CSCI 188 Project Proposal

RIM BlackBerry-based Robotics Controller

October 5, 2009
Frank Ervin

Project Abstract

This application will explore the multimedia, web services, and perhaps the accelerometer capabilities of the BlackBerry in order to develop the mobile underpinnings for a lightweight robotics control application. The application will be called CamBot.

Strategy

I will write this application for the RIM BlackBerry Storm using either the RIM JDK or the RIM BlackBerry Plugin for Eclipse. The libraries I intend to use are:

javax.microedition.xml.rpc (for consuming a quasi-functional controller web service)
net.rim.plazmic.mediaengine (for the viewing of video streamed from r/c device)
net.rim.device.api.system.AccelerometerSensor (time permitting, for raw sensor reading using the RIM system api)

Unknowns & Problems

I am not yet particularly familiar with any portions of this design. I am anticipating issues with BlackBerry-specific variables (such as code-signing and Mobile Data Service / IDE configuration). I am unsure of the latencies to expect over Verizon's carrier network, but imagine it could be significant. I also anticipate that it will be challenging to overcome issues with multithreaded performance while the device is rendering video, listening for commands, and sending web service calls. I have read about web service performance optimization in "real-world" blackberry apps that use manually created web requests using raw XML strings for optimal performance. Minimally, I will likely need to look at RESTful web services for most of the commands being sent on a timer interval. Since this is application is being written for a J2ME class, I will be focusing on the elements of the design specific to the handheld. An option for simplicity might be to integrate with a MS Robotics Studio simulation.

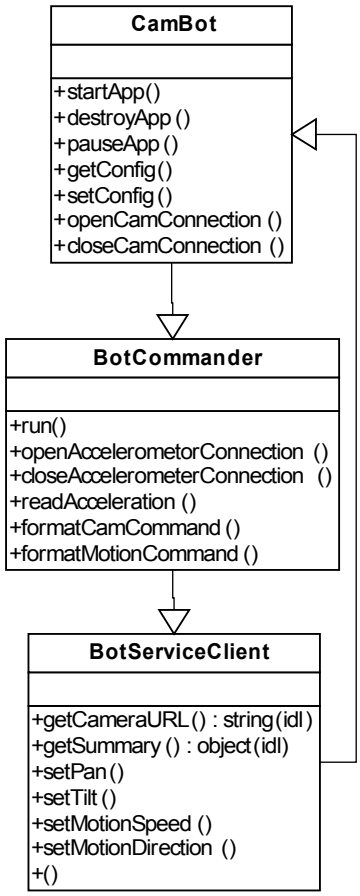
Implementation Plan

Task	Time Estimate	Completion Date
Configure RIM-specific simulator, IDE, and MDS	2 weeks	
Develop quasi-functional control web service (to be completed time-permitting or in the	3 days	

future)		
Deploy a suitable video stream for the project	2 days	
Implement mechanism for configuring and persisting application settings	4 days	
Create a canvas class with an embedded multimedia viewer	4 days	
Develop command listener (for conveying commands and keystrokes to web service)	4 days	
Develop a web service class for talking with the control web service that is thread safe	2 weeks	
Develop a status panel for reading and displaying robot status (battery, GPS, etc)	4 days	
Develop class for reading accelerometer data for use in addition to keystroke control	2 weeks	

As discussed with the professor, I need to be aware of time drain caused by non-J2ME design intricacies and minimize these throughout the projects. These issues can be resolved at a later date. I plan on doing most of my initial development on simulators and move these to physical devices as the project matures. Since I will be using MDS as the channel to access my web service, I hope to not have to worry about security concerns associated with deploying web services and video out to the Internet.

At a very high level, the application could be described with the UML that follows. The end result is very likely to contain an additional class for managing robot state, but this diagram minimally articulates the distinct threads will need to be managed to have a responsive application.



Below is a mock-up of a possible UI for my robot controller: