Notes for report

Intro

Plan was to build a website that could provide a central hub for lots of different types of remote data monitoring systems

Design goals

Using java to build a relational database to efficiently store the data, and allow it to be reconstituted

Using jsf to dynamicly display that data

Provide facility for users to upload data to the website using a selection of remote methods and allow users to manage a variety of data types and sources

Display the monitoring points using google maps, with the ability to select different data variables

Use a chart to display the selected data on the page

Support mobile and static data sources

Use Test Driven Development (huw’s request about development priority)

Support the uploading of calibration information to the database, that will adjust long duration instrument results automatically.

Allow a data dump service (CSV).

Status

Currently the system can set up multiple systems, but store data for only one system, display the stations on the map, and display some (fixed, representative) data on the graph.

Problems encountered and resolved with reasons

Problem with google maps setup, sensor=SET\_TO\_TRUE\_OR\_FALSE" caused an unrecognised API key error, line removed, but not sure of implications

Removed the ManyToMany relational map between Node and NodeVariable tables. was deemed unnecessary, as OneToMany from each object to a central object simulates the effect.

TDD dropped due to timescale (already learning Javascript, Jquery CSS etc. so deemed too much)

Issues with making changes to the database (after being built) and then getting persistence errors (solution was to rebuild using a fresh persistence uinit to ensure that the database matched the entity framework

Issue with JPA scope, when iterating over the CSV, the temporary array was being initialised each time, so only storing the last variable set, fixed by moving initialisation outside of the loop

Error with the autogenerated primary key not being recognised as correct. Reloading the IDE and rebuilding the program several times resolved the problem, reason unknown.

EJB error caused on adding nodes to the databse (autogenerated code) error in comparing strings, tracked town to a badly constrained for loop, going over the end of the array. Did not present as Array out of bounds exception.

Learing curve on javascript. Passing data from JSF to javascript and the correct way to so, #{bean.methodcall(withvariables)} remember, can return a variable, an object or an array

Had to learn how to work around the asynchronous nature of jsf to javascript. Can’t pass data back and forth without a reload. Loading all of required variables on load is one possibility, the other is learning AJAX

“big problem sending variables between JSF and Java, can’t send back and forth, as jsf runs constantly, jvascript runs once, soliution was to put all calculation output into a persisted variable, minimise calculation done in javascript”

Database was truncating the GPS Coords, meaning that all of the points were on a single point . The problem was a scale set to 0, resulting a decimal with 0dp, previously stored as a float, the entity was changed to a BigDecimal with scope of sufficient size to support a decimal gps.

Making the popup uses the infoWindow in google api, initially buttions created using divs and formatting, switched to a procedurally generated dropdown menu, but onChange action listener not functioning.

Initially used canvas to generate a graph, but was advised by andy starr that google charts wouyld be more efficient. Can pass an array directly to the function, but the function that should do the same is currently not working due to passing a string, not an array.

Areas of planned improvemnts

Sockets

Data dump

Security

Areas that could be improved, benefit, reason, and why not

Multithreading

Performance imporvements

Timestamp – should it be first? Currently assumed index 0 – if that’s not the case then need to add variables to db to indicate where the timestamp is, then start changing code to make callbacks and work out where that is from database before you start to interrogate the data in the chart object