On Demand Adaptive Computing With Mobile Devices

Agustin Rivera, Thomas Langford
Mentor: Dr. Qijun Gu, Dr. Mina Guirguis
Department of Computer Science
Texas State University – San Marcos

Introduction

On-demand adaptive computing stands on the principles that machines or devices might not always have every resource available to complete a desired task, and that for unexpected events or scenarios, it may not be know what devices and capabilities are available in the field and thus cannot deploy applications in mobile devices in advance. Therefore, instead of limiting the tasks that could be completed to the capabilities of a single device, a general framework would allow for different devices to be programmed according to their capabilities.

Objective: This research was conducted to:
- Create a general framework for on-demand computing that will automatically generate a program based on a device's capabilities without need for direct human input

Potential Applications:
- eHealth
- Military applications
- Environmental disasters

Example:
In natural disasters situations, mobile devices can collectively perform search and rescue operations. One major challenge for these systems is to harness the available resources (e.g., other robots, sensors, cell phones, cameras, etc...) in the surrounding environment to accomplish their assigned tasks. This necessitates that the mobile devices be aware of other nearby devices, and know their capabilities and how to utilize them. To that end, this project develops an on-demand computing platform in which certain tasks are achieved by partitioning applications into portions and deploying those portions on various devices based on their capabilities.

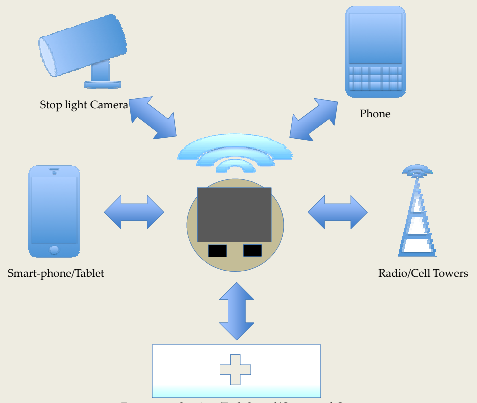

Figure 1. The mobile device carrying the general framework will harness the capabilities of other mobile devices to accomplish a task and communicate to command center.

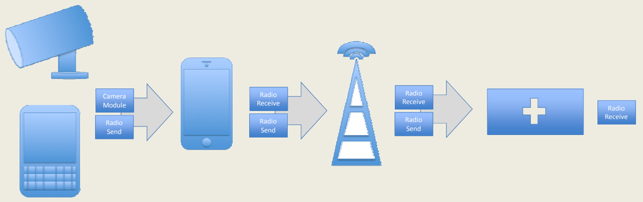

Figure 2. Possible Workflow depiction of the on demand applications of the partitioned task. Devices (left) are assigned modules (right) based on capabilities.

Implementation

Experimental Setup:
- Dell Inspiron
- MicaZ motes
- iRobot Create

1. iRobot Create searches area for MicaZ motes using Java listening program
2. Once within range of a mote the home-base parses the data received and determines the current version of software and ID on the mote.
3. Based on version and ID an application is assigned.

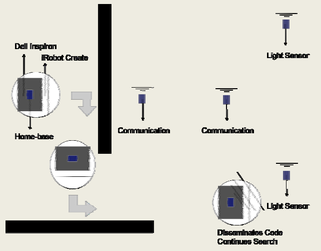

Figure 3. iRobots navigate through obstacles to find motes and assign them tasks.

Framework

On-demand computing depends upon the ability to split up a task into several pieces that can be distributed to other systems to be completed. Our framework enables one device to generate specific programs that is then distributed to another device, such as a sensor. This framework is built on modules and templates. Templates provide the most basic functionality such as ability to boot on the sensor. They are able to compile by themselves with no modules added. Modules provide more advanced functionality such as finding an average of a dataset or sending a packet over radio waves. Modules are plugged into a template to create a finished application.

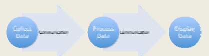

Figure 4. A general representation of a possible application workflow.

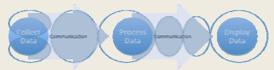

Figure 5. Application workflow split up into different possible applications. Each circle represents an application.

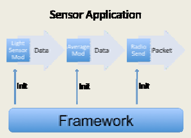

Figure 6. Workflow of a possible sensor application. Three modules (light sensor module, average module, radio send module) are plugged into the framework to build this application.

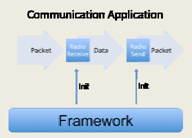

Figure 7. Possible communication application built with the on-demand adaptive computing framework.

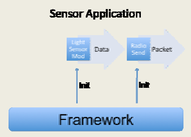

Figure 8. Another possible sensor application. Shows the modularity of the modules by having the ability to remove the Average Module and place it in the communication application.

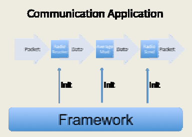

Figure 9. Another possible communication application. Shows the modularity of the modules by inserting the average module from the sensor application.

Results

- The robot can keep track of distances traveled and degrees turned, to be later used to record areas covered.
- The robot can search for and find a mote. It then disseminates code to the mote with the assumption that the mote's capabilities are already known to the robot (i.e. the robot has a database of the ID's and disseminates a predetermined application).
- Netbook can queue motes if many are in the vicinity by checking queued motes. It then uses a version number to determine if the mote has already been reprogrammed, if not it will queue the mote.
- The framework creates modules by taking a template file and filling in xml tags with the information provided in the module list. Then it creates a working application to be disseminated to the mote.

Further Work

Further Work:
- The framework expects the modules to be handed to the application builder in the correct order. If they are out of order, then it is unlikely that the finished application will compile for the motes.

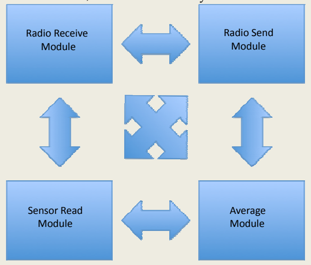

Figure 11. Ideally the options for creating an application would create a multiple path graph, eliminating the need for modules to be handed in a specific order.

- Incorporate the framework into the robot; as of now the robot is disseminating a prewritten application.
- Include a method to keep track of where the robot has already been to increase efficiency of the framework (save time and battery life of netbook, motes, iRobot).
- Increasing the amount of mobile devices using the framework (e.g. multiple robots) and have them communicate areas already covered.

Acknowledgements

This research is co-funded by the Department of Defense in partnership with the NSF REU program and HLSAMP.