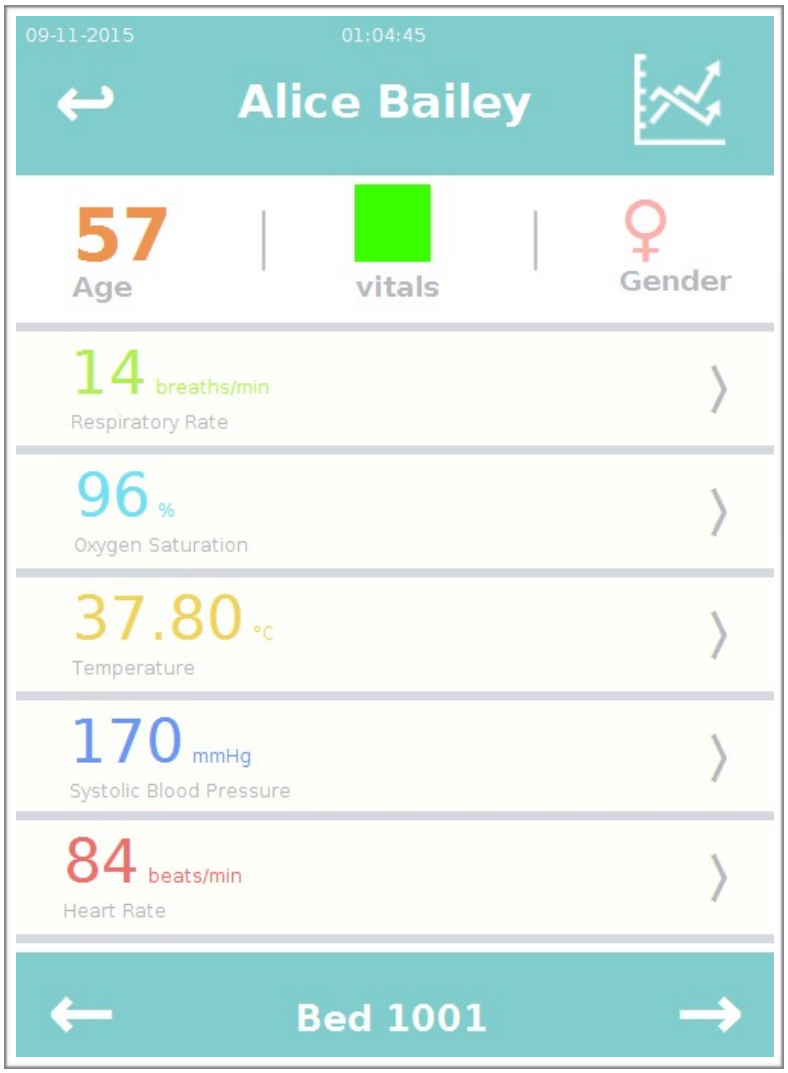
Human Computer Interaction

Digital Hospital Ward

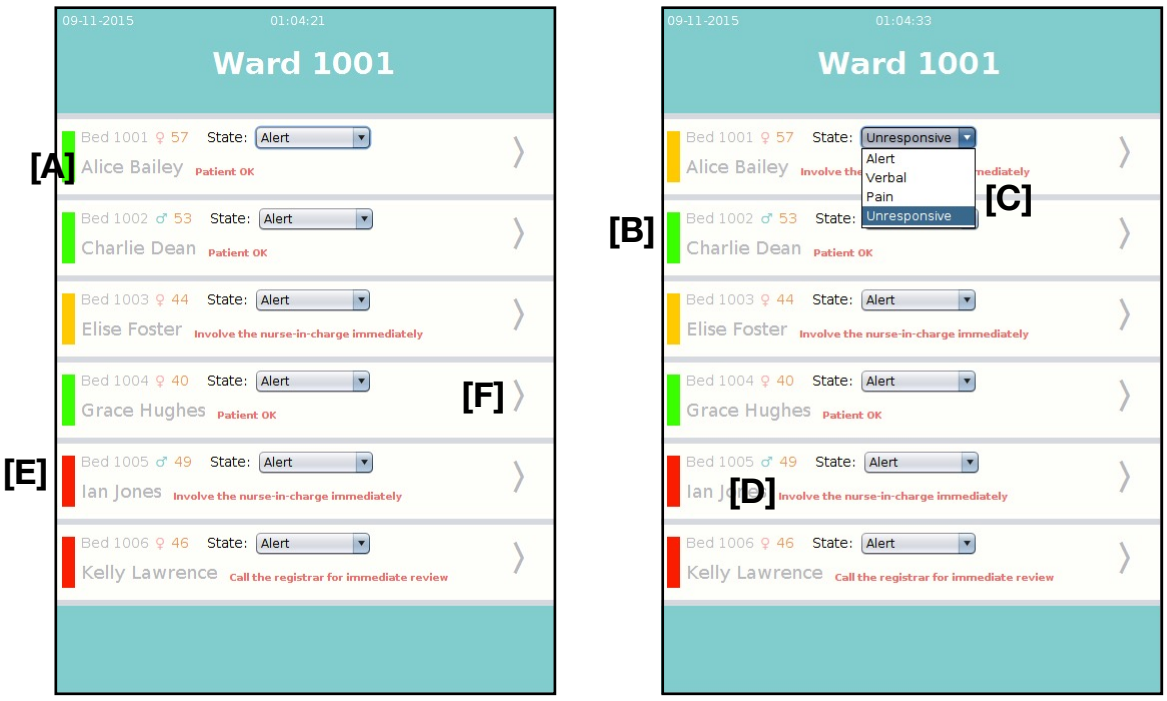


Michael Inglis

s1232123

November 2015

Ward View



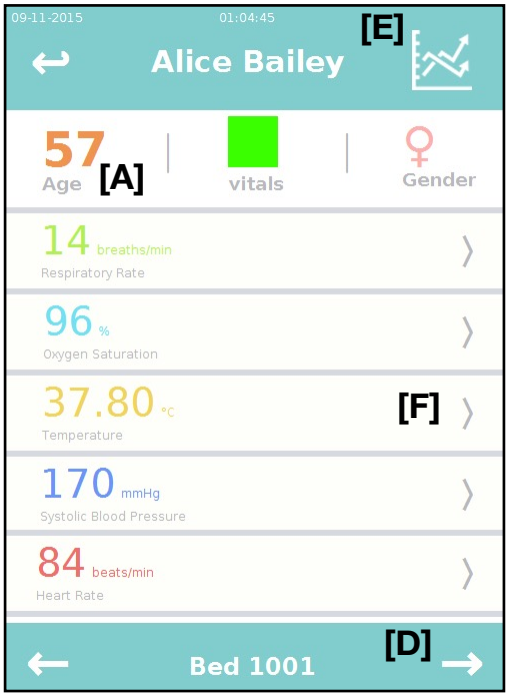
The first thing we notice is that the colour scheme and layout is the same. This preserves consistently within the app, allowing the user to know they are still in the correct application. We also have a similar design in both what information is displayed and the colours/fonts that are displayed. Information is sized based on what I considered most important. We see emphasis on the user's name **[A]** and pSEWS. The rest of the information; their bed number age and gender aren't hugely relevant to what this page is about, so are smaller. We want to be able to determine from here who needs assistance and then figure out what bed based on that.

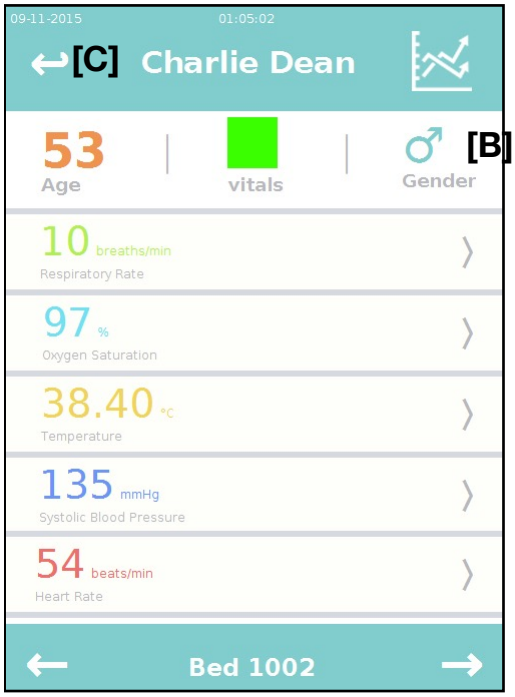
The pSEWS traffic light is arguably the most prominent feature when it is in action **[B]**. The colours do not need to be learned as each ones significance can be gauged instinctively (red == bad etc). It was my intention to have the "State" and dropdown menu **[C]** hidden if the pSEWS colour was green. I however was unable to hide the dropdown menu within Swing. Having this hidden would allow for a very quick look to figure out who needed a visual inspection. It does however correlate with the pSEWS colour, but the user would have to learn that.

The small text based on the SEWS score **[D]** is arguably the most important, yet is the smallest text, simply because of technical limitations within Swing. It is red to warn the user. If in a real app, it would post an overlaying notification to the screen, vibrate and make a noise, dynamically passing on the information.

Having each patient in a "module" **[E]** to ensure it is obvious what belongs to what. The Arrow on the right of the screen **[F]** also implies that this expands the information of the selected user, again keeping up with consistency, visibility, predicability, all while building on what the user has previously learned within other applications.

Patient View





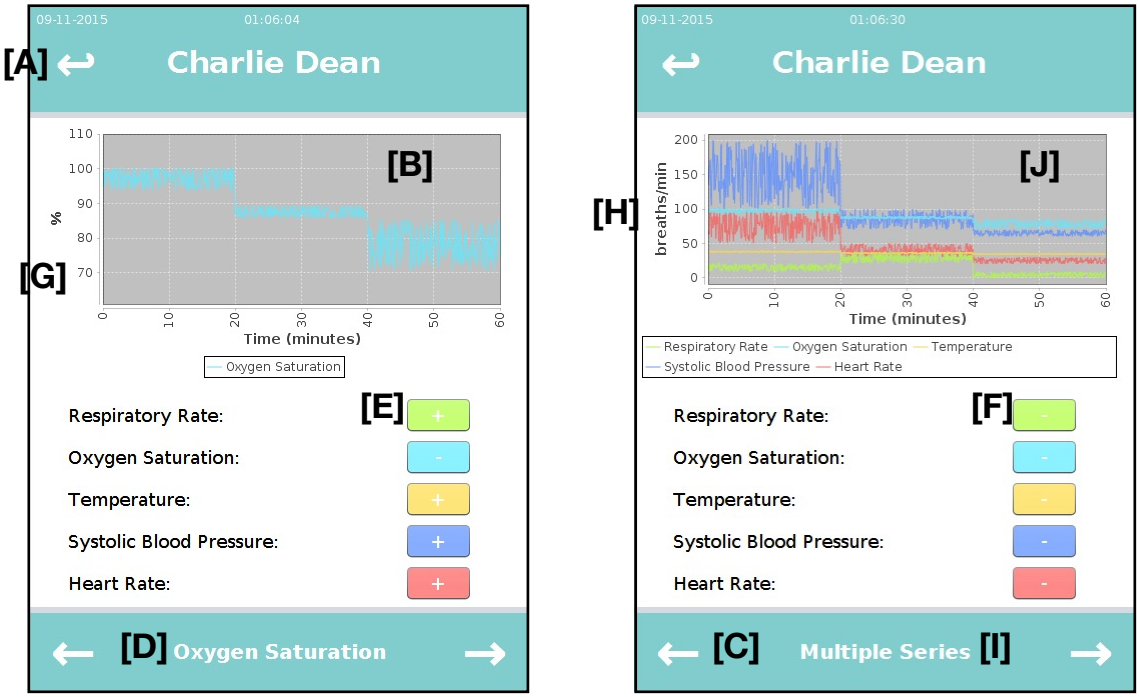
We can notice here that the data some of the data provided for each user has been removed or manipulated. The date of birth has been reduced to an age **[A]** and the gender to a coloured male/female symbol **[B]** and the temperature reduced to 2DP. This reduces clutter and emphasises the information which is important, somebody's birthday is irrelevant when they are ill in a bed. Having a colour associated with attributes (male/female with blue/pink), also reinforces consistency with the application as well as visibility for that attribute. The name and bed number have been enlarged to better enable nurses to distinguish between patients.

The Ward button has been replaced with a back arrow in the top left **[C]**, this is standardised practice for touchscreen applications and allows the user to instinctively reuse what they have learned from their previous interactions with other apps, reducing the learning curve when using this. The user here is using skills here that they have already learned.

The patients can be cycled through on a carousel. This is done by the right and left arrows at the bottom **[D]**. having these here with the bed number in-between imply that it is iterating through the bed numbers. There is feedback in that the name and information changes.

I will explain what the graph icon in the top left **[E]** and the arrows **[F]** on each medical parameter do in the History View justification. Their design here however is fairly self evident. The graph icon tells the user what it goes to - a graph. The arrows quite obviously expand upon the parameter. The fact that these are clickable is obvious to the user as they visually invite the user to press them. Having these stops the user having to search for functionality within the app as their options can be seen.

History View



Consistency here is again shown with the interface. The same colours for medical parameters and overlay are evident. The back button **[A]** also features, continuing with what the user has previously learned. This view is for the user to visualise what the parameter data A) looks like and B) how it correlates with other parameter data.

The graph view is reached from the graph image and expanding each medical parameter on the patient page or pressing the graph icon (which activates all graphs) **[J]**. When expanding parameters, that individual parameter is displayed solely on the graph **[B]**. The user may cycle through parameters using the white arrows below **[C]**. The name of the data **[D]** and colour of the graph change depending on the parameter.

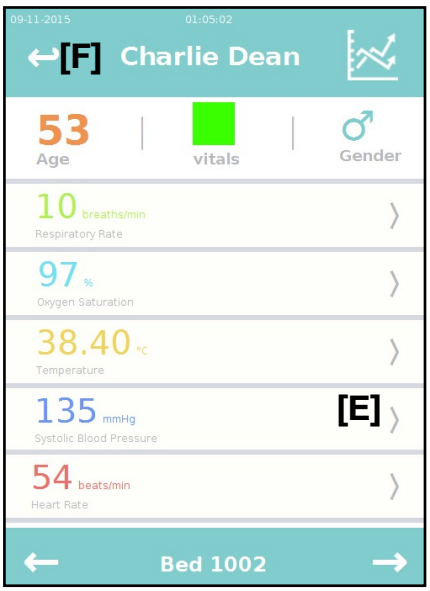
The user may choose to add/remove parameters on the graph. This is done by clicking one of the colour coded "+" buttons **[E]** (colour again consistent with each parameter). Once clicked, feedback is given with the graph by the graph appearing and the sign on the button changing to a "-" **[F]**. Clicking a "-" button will result in the graph being removed and the button changing back to a "+". These actions are predictable based on what we see the on the UI.

On the graphs themselves, the x-axis has been set to range ±10 of the max and min of each parameter to enhance readability **[G]**. The seconds also have been converted to minutes based on how much more interpretable that is.

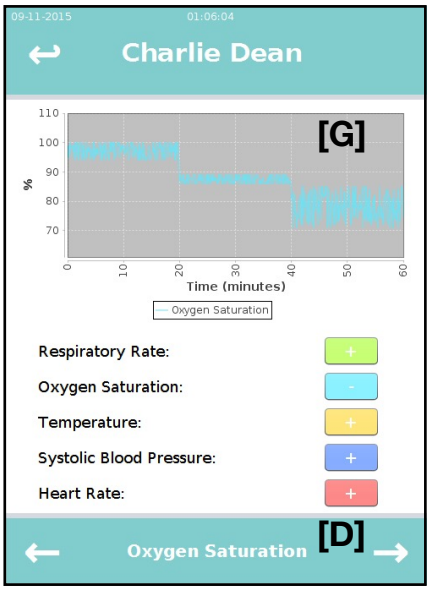
NOTE: When writing up, I've noticed my X-axis title doesn't change when >1 parameters are added **[H]**. This should utilise a similar algorithm as the graph title which changes to "Multiple Series" **[I]** when >1 parameters are selected. This would have to be fixed to fit in with the consistency and predictability of the app.

Evaluation

[A]



[B]



[C]

I went for a hierarchal approach for this UI. The ward view **[A]** would be where the application opens. Here, there is patients to chose from which would take us to the patient view **[B]** and finally, each patient has parameters which can be shown as graphs on the history view **[C]**. Each screen can only be accessed from the one before it. This gives us a clear path through the app, enhancing learnability and predictability; the user isn't going to find themselves in an area they don't know how to get out of.

The design of the application remains consistent throughout the entire app. This is a flat design with a green header and footer, with white buttons in the corners. The green was used because of it's association with hospitals and cleanliness. Keeping this throughout the app, visually reassures the user where they are on their device.

The arrows on the footer **[D]** always cycle through what was seen on the previous page; on patient view, they cycle through the patients seen on the ward page and on the graph page, they cycle through the parameters seen on the patient page. The ">" arrows for selecting a patient **[E]** and an attribute also remain consistent and give predictable interactions. The back arrow **[F]** was chosen over a "back" button simply because it was more in keeping with the flat design. It again was used throughout and contributed to consistency, visibility, learnability, feedback and predictability by maintaining the hierarchy of the application.

I chose to exclude the "exit" button as all touchscreen devices have some sort of hardware button which takes them out the app. This builds upon what they have previously learned using their device. I also moved around the date and time, splitting them up and standardising the order, making more easily interpretable.

We see a lot of abstraction of data throughout the app; the data for each parameter over the hour **[G]** is viewed as a graph which is far more visually appealing and understandable than a list of 3600 integers. We do this because we consider the potential users, most likely nurses, but may be used to show patients their vitals (though not handed to the patient due to other's private information). Doctors may use this also, though, while

useful, one would assume they could easily interpret the raw data we show in the graph and they are only going to come if summoned by a nurse due to the app's given status of the patient.

Stakeholders in this app would include the hospitals board and whatever governs that (a trust or a government). It is vital it works as intended as any failings ultimately risk lives and could result in death or serious injury.

Unfortunately, the application is far more passive than I would intend it to be as regards to feedback. It doesn't make any noises as I didn't have time to work out how to do so on Swing. Ideally, the patient page shouldn't have to be opened, just running in the background until it alerts the nurse, noisily and visually about a patients score.

Notes on the Code:
I expanded the Patient class significantly and added an external "wardPatient" JPanel, so I didn't have to do a lot of dragging and dropping on the WardJFrame. I will include both of these in the top level of the submission. There is also an image I imported into my project. I have included the whole project in a zip file within my submission. You may wish to look at them if you are intending on running it.