Develop a lightweight communication protocol for constrained devices in a closed loop system

Fergus W. Leahy

December 13, 2012

1 Project description

In the past few years more and more cheap, open source, constrained devices have hit the market. Examples of such are micro-controllers like the Arduino(+derivatives)[2] and ARM core pc's like the RaspberryPi[1]. These devices have had a significant impact on the *Internet of things* paradigm, due to their low cost, small size and ability to easily integrate with many types of sensors and actuators.

Similarly many new startups and companies have tried to take advantage of these new devices and remodelled them into their own Internet connected thing.[4][5]

The current problem with all these new devices is that whilst they provide powerful tools to create a Internet connected "thing" there isn't much support or many suitable tools to help connect these "things" together in a closed loop system i.e. automation. By connecting devices together in this way much more powerful systems emerge with far richer capabilities.

There are some open source existing solutions for connecting devices together in this manner such as the Java Messaging Service (JMS) or eXtensible Automation Protocol (xAP)[3].

But when taking into account the constrained limitations of devices such as the Arduino, which has only 16MHz processor, 32kb ROM and 2kb ram, solutions like the heavyweight JMS become impossible to implement. Attempts have been made by the open source community to create a suitable protocol/system for constrained devices such as xAP, but it's not without it's limitations.

One of the design principles for xAP was that for it to create a fault tolerant distributed network, this was implemented by the system using only broadcasting when communicating between devices. This would allow the system to not need a central controller/name server to facilitate communications. Whilst trying to alleviate one problem the protocol introduces another much larger one, scalability. As more and more devices are added to the network, these devices then flood the network with unnecessary data regardless of whether any other device wants to listen or not.

This project intends to learn from these problems and develop a new communication protocol to facilitate communication between constrained devices in this manner, without the issues of scalability and efficiency which these other protocols/systems present.

The design of the protocol will aim to be system/network agnostic but for this project it will be implemented the TelosB motes due to their availability within the school.

2 Progress

So far the project has come from just an idea which was then fleshed out with use cases and scenarios to become a full specification. The design of the protocol has been heavily worked upon, taking lessons learned from previous attempts such as xAP and combining it with other technologies such as RPC. From looking at these and working through the needed semantics of the use cases we've designed a protocol with a simple means of creating and managing a network of these *Internet of things* devices whilst still providing flexibility, scalability, efficiency and resilience.

After designing message diagrams and state machines to show how the protocol would facilitate message passing the implementation for the TelosB motes begun using the Contiki operating system

for wireless sensor nodes. The protocol is to be written in a variant of C developed for the Contiki OS which combines an event driven system with proto-threads.

So far the basic mechanisms for device discovery on the network have been created initially in C for Linux, but were then ported over to Contiki.

3 Plan

The plan for the rest of the duration of the project is twofold, which is intended to be done side by side. The first is to further develop the system, implementing the necessary functionality. Secondly the project report will need to written alongside the development in order to provide enough time and to ensure the necessary detail is included as work is done. After the implementation is complete, test cases will be developed to test the performance of the protocol using metrics such as network bandwidth and scalability.

4 Problems

So far there have been various problems which have been encountered and overcome.

Problems such as the initial development platforms for the Arduino device not easily supporting the needed functionality to create an efficient implementation(no network interrupts) which forced the implementation to be done on the TelosB motes instead.

Following on from that a simple one such as the lack of knowledge and expertise in developing on an event based system introduced a significant learning curve to be able to develop for the intended platform.

Another such problem was the difficulty in finding appropriate documentation and support for the Contiki OS and its development environment which initially made developing for the platform slow. It also caused problems later on when learning how to use its IPv4 stack and trying to understand the functionality that it provided.

One piece of functionality which was mentioned within the ill-maintained documentation is the availability of broadcasting UDP packets. Which, although is mentioned and fragments of relevant code exist within the stack, the examples given simply show that it doesn't in-fact work. After asking around the community of developers, few replied with useful information, so a fix was self-implemented to create the necessary functionality.

From now onwards problems such as good time management between the implementation and report writing are bound to arise due to the further workload experienced from other subjects in the next semester. Further problems with the lack of thorough documentation could prove to have a significant impact on the time scale of the project.

References

[1] RaspberryPi, http://www.raspberrypi.org/

[2] Arduino Platform, http://www.arduino.cc/

[3] xAP, http://www.xapautomation.org/

[4] Smartthings, http://smartthings.com/

[5] Twine internet thing, http://supermechanical.com