Software Programming Assignment #1
Due: Feb 12, 2014 @ 6PM

Note: This homework is to be done in teams of two. You would need to submit all your code, a report describing the design, and link to a video or screencast demonstrating the capabilities of your system. I may also request an in-person demonstration. Submission link will be provided later - watch this space.

The objective of this homework is to explore organizing embedded software using mbed as the platform in context of a simple system. The homework only specifies the high level functionality, and you are responsible for fleshing out the details. Your specific goal is to make the mbed into a configurable internet-connected data collection system - an end node in an IoT systems.

Your mbed has several sensors: 3-axis accelerometer, magnetometer, light sensor, and touch slider. Additionally, it has a 6-channel ADC to which up to six external analog sensors can be connected. Your system has to sample these sensors and upload the time series to the Xively service (http://xively.com) in real time. The data from different sensors of an mbed should go a different channels of a single Xively feed. Since mbed does not have a direct network connection, the transport will happen via a host to which mbed is connected over USB. Your system would need a helper program running on the host to acquire the data from mbed via serial-over-USB, and upload to Xively. You can write the helper program in various languages and the mbed cookbook describes how you can interface to mbed from a host over USB in various languages.

You need to provide a graphical or interactive shell type user interface via which a user can configure and dynamically control your system. At the minimum, the following elements should be configurable:

a. Information about Xively feed, channels, and key.
b. Which sensors on mbed to currently sample (this should be dynamically configurable)
c. Sampling rate for the sensors (this should be dynamically configurable, and settable individually for each sensor)

Since Xively imposes rate limits on how often you can calls its API, you may need to aggregate multiple samples in each API call to XIvely. Moreover, this rate limit together with mbed's limitations would limit the sampling rate.

To test the functionality of sampling the ADC, use the second mbed in your team to write a program that generates a sine wave with configurable frequency and amplitude on the analog output pin (DAC), and which you connect to one of the analog in pins (ADC) of the first mbed.

Report the highest sampling rate you are able to sustain for (i) only one sensor being sampled, (ii) all sensors being sampled.

You may use the mbed RTOS if you like, or organize the software in the form of event handlers in a foreground/background structure.