Innovating in IoT Using Eclipse

Experience and Recommendations

Max Domeika
Intel Corporation
Innovation Architect
March 10, 2015
Purpose & Agenda

Purpose
• Share what’s needed in IoT to move the industry forward
• Experiences and recommendations on Eclipse*

Agenda
• The Democratization of IoT and the Developer
• Features Needed to Assist the Developer

*Other names and brands may be claimed as the property of others.
Your Speaker: Max Domeika

IoT Innovation and Productization

Smart Helmet – Amazon re:Invent
Internet of Things
A Battle is Brewing

Traditional Embedded
Democratication

Monitoring
Lifestyle

Manufacturing
Connected Home
Who are these Democratized IoT Developers?

Intel® Internet of Things Developer Program

A comprehensive developer program for hobbyists, students and entrepreneurial developers with outreach, training and tools required to rapidly develop, test and deploy applications for the Internet of Things (IoT).

Program includes:
- Developer kit based on Intel® Galileo and Edison Technology with package of easy to use hardware, software tools, libraries and cloud services
- 10 City IoT Roadshow distributing 1,000 kits
- On-line community for learning, building, sharing

Join the community today at software.intel.com/IoT

HTML5 and JavaScript Developers

- Intel® XDK integrated development environment
- Build NodeJS applications
- Leverage open source project.s
- Debug remotely.

Entrepreneurial Professionals

- Eclipse* C++ IDE.
- Deploy to Intel® Galileo and Intel® Edison.
- Debug projects remotely.

Students and Hobbyists

- Software- and hardware-compatible with Arduino.
- Intel® donated 50,000 Arduino compatible development boards featuring Intel® architecture to 1,000 universities around the world over 18 months.

"If we are not a part of that [The maker movement], we are going to miss the future of where computing is going."
- Intel® CEO Brian Krzanich
  At Maker Faire Rome
What are These Developers’ Problems?

- Board Support
- Startup Experience
- Security for Activation
- E2E Development
- Power Analysis
- Sensor Programming
- Domain Specific Functionality
- 3rd Party Frameworks Integration
- PCB Creation
- Custom OS Creation

Concerned with customized SW and customized HW!
Maturity level of tools is much less than app developer tools!

Need more powerful developers or more powerful abstractions!

Hardware
IDE
PCB Process
3rd Party
APIs/
Frameworks

Security
Cloud
Power
Sensor HW
Software
Libs
Two Choices

Simplify.

- Arduino* IDE
  - Easy to get started
  - Easy to use
  - Simplified, but capable

Win them over.

- Eclipse IDE
  - Powerful
  - Feature-laden
  - Extensible

*Other names and brands may be claimed as the property of others.
Why Not Both?
Provide a Path from Maker to Pro

Improvement Areas

• Startup Experience
• Sensors
• Connectivity
• Edge to Cloud Development
• Customized Hardware
• Productization
Startup Experience
First Impressions Matter

Ease the Install

- Show and tell
- Downloads automatically
- Choice for a range of developers types

Offer Choice - Minimize download

- Choice, but not too much choice
- Self-contained
- Browser-based IDE, how novel

Intel® IoT Devkit Install

BeagleBone* Black Setup

*Other names and brands may be claimed as the property of others.
Connectivity

Eclipse Remoting

• Powerful, but…
• Overwhelming to the uninitiated

Arduino Remoting

• Still Difficult – USB/Serial comms
• Easier - LightBlue* Bean wireless & multiple connections

*Other names and brands may be claimed as the property of others.
Sensor Tools

• Provide Building blocks
  – Libmraa
    https://github.com/intel-iot-devkit/mraa
  – UPM
    https://github.com/intel-iot-devkit/upm

• Use Software Emulation for sensors and board (concept)
  – Sensor sends ID and info to ‘aware’ IDE
  – Execute code before hooking up physical sensors
  – Can perform verification

Temperature Sensor – Analog Input

```c
#include <unistd.h>
#include <iostream>
#include "grove.h"

int main(int argc, char **argv)
{
  // Interesting
  upm::GroveTemp* s = new upm::GroveTemp(0);
  std::cout << s->name() << std::endl;
  for (int i=0; i < 10; i++) {
    std::cout << s->value() << std::endl;
    sleep(1);
  }
  // Interesting
  return 0;
}
```

UPM Sample Code

Sensor Emulation

Intel Galileo
Ethernet: Connected  IP Addr: 127.0.0.1  CPU Usage: 12%
Memory Usage: 155MB Debug Port: Connected
The Action is in the Cloud

Electric Imp*
- Register Using BlinkUp
- Browser-based IDE for Agent (cloud) & Device (client)
- DevOps console – maintain and push your software

Wyliodrin*
- Browser-based IDE & remote programming (client only)
- Basic DevOps Console
- Visual Programming

*Other names and brands may be claimed as the property of others.
IoT is Customized SW and HW

Help with both

Circuits.IO*

- Intel® Edison Breakout
- Execute sketch on simulated board layout
- Design a custom PCB

*Other names and brands may be claimed as the property of others.
Power Optimization

The Device is Off Most of the Time

- Today’s State of the Art is a manual process
- Power monitoring tools provide the feedback needed.

SoCWatch Output in Vtune™ Performance Analyzer
Summary & Call To Action

Problem:

IoT Development is at least an order of magnitude more difficult than software app development.

Therefore:

IoT will either need more developers or more powerful abstractions.

Opportunity:

Eclipse can provide what developers need

- Startup Experience
- Sensors
- Connectivity
- Edge to Cloud Development
- Customized Hardware
- Productization