages, in addition to standard NMEA sentence, may also be supplied in the NMEA sentence [24].

3.2.6 Other technologies and terminologies

The following maintains a list of important tools and technologies which are related to various components in this project:

- WebSocket is a standardized communication protocol which provides two-way data sharing capabilities for connected devices. It is mostly used for real-time data transfer among
- Coordinated Universal Time, commonly referred to as UTC is a time standard identified by International Telecommunication Union. As per UTC, time is divided into days, hours, minutes and seconds. At any moment, the UTC time is the same throughout the world.
- Create-react-app is a quick starter development tool to start a React.js application with integration to Babel and Jest JavaScript libraries. [25]

4 Implementation

This section elaborates the techniques used for developing weather forecast delivery system over the SMTP protocol. The main focus of the section is development carried our two different modules, Vessel module called Remote System and Shore module called Back Office. Based on the requirements discussed in Project Specifications, technology selections were made by the project development team at Navidium. The initial implementation started roughly 8 weeks before the planned deployment on the 21st of July 2018.

4.1 Architecture Design

The architecture of this project is drawn as per the discussion among a team of four members. It is mainly divided into Remote System and Back Office components as shown in Figure 4. Additionally, each of these components has individual subcomponents responsible for a certain aspect of the module itself.

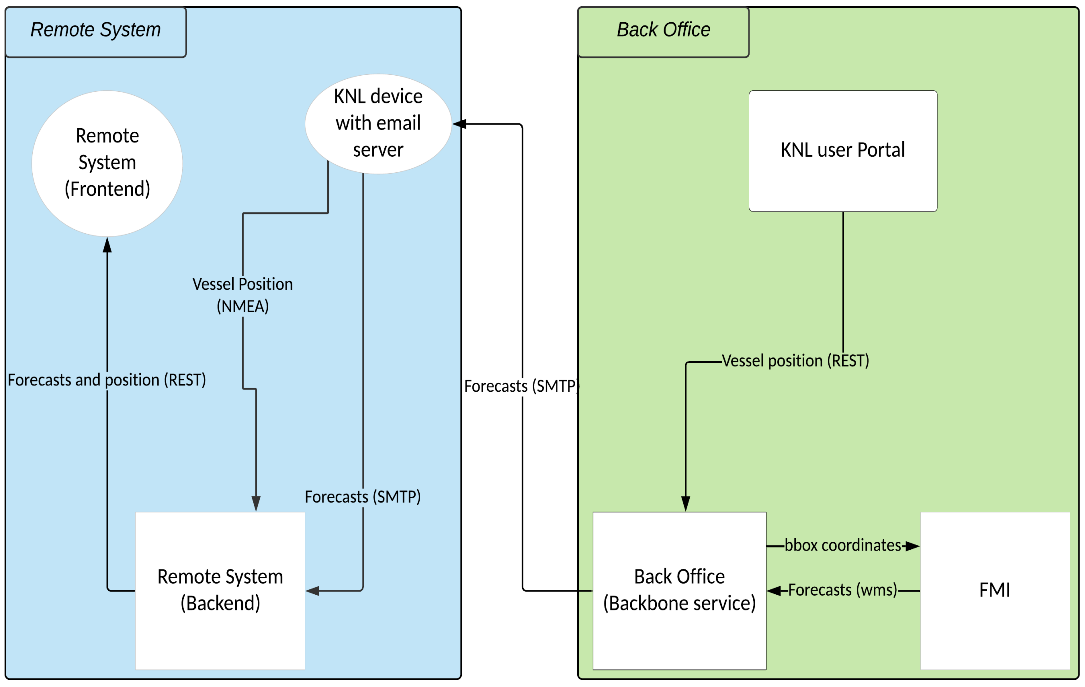


Figure 4. The architecture of FINYOPP based on the location of modules installed

Remote system component in Figure 4 demonstrates workflow starting from receiving of forecast emails until displaying those on the map. Similarly, the Back office component describes the start of the forecast chain by initially collecting vessel location and requesting weather forecasts and observations based on informed received from KNL API.

The subsections listed below are based on implementation contexts, Back Office and Remote System.

4.2 Back Office implementation

The software stack of Back office mainly consists of service written in .net framework version 4.7 and Microsoft SQL Server 2016 as the database. It is designed to send forecasts images as email attachments to Remote System, based on the location information received. The bounding box (bbox) is calculated using the vessel location and through their API gateway, FMI is requested to provide with forecasts or observations for the particular bbox.

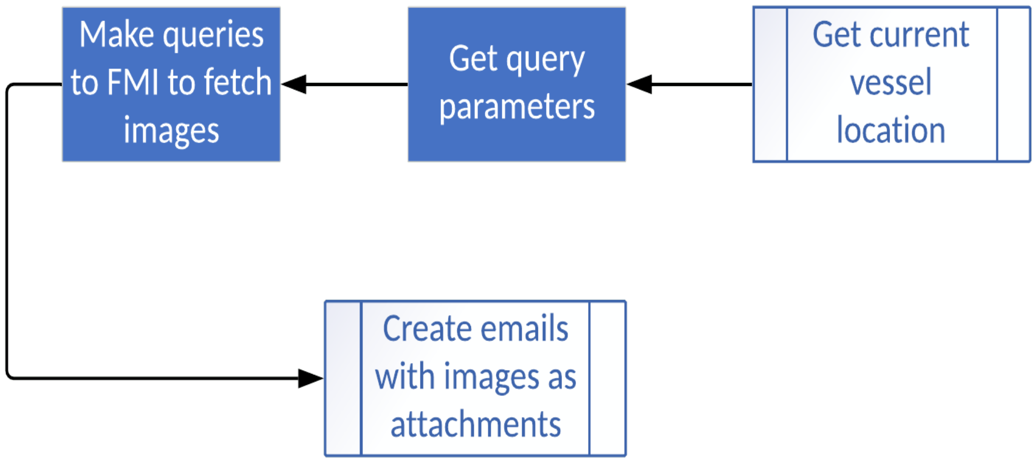

Make queries to FMI to fetch images
Get query parameters
Get current vessel location
Create emails with images as attachments

Figure 5. The workflow of the Back Office system

As illustrated in Figure 5, there are multiple processes involved to complete the workflow in Back Office. The subsection below goes through such processes in detail.

4.2.1 Receiving vessel location

Vessel location is received in Back Office through REST API provided by KNL. Using the API, the following information is received:

- Unique id
- Name of vessel
- Bearing or heading of the ship in degrees
- UTC time of last received data
- Speed in knots

Listing 4 provides an example query with the received response from KNL API.

```
// Endpoint URL for stations data:
https://portal.knlnetworks.com/api/v1/stations

// An authorization token has to be included either in the header or as a URL
parameter. The token can be received from:
https://portal.knlnetworks.com/api/v1/token


// Example of received JSON data:

[
    {
        "bearing": 345.882,
        "ok": false,
        "name": "Oden",
        "station_status": "red",
        "longitude": 12.7006,
        "rtposition_id": 66885983,
        "station_id": 136,
        "time": "2019-05-20 09:18:46.932120",
        "latitude": 56.0253,
        "speed": 0,
        "id": 136
    }
]
```

Listing 4. JSON response received from a request to KNL API

Based on the received JSON data, it is then possible to receive the location of the vessel which in this case is Longitude: 12.7006 and Latitude: 56.02353.

4.2.2 Creating Bounding box from vessel position

After receiving the vessel position, BO backbone service calculates the bounding box. In the case of the Mercator projection, the bounding box is rectangular in shape and in Polar projection, it is square and covers a larger area.

For the vessel coordinate of Longitude: 12.7006 and Latitude: 56.02353, the bounding box calculation is provided in Formula 1.

$$\text{bbox for Mercator Projection} = (\text{lng} - 3), (\text{lat} - 3), (\text{lng} + 3), (\text{lat} + 3) \qquad (1)$$

In Formula 1, bbox refers to the bounding box, lng refers to longitude and lat refers to latitude of Vessel position. Using the formula, the bounding box is calculated as 9.7006, 53.02353, 15.7006, 59.02353. This formula is included in the BO backbone service code-base and it does calculation automatically.

4.2.3 Query to FMI API for forecast images

The bounding box generated is a requirement in order to fetch weather forecast images. In order to receive the images, a request has to be placed to FMI's WMS server. The response format can be set in the URL parameter. An example of wind forecast fetch for the bounding box calculated from Formula 1 is listed in Listing 5.

```
GET: https://data.fmi.fi/fmi-apikey/API_KEY/wms?service=wms&version=1.3.0&re-
quest=GetMap&lay-
ers=fmi:ecmwf:wind&styles=&crs=EPSG:4326&bbox=9.7006,53.02353,15.7006,59.02353
&width=1080&height=640&format=image/png&transparent=true&time=2019-05-
31T06:00:00Z
```

Listing 5. An example of forecast fetch request to FMI's WMS server

The URL in Listing 5 requires a valid key in place of “API_KEY”. After the authorization from FMI server, the forecast image below is received:

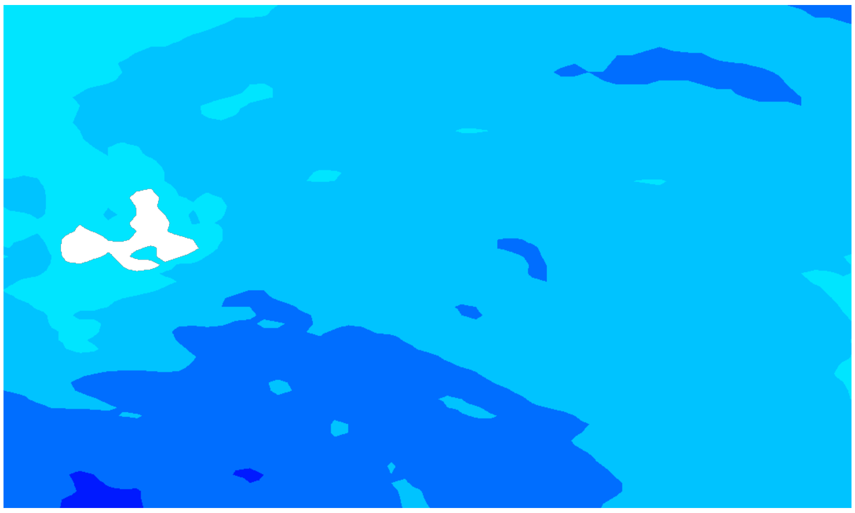


Figure 6. A wind forecast image received after querying FMI WMS server with Listing 5

Similarly, other forecasts for different areas are fetched by BO backbone service. A list of available weather properties is maintained in Navidium's own documentation.

4.2.4 Sending emails to Remote System

After fetching forecast images based on vessel location, the final work of BO backbone service is to send those images as email attachments. A preselected email address is used for forwarding the images to the Remote System which receives emails at the same email address. While sending the emails, BO is required to name the individual file with information about weather property, forecast date time and bounding box of that forecast. Also, the email includes batch id in the subject field to inform Remote System about the number of files sent in a specific batch id.

```
5_20190531T060000Z_9.7006,53.02353,15.7006,59.02353.png
```

Listing 6. The naming format used for forecast image

In listing 6, the first part, 5, denotes feature number 5. From Appendix 1 (Table 1), feature 5 is the wind forecast. The second part until the "_" character and the third part after the character refers to forecast time in UTC and bounding box of the forecast. The Tables in Abstract 1 maintains a detailed list of weather properties.

4.3 Remote System implementation

The Remote system mainly consists of four components namely Mail fetcher client, NMEA Reader service, Finyopp UI backend, and Finyopp UI frontend. These components are installed in a Linux Mint operating system with a Graphical User Interface (GUI).

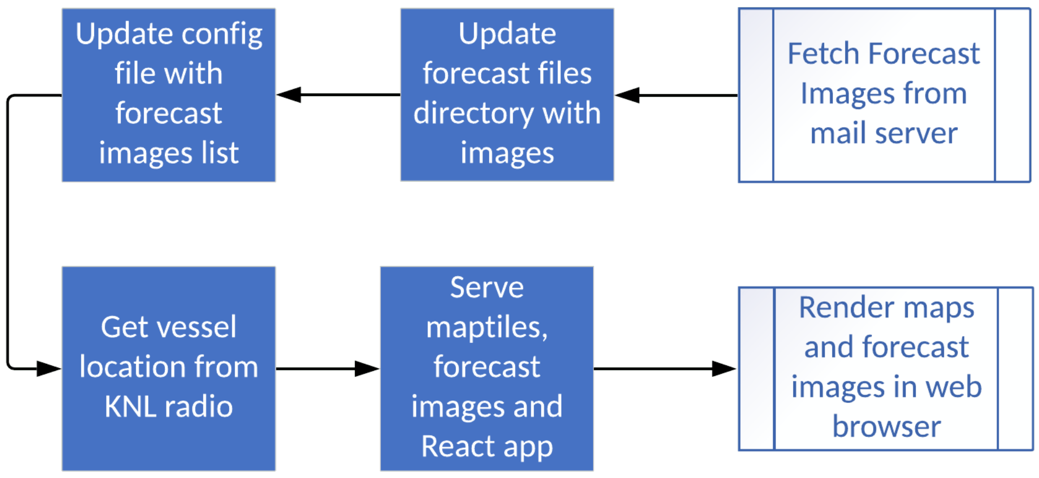

Update config file with forecast images list
Update forecast files directory with images
Fetch Forecast Images from mail server
Get vessel location from KNL radio
Serve maptiles, forecast images and React app
Render maps and forecast images in web browser

Figure 7. The workflow of the Remote System

The subsections below describe each role and mention the component responsible for fulfilling that role.

4.3.1 Fetch forecast from the mail server

This process is handled by the mail fetcher client. It is scheduled to run every 15 minutes and is allowed to run for a maximum of one hour by the Operating system. The code for the client is written in Python programming language. The workflow of this process is listed in Figure 8.

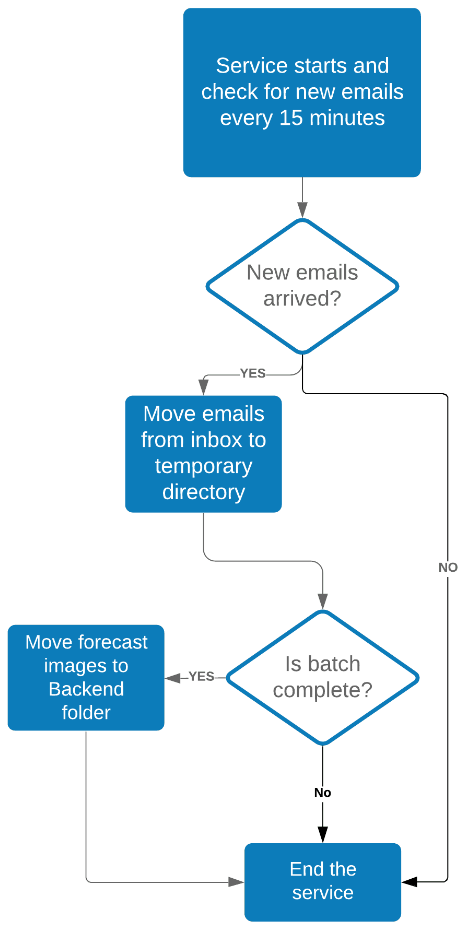

Service starts and check for new emails every 15 minutes
New emails arrived?
YES
Move emails from inbox to temporary directory
NO
Is batch complete?
YES
Move forecast images to Backend folder
No
End the service

Figure 8. The workflow of Mail fetcher service

The workflow starts with the running of service which is restarted every 15 minutes by the system operating system. In the first step, mail fetcher connects to KNL email server, and checks for new emails. If there are new emails, it then moves those to a temporary directory. Otherwise it the process is ended. Then, based on the number of files in emails and the batch information received, the batch is deemed either complete or incomplete. If the batch is complete, those files are moved to UI backend directory otherwise they are left as is until next batch arrives. The workflow thus ends with stopping of the service.

4.3.2 Vessel location from Radio

The KNL device installed in the vessel provides a location using NMEA sentences. These NMEA sentences are similarly structured as provided in Listing 3. The NMEA service reader is component installed in Remote System reads, parses and forwards data from the NMEA sentence in an appropriate structure. The formatted data structure is provided in Listing below. The code for this component is written in C language and runs as a Linux daemon.

```
// data is parsed from:
$GPRMC,190306.828,A,6010.458,N,02456.997,E,,,100419,000.0,W*74

// to this:
60.174300  24.949950, 20190410T190306Z
```

Listing 7. NMEA sentence data parsed to another format

The parsed text is then utilized by the Backend to send location coordinate to the frontend.

4.3.3 Finyopp UI backend

The backend of Finyopp UI is a JavaScript application running in Node.js. It is responsible for the following processes:

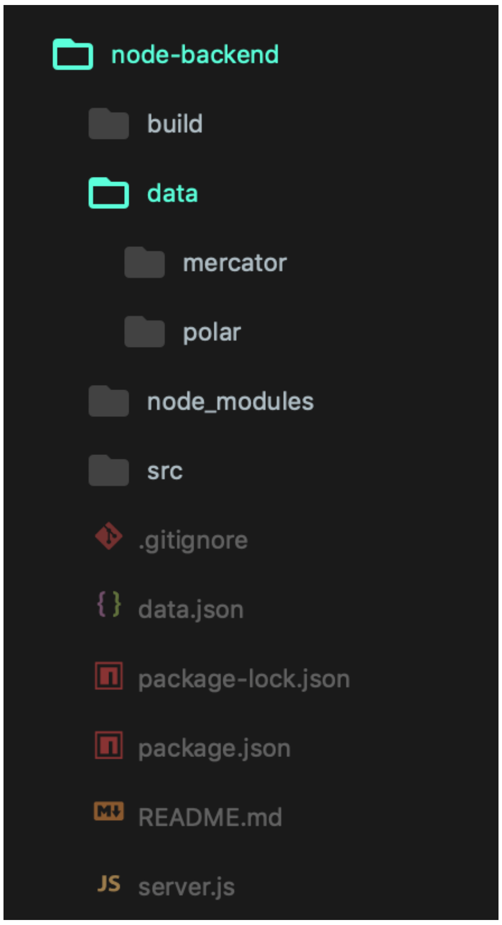
- Watching the forecast directory for new forecast images.
- Updating config.json file with lists of files inside forecast directory.
- Receiving location data from NMEA reader service.
- Serving Mercator and Polar projections Maptiles
- Serving UI Frontend with forecast images

The following node modules are used in order to facilitate tasks to be handled by the backend:

- Pm2 serves both Frontend application as well as Maptiles directory.
- Express.js module provides REST API for the Frontend with an exposed endpoint for the config file, vessel location, and forecast images.

- Socket.io provides real-time connectivity between frontend and backend. This allows the backend to immediately send an updated list of forecast images once there are changes in forecast image files.
- Svmq module extends Node.js capability to read from UNIX System V message queue. It is required as NMEA service publishes location data as System V message queue.

The directory structure for the backend component is shown in Figure 9



```
node-backend
  build
  data
    mercator
    polar
  node_modules
  src
  .gitignore
  data.json
  package-lock.json
  package.json
  README.md
  server.js
```

Figure 9. The directory structure of Finyopp UI Backend project

The build directory hosts Frontend code generated by the build step in create react app. Similarly, data directory, containing Mercator and polar subdirectories, is where the forecast images are placed by mail fetcher. This directory is continuously checked by Node.js process for changes in contents. The data.json file in the root directory is served by the backend to the frontend as a list of all forecast images that are available in the Remote System.

There are three REST endpoints exposed by Node.js application running in port 9000. The endpoints are:

- /api/config endpoint responds with the content of the data.json file, which is a JSON file listing the content of the data directory.
- /api/gps endpoint provides JSON data with “updatedTime” and “gpsCoordinates” of the vessel read from KNL device’s radio.
- /mercator/5/5_20180711T040000Z_61.3663183333333,17.480015,65.3663183333333,23.480015.png is an example endpoint of an image file served at port 9000 by the backend.

The Maptiles are served from a separate directory named Maptiles. The Maptiles directory has two subdirectories, “mercator” and “polar” consisting of mercator and polar projected Maptiles respectively as listed in Listing 8.

```
./maptiles
├── mercator
│   ├── 3
│   ├── 4
│   ├── 5
│   ├── 6
│   └── 7
└── polar
    ├── 1
    ├── 2
    ├── 3
    ├── 4
    ├── 5
    ├── 6
    └── 7
```

Listing 8. Maptiles directory tree view

Mercator map is supported from zoom levels 3 to 7 whereas Polar map has support from levels 1 to 7 as evident from listing 8. These offline packages are downloaded from [26] and [27] which provides map services to use freely for research projects like this project.

4.3.4 Finyopp UI Frontend

The Frontend of Finyopp application running in a web browser is a Single Page Application built with React.js. The UI of the application is developed based on the design guidelines provided by the designer of the company. It takes into consideration all the requirements discussed in Project Specification. The components of the frontend application are briefly discussed below:

Weather station view with animation player:

The frontend application has a weather station view along with an animation player to automatically cycle through forecast images in one second time duration. The animation player can be paused in while playing and that stops the player while maintaining the last current progress of cycling through the list of images.

Page routes with Weather forecast, Ocean and Ice forecast, and Observation:

There is routing capability in the application and the users are able to navigate to different pages based on the feature that they are wanting to use. If a certain weather property is selected before navigating, then visiting back to the previous page maintains the selected state. This provides a pleasant user experience and different routes of the application collectively behave like a single application.

Polar and Mercator projection image forecasts:

Both Polar and Mercator projected image forecasts rendered are properly implemented during the development. A projection toggle is programmed to appear once the vessel location is found to be above 75 degrees latitude. In comparison to Mercator projection, the bounding box is fixed and bigger for polar projection. This allowed a polar forecast image to cover and could be used for a large area. The area of the bbox was set mostly focusing the North Pole.

In the case of polar projection, the forecast images required re-projection from EPSG:4326 to EPSG:3575, while rendering them on the map. In order to achieve this, the JavaScript port of Proj4 called Proj4js is used. The code snippet listed in Listing 9 is used for transforming forecast image from EPSG:4326 to EPSG:3575 and providing the capability to Leaflet mapping library to properly display polar map tiles.

```
// START: Calculates polar projection crs with fixed projection bounds
let max_zoom = 7;
let tile_size = 512;
let extent = Math.sqrt(2) * 6371007.2;
```

```
  let resolutions = Array(max_zoom + 1)
    .fill()
    .map((_, i) => extent / tile_size / Math.pow(2, i - 1));

  let crs = new L.Proj.CRS("EPSG:3575", "+proj=laea +lat_0=90 +lon_0=10 +x_0=0
+y_0=0 +datum=WGS84 +units=m +no_defs", {
    origin: [-extent, extent],
    projectedBounds: L.bounds(L.point(-extent, extent), L.point(extent, -ex-
tent)),
    resolutions: resolutions,
    worldCopyJump: false
  });
  let polarBounds = new L.LatLngBounds(new L.LatLng(61.4021, -124.99), new
L.LatLng(61.4021, 55));
  // END

// In the code above, L refers to instantiated Leaflet library import
```

Listing 9. JavaScript code to transform geospatial coordinates from EPSG:4326 to EPSG:3575

The use of snippet listed in Listing 9, Polar maps and forecast images in EPSG:3575 are properly rendered by the Leaflet library.

Day / Night theme and Notification to Remote System User:

The requirement of day-night themes could not be achieved due to time constraints. Also, the requirement of notifying the user of new content could only be completed partly. This was also mainly due to a lack of time allocation for this feature. But, a part of the functionality could be implemented. In spite of notification not being shown, the Frontend is immediately updated with new forecast images URL by the backend once there are new forecasts available. This is possible by using Socket.io which in turn uses WebSockets protocol.

5 Results and Discussion

As elaborated in the Project implementation, most of the requirements set initially were achieved by the end of the implementation phase. One optional and another important requirement could not be achieved mostly due to time constraint rather than technical difficulty. The initial feedback received from the vessel users were mostly positive. The users liked the easy navigation within the application. In addition to that, information such as mouse coordinates and Vessel location being readily available was regarded as a positive aspect of the project.

In terms of data delivery, the data transmission very rapidly and thus impacted the sending and receiving of the emails. This was not a problem when the vessel was within reach of terrestrial networks i.e. whenever the vessel was near the coastlines. But once it was completely relying on the KNL's network, the successful data delivery was less than 50%. This rate could be known as KNL maintained statistics of the emails sent and received in its systems.

Remedy measures such as reducing the size of emails were taken immediately once the issue was known. The packaged email size was reduced by more than 70% by compressing the forecast image quality. This brought down the size for all the forecasts to roughly 200 KB. Once the vessel crossed Svalbard for Arctic expedition, not many emails were seen as received.

As per the email communications between FMI, Navidium and KNL networks, the initial problem was pointed out to be some of the devices brought on board in the vessel. As the vessel was traveling with a group of scientists for research expeditions, there might have been equipment which disturbed radio signals received by the KNL device radio antenna. Also, further investigation showed a similar pattern of radio noise in the Svalbard area possibly emitting from a nearby university. As of this thesis writing, further findings are not known about this matter.

6 Conclusion

The importance of the Polar region is only going to increase regardless of whether or not there are enough researches carried out to facilitate highlight its importance. As a result, there would be more business and non-business-related activities that would be carried out. The sooner the government institutions or companies realize this potential the better. As evident by this project, the investment in research related to forecasting prediction in the polar region is increasing.

For the longest time in history, Arctic has been considered a not-so-important area to operate business or only place where scientists conduct their research. As we are challenged by Climate change and its adverse effects, the place to look and prove about those facts for those who ignore would be the North Pole region. The vast amount of ice melting may open up Marine traffic route and on the contrary that raises the question the danger it possesses by facilitating of sea level increase.

Thus, researches like these, may or may not provide a solution to a huge problem like Climate change, but it is, without a doubt a step-in right direction. The results of the project highlighted that the solution like these is in demand. It could be more beneficial to vessels operating in the Arctic region if there would be more research and investments towards the next iteration of this project by first overcoming the shortcomings observed by the end of the project. As per the shortcoming itself, the problem may have been solved a little while after the publication of this thesis.

References

1 Finnish Meteorological Institute to use KNL Networks for improving Arctic maritime operations - KNL Networks [Internet]. KNL Networks. 2018 [cited 2019 March 20]. Available from: https://knlnetworks.com/2018/06/05/finnish-meteorological-institute-to-use-knl-networks-for-improving-arctic-maritime-operations

2 Ken S, Jeff S. The Scrum Guide [Internet]. 2016. Available from: https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf

3 FMI. The Finnish Meteorological Institute's action plan for Finland's chairmanship of the arctic council 2017–2019 [Internet]. 2017. Available from: https://en.ilmatieteenlaitos.fi/documents/30106/42393/arktinenstrategia_fmi.pdf

4 Weather forecasting. Science Daily [Internet]. [cited 2019 March 29]; Available from: https://www.sciencedaily.com/terms/weather_forecasting.htm

5 Rose B, Floehr E. Analysis of High Temperature Forecast Accuracy of Consumer Weather Forecasts from 2005-2016. 2017 Sep; Available from: https://www.forecastwatch.com/wp-content/uploads/High_Temperature_Accuracy_Study_12_Years.pdf

6 Effects of Climate Change | Threats | WWF [Internet]. World Wildlife Fund. [cited 2019 March 28]. Available from: https://www.worldwildlife.org/threats/effects-of-climate-change

7 React – A JavaScript library for building user interfaces [Internet]. [cited 2019 March 29]. Available from: https://reactjs.org

8 Using React Router for a Single Page Application [Internet]. [cited 2019 Apr 10]. Available from: https://www.taniarascia.com/using-react-router-spa/

9 Li D. Building Enterprise JavaScript Applications: Learn to build and deploy robust JavaScript applications using Cucumber, Mocha, Jenkins, Docker, and Kubernetes. Packt Publishing Ltd; 2018.

10 The State of JavaScript 2018: Front-end Frameworks - Overview [Internet]. [cited 2019 Apr 15]. Available from: https://2018.stateofjs.com/front-end-frameworks/overview

11 Node.js Foundation. About | Node.js [Internet]. Node.js. [cited 2019 Apr 10]. Available from: https://nodejs.org/en/about/

12 V8 JavaScript engine [Internet]. [cited 2019 Apr 10]. Available from: https://v8.dev/

13 Express - Node.js web application framework [Internet]. [cited 2019 Apr 21]. Available from: https://expressjs.com/

14 About npm | npm Documentation [Internet]. [cited 2019 Apr 10]. Available from: https://docs.npmjs.com/about-npm/

15 Oluwatosin HS. Client-server model. IOSRJ Comput Eng [Internet]. 2014;16(1):2278–8727. Available from: https://pdfs.seman-ticscholar.org/e1d2/133541a5d22d0ee60ee39a0fece970a4ddbf.pdf

16 Web Map Service | OGC [Internet]. Open Geospatial. [cited 2019 Apr 25]. Available from: https://www.opengeospatial.org/standards/wms

17 Garca R, de Castro JP, Verd E, Jess M, Mara L. Web Map Tile Services for Spatial Data Infrastructures: Management and Optimization. In: Bateira C, editor. Cartography - A Tool for Spatial Analysis [Internet]. InTech; 2012. Available from: http://www.intechopen.com/books/cartography-a-tool-for-spatial-analysis/web-map-tile-services-for-spatial-data-infrastructures-management-and-optimization

18 Snyder JP, Voxland PM. An album of map projections [Internet]. United States Government Printing Office; 1453. Available from: https://pubs.usgs.gov/pp/1453/report.pdf

19 Commonly Used Map Projections | Intergovernmental Committee on Surveying and Mapping [Internet]. [cited 2019 Apr 22]. Available from: https://www.icsm.gov.au/education/fundamentals-mapping/projections/com-monly-used-map-projections

20 Google Maps says goodbye to Mercator (but only on certain scales) [Internet]. Maptorian. 2018 [cited 2019 Apr 21]. Available from: https://www.mapto-rian.com/google-maps-says-goodbye-to-mercator-but-only-on-certain-scales

21 Strebe SA. Mapthematics and Geocart Projections list - Lambert azimuthal equal-area [Internet]. [cited 2019 Apr 21]. Available from: https://www.mapthemat-ics.com/ProjectionsList.php?Projection=188#Lambert%20azimuthal%20equal-area

22 PROJ — PROJ 6.1.0 documentation [Internet]. [cited 2019 May 15]. Available from: https://proj.org

23 NMEA [Internet]. [cited 2019 Apr 21]. Available from: https://www.nmea.org/con-tent/nmea_standards/v411.asp

24 Gakstatter E. What Exactly Is GPS NMEA Data? [Internet]. GPS World. 2015 [cited 2019 Apr 25]. Available from: https://www.gpsworld.com/what-exactly-is-gps-nmea-data/

25 Create React App · Set up a modern web app by running one command [Internet]. [cited 2019 May 21]. Available from: https://facebook.github.io/create-react-app

26 Map previews [Internet]. [cited 2019 Apr 21]. Available from: https://tile.gbif.org/ui

27 maps.stamen.com/toner-lite [Internet]. [cited 2019 May 20]. Available from: http://maps.stamen.com/toner-lite/#12/37.7706/-122.3782

Weather Forecast Properties

Table 1. ID and update frequency of Weather Forecast properties

ID	Feature	Resolution	Update Frequency
5	Wind	6 hours	12 hours
6	Wind Arrows	6 hours	12 hours
7	Fog	6 hours	12 hours
8	Temperature	6 hours	12 hours
9	Pressure	6 hours	12 hours
10	Humidity	6 hours	12 hours
11	Waves	6 hours	12 hours
11	Icing	6 hours	12 hours

Ocean and Ice Forecast Properties

Table 2. Id and update frequency of Ocean and Ice Forecast properties

ID	Feature	Resolution	Update Frequency
13	Salinity	24 hours	24 hours
14	Sea Level	24 hours	24 hours
15	Sea Temperature	24 hours	24 hours
16	Ice concentration	24 hours	24 hours
17	Ice Thickness	24 hours	24 hours

Observation

Table 3. Id and update frequency of Observation properties

ID	Feature	Resolution	Update Frequency
18	SAR images	24 hours	24 hours
19	Sea Ice concentration	24 hours	24 hours