Interactive Visual Interface Development to Explore Large Quantities of Time Series Data and Patient State in Intensive Care Units

Abhishek Das
das.abhshk@gmail.com

Summary

In an intensive care unit (ICU) there are multiple patients and each patient has multiple sensors emitting data continuously. For example, heart rate is generated once every two seconds whereas frequency of electrocardiography (ECG or EKG) is greater than 120Hz. ICUs generate large amount of streaming data that needs to be analyzed and visualized in real-time.

There are numerous libraries like D3 or Highcharts to visualize data. This projects aims to develop a web-based interface where clinicians can quickly interact and understand the trends in multiple data streams of a patient along with clinical information from electronic health record, and observe the overall patient state changes. Patient state is calculated in real-time using software developed in the above mentioned algorithms project. In conjunction with the above algorithms project, this forms the visualization and interaction part of the ICU real-time decision support system to monitor multiple patients in an ICU.

Statement of Purpose

The goal of this project is to design a next-generation Intensive Care Unit monitoring visualization application that analyzes data streams in real-time and displays patient state. The focus will be on human interaction with the interface so that clinicians can intuitively comprehend trends and explore details. Population views will also be a part of this application to enable examination and assessment of trends and states of multiple patents across units or hospitals.

The web application will be based on the latest technologies (HTML5, CSS3, JavaScript, WebSockets, AJAX), which are lightweight, asynchronous, real-time and portable. This will provide a next-generation toolset for clinicians to interact and assess patient state in intuitive ways. Professional coding practices will be followed so that the code is elegant and well-organized into MVC (or MV*) and can be easily extended by future developers.

Proposed Method & Designs

LANGUAGE & FRAMEWORK

The application will be built using HTML5, CSS3 & JavaScript (client-side & server-side). Multiple frameworks and libraries will be used to accomplish different functions.

Node.js (server-side JavaScript) is the current day industry standard for building real-time, lightweight applications. It is perfect for building data-intensive and scalable applications.

Socket.IO is an event-driven JavaScript library ideal for listening to, broadcasting, and persisting data across clients. It primarily uses the HTML5 WebSockets API but gracefully falls back to long polling and JSONP in it's absence.

Node.js and Socket.IO will be used to build the server & client that transmits and listens for real-time data streams respectively. Having worked with this combination before, I am aware of how powerful, lightweight and fast this is.

jQuery is a popular client-side JavaScript library that simplifies DOM (Document Object Model) manipulation, event binding and AJAX (Asynchronous JavaScript and XML) calls.

D3.js is a JavaScript library that provides powerful data visualization components using SVG and HTML5 Canvas. D3 complies with the latest web standards and works seamlessly across a variety of web browsers.

I have worked with both HighCharts as well as D3. D3 should be preferred over HighCharts because of the level of customization it offers and it's data driven approach to visualizations.

Since this is a front-end heavy application, I would use an MV* framework such as **Backbone.js** or **Angular.js** or **Ember.js**. This would ensure a well-structured flexible codebase. Future developers would be easily able to understand and extend it.

I have previously worked with Backbone.js and am aware of it's application architecture and flow.

Still being in active development, HTML5 is not supported uniformly across all browsers. I plan to use **HTML5 Boilerplate** that helps in displaying HTML5 content in browsers which lack the native support for them. Apart from that, there are feature detection libraries like **Modernizr**, which helps us to detect the extent of functionality a browser can provide and accordingly write fallback functions (called polyfills).

I would keep the good coding practices in mind while development. For every module or feature I work on, I will make sure that the documentation is proper and unit tests are written to test the functioning across clients.

WEB APPLICATION STRUCTURE

The whole web application will be split into multiple sections. Keeping in mind to create a simple and intuitive yet informative user interface, I propose to have the following sections:

Visualizations section

This will be the main focus of the user interface. It will have several sub-sections for visualization of different data streams. All visualizations will be updated real-time. They will have support for zooming (to increase / decrease resolution) and panning (to investigate earlier data). Customization options such as organization of visualization data based on clinical feedback will be provided.

Patient State section

This section will have the current patient state information as obtained from the calculations made by the ICU decision support system. This will also be updated real-time.

Raw Data section

This section will have the raw data as obtained from the sensors. It will have several subsections for the different data streams. It will be updated real-time. Often, clinicians might want to have a look at raw data obtained from the sensors, not just the visualizations. This will also have the reports from various tests that the patient might be subjected to.

Population View section

This section will have cumulative visualization options of the entire hospital or unit. This will act as a control system for the rest of the sections. The clinician can opt for the entire hospital / unit or select a few patients whose data he wants to examine. Accordingly, the other sections will be updated to reflect that data. This is a useful way to get an overview on the patient state of the entire hospital as well as compare the data of multiple patients.

IMPORTANT FEATURES

Real-time visualizations

Powered by the Node.js backend combined with Socket.IO, the D3 visualizations would be updated real-time no matter what the frequency of data emitted from sensors. This will be fast and lightweight.

Zooming and Panning

Clinicians can examine data more carefully by zooming and panning in visualizations. Zooming would allow them to increase / decrease the resolution of data, while panning would allow them to examine earlier data.

Export options

Being rendered using SVG and HTML5 Canvas, there will be options to export the visualizations at any point of time in various vector and raster formats such as PNG, JPEG, PDF, SVG, etc.

Population views

Data streams and visualizations of multiple patients can be examined together on the same interface. This is an easy way to get a quick overview on all patients.

Emphasis on Critical Patients

Based on the data obtained, critical patients will be highlighted in the population view. This can be achieved simply by suitable colour coding of the patient along with his corresponding data visualization.

Next-generation ICU monitoring application

Using modern technologies and building a web application reduces the installation and setup barrier. Clients only need a modern web browser to open the application. The application will work seamlessly irrespective of device and operating system.

Timeline

May 28 - June 16

Get to know the mentor and read up and decide on the structure of the interface and finalize the resources to be used. The purpose of the application is not to just render graphs and images but to also be intuitive and display data correctly. We should also keep in mind that the app should be fast.

Create the skeleton and basic structure of the application.

June 17 - June 28

Set up the Node.js and Socket.IO backend for transmitting real-time data.

June 29 - July 18

Implement the real-time visualizations using D3.

July 19 - July 29

Implement panning & zooming, and export options.

Mid Semester Evaluation

July 30 - August 10

Implement patient state and raw data display sections.

August 11 - August 31

Implement population views.

September 1 - September 16

Debug, complete documentation and write unit tests.

Background

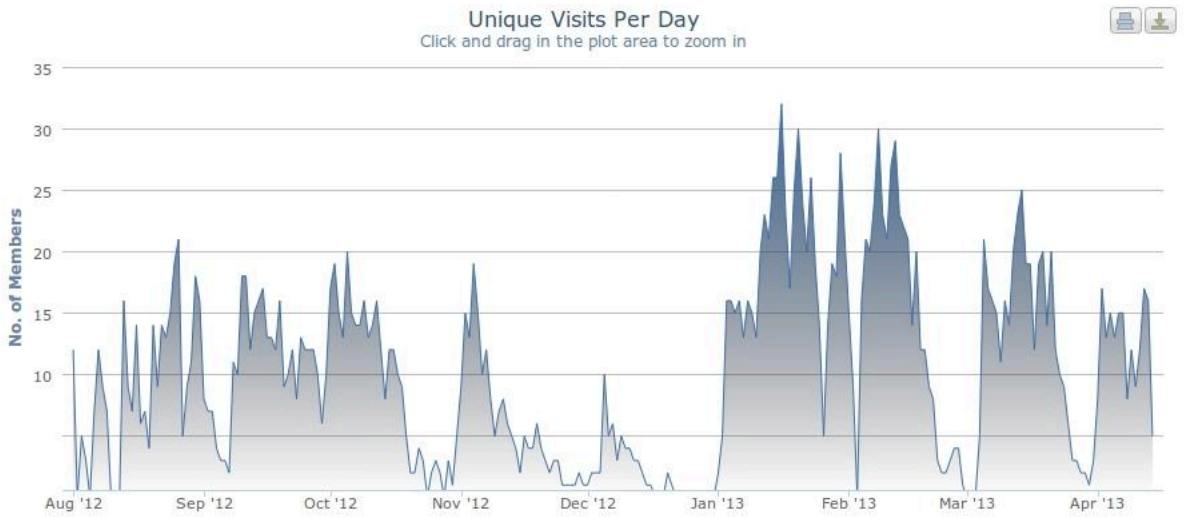
I'm a 2nd year undergraduate student at Indian Institute of Technology Roorkee, India. Web development is one of my primary hobbies and I'm an active coder for the Software Development Section of my campus. Recently, I won 3 prizes for the applications created over here at the annual exhibition. I also built the website for the annual technical festival of my institute, Cognizance: http://cognizance.org.in/

I have a strong experience in visual interface development using HTML5, CSS3, JavaScript and building real-time web applications using Node.JS, Socket.IO, WebRTC and visualizations using D3 and HighCharts.

RELEVANT PROJECTS

Presence

I built an application called Presence for keeping track of people present in my laboratory. It got data streams in JSON format from scans run over WiFi, Bluetooth and LAN. This data was plotted real-time using HighCharts, with zooming capability. Here is a screenshot:



HackView

I took part in a hackathon organized by Yahoo! at IIT Delhi. It was a 24-hour event. My team developed a real-time video conferencing application with collaborative document editing. It was built on Node.js and made use of WebRTC which is the latest in peer-to-peer communication technology. Collaborative editing was implemented using ShareJS. You can see the code here: https://github.com/abhshkdz/hackview

HackFlowy

This is my latest project. It is a real-time collaborative note-taking application built using Node.JS, Socket.IO for the backend and Backbone.js, jQuery, Lo-Dash for the front-end. Modernizr was used to to detect the extent of functionality a browser can give and accordingly write fallback functions. You can see the code here: https://github.com/abhshkdz/HackFlowy

MOTIVATION

I am highly passionate about software development and open-source. You can see my other open-source projects here: https://github.com/abhshkdz. I assure dedication of at least 40 hours per week to the work and that I do not have any other obligations during the period of the program, with the obvious exception of regular academics. A major part of this duration is summer holidays where I'll be working from my home. Also, if any part of

the proposal is not clear, I'll be very happy to clarify.