Google Summer Code 11

Levee Health Monitoring and Surveillance

Milestone 1.0 alpha - Proof Of Concept (POC)

My GSoC background

- GSoC Project types:
  Patch, Bug fixes, Add-on, Hack,
  Community Todo list, Vanilla code base,
  Proof of concept etc

- GSoC 09 - ASCEND at CMU
  Add-on!

- GSoC 11 - Fibercorps
  Proof of Concept (POC)

Levee Health Monitoring and Surveillance

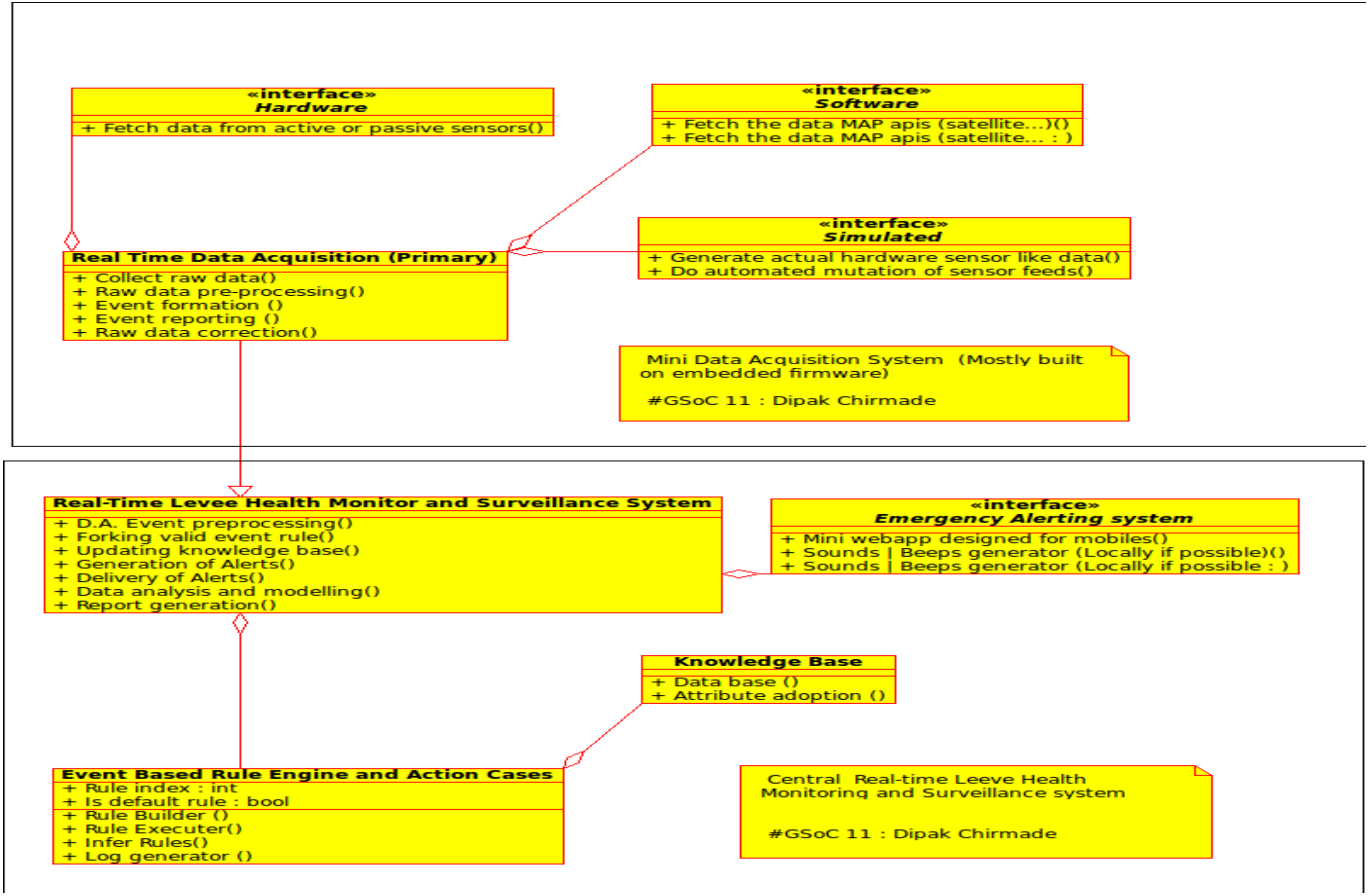
The awesome part:

- No previous source code
- No previous hardware and software involvement
- Real world problem (unlike GSoC 09)
- Just not all about hardware sensors

Why part:

- C, C++ at core. Wny not Java or so!
- Is it a firmware or an embedded or standalone i386+?
- OpenWrt to X86 coverage
- Agile approach which sprint of 1 week!

Proof of concept (POC)



«interface»
Hardware

+ Fetch data from active or passive sensors()

«interface»
Software

+ Fetch the data MAP apis (satellite...)()
+ Fetch the data MAP apis (satellite... :)

«interface»
Simulated

+ Generate actual hardware sensor like data()
+ Do automated mutation of sensor feeds()

Real Time Data Acquisition (Primary)

+ Collect raw data()
+ Raw data pre-processing()
+ Event formation ()
+ Event reporting ()
+ Raw data correction()

Mini Data Acquisition System (Mostly built on embedded firmware)

#GSoC 11 : Dipak Chirmade

Real-Time Levee Health Monitor and Surveillance System

+ D.A. Event preprocessing()
+ Forking valid event rule()
+ Updating knowledge base()
+ Generation of Alerts()
+ Delivery of Alerts()
+ Data analysis and modelling()
+ Report generation()

«interface»
Emergency Alerting system

+ Mini webapp designed for mobiles()
+ Sounds | Beeps generator (Locally if possible)()
+ Sounds | Beeps generator (Locally if possible :)

Knowledge Base

+ Data base ()
+ Attribute adoption ()

Event Based Rule Engine and Action Cases

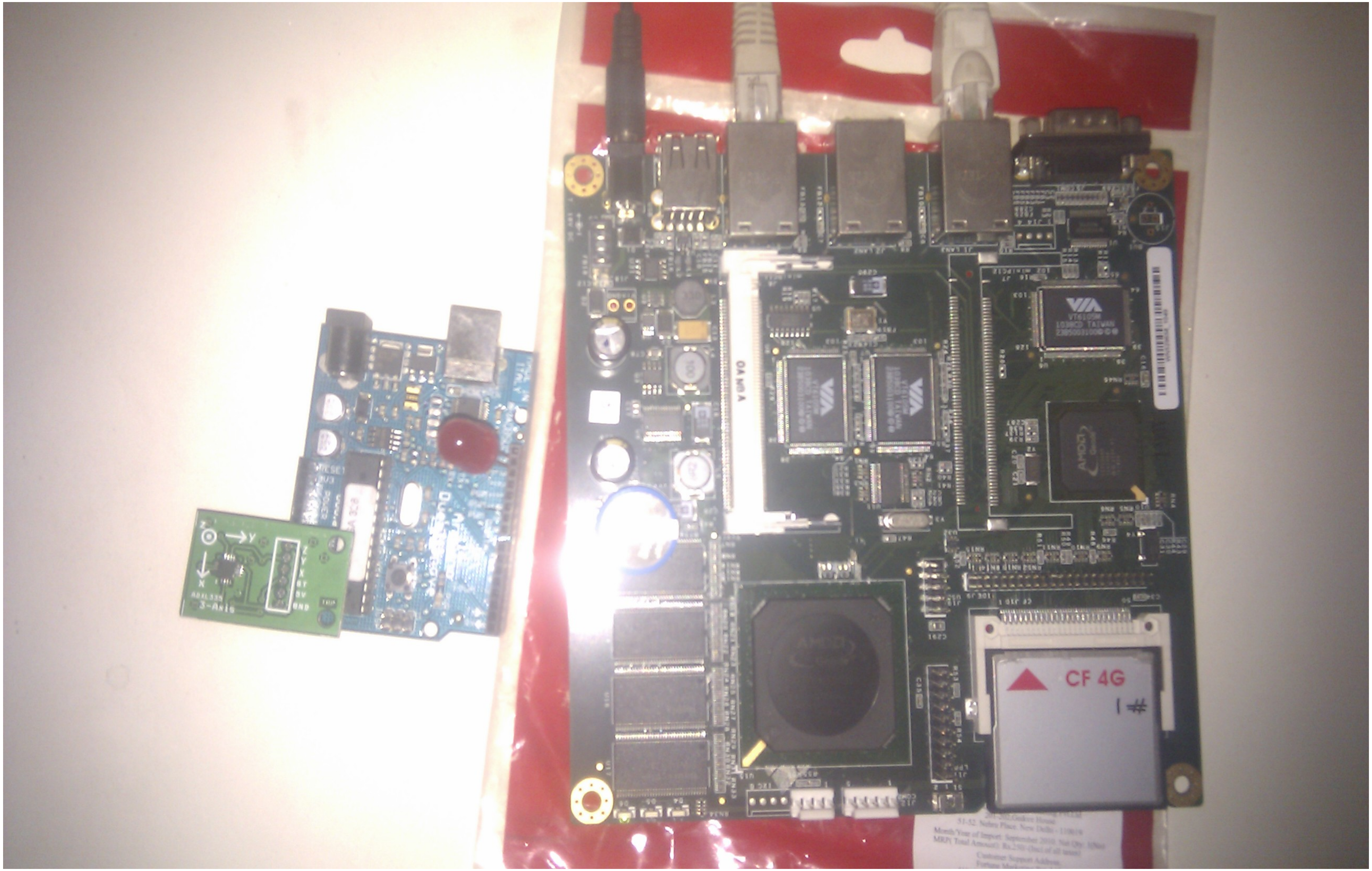
+ Rule index : int
+ Is default rule : bool

+ Rule Builder ()
+ Rule Executer()
+ Infer Rules()
+ Log generator ()

Central Real-time Leeve Health Monitoring and Surveillance system

#GSoC 11 : Dipak Chirmade

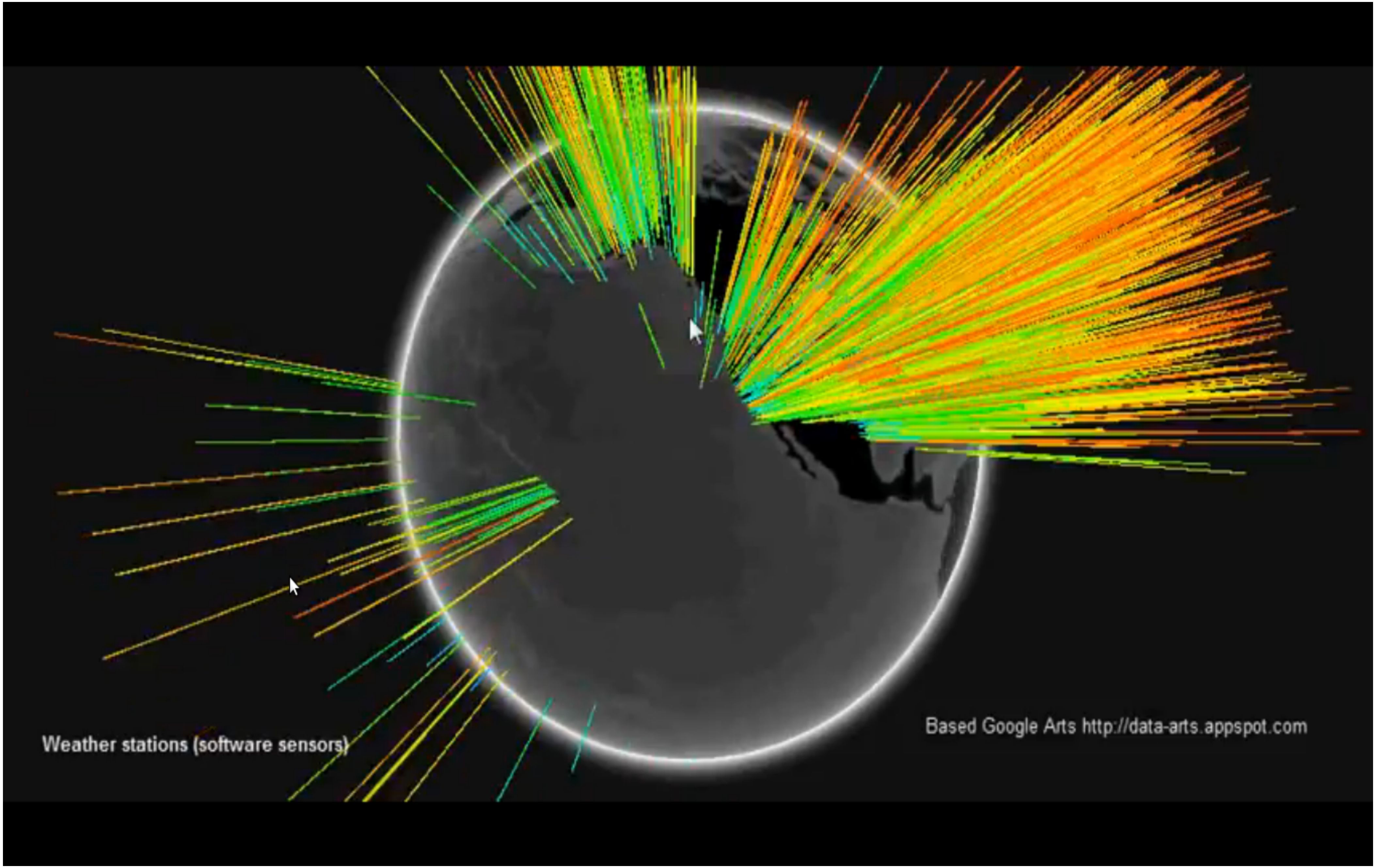
Hardware sensor | Arduino, Google's smart home core!



Hardware sensor's event

```
/**
* Desc: This method should generate an event notification to send to D.A. system
* Params: @String tSensorType, Is this sensor digital or analog?
* @String tSensorDescription, Vendor id
* @String tSensorFunction, What does it do?
* @String tSensorStatus, Current sensor status active or inactive
* @String tSensorEvent, Reading changed or not changed from last snap etc
* @String tSensorEventType, Is optional or compulsory
* @unsigned long tSensorReadingA,Value that need to be send to D.A. module
* @unsigned long tSensorReadingB, (Optional) Value that need to be send to D.A. module
* @unsigned long tSensorReadingC, (Optional) Value that need to be send to D.A. module
* @String tSensorMountPorts, Port numbers where the sensor is mounted
* Returns: void, nothing
*
*/
```

Software Sensors | Google's Arts Creation



Weather stations (software sensors)

Based Google Arts http://data-arts.appspot.com

Software sensor's event:

```
<suggested_pickup_period>60</suggested_pickup_period>
        <location>La Junta Municipal, CO</location>
        <station_id>KLHX</station_id>
        <latitude>38.05</latitude>
        <longitude>-103.52</longitude>
        <observation_time>Last Updated on Jun 18 2011, 11:53 pm
MDT</observation_time>
        <observation_time_rfc822>Sat, 18 Jun 2011 23:53:00 -0600
</observation_time_rfc822>
        <weather>Fair</weather>
        <temperature_string>64.0 F (17.8 C)</temperature_string>
        <temp_f>64.0</temp_f>
        <temp_c>17.8</temp_c>
        <relative_humidity>56</relative_humidity>
        <wind_string>East at 9.2 MPH (8 KT)</wind_string>
        <wind_dir>East</wind_dir>
```

Mini Rule Engine

```
// Crude rule attributes
      struct CrudeRuleBase{

       int ruleIndex;            // Rule index number
       bool isRuleActive;        // If rule is active or inactive
       string ruleName;          // Human readable rule name
       string ruleDescription;   // Rule description if any
       unsigned int ruleAction;  // Action attached with rule
       string rulePayload;       // Any payload params needed by rule
       int ruleDelay;            // Wait time in rule execution sequence
       int nextRuleIndex;        // Next rule to be executed
      }crudeRuleDefinition;
```

Mini Rule Engine in action part 1

```
Sun Jul  3 18:03:36 2011 : Appended new rule: Start Point!
Sun Jul  3 18:03:36 2011 : Appended new rule: ADXL335
Sun Jul  3 18:03:36 2011 : Appended new rule: NOAA Weather feeds
Sun Jul  3 18:03:36 2011 : Appended new rule: End Point!
Sun Jul  3 18:03:36 2011 : Executing current rule : Start Point!
Sun Jul  3 18:03:36 2011 : Next rule in queue     : ADXL335
Sun Jul  3 18:03:36 2011 : Executing void action with payload: nothing
Sun Jul  3 18:03:37 2011 : Executing current rule : ADXL335
Sun Jul  3 18:03:37 2011 : Next rule in queue     : NOAA Weather feeds
Sun Jul  3 18:03:37 2011 : Executing 'ADXL335' sensor action with payload: nothing
<hardwaresensor>
<sensortype>digital</sensortype>
<sensordescription>ADXL335 3-Axis Accelerometer Sensors</sensordescription>
<sensorfunction>It gives a reading of acceleration per 3-axis</sensorfunction>
<sensorstatus>active</sensorstatus>
<sensorevent>changed</sensorevent>
<sensoreventyype>compulsory</sensoreventtype>
<sensorreadinga>362</sensorreadinga>
<sensorreadingb>371</sensorreadingb>
<sensorreadingc>437</sensorreadingc>
<sensortimestamp>17:51:0</sensortimestamp>
<sensormountports>A0,A1,A2,A3,AIN,AOUT</sensormountports>
</hardwaresensor>
```

Mini Rule Engine in action 2

```
Sun Jul  3 18:03:39 2011 : Executing current rule : NOAA Weather feeds
Sun Jul  3 18:03:39 2011 : Next rule in queue      : End Point!
Sun Jul  3 18:03:39 2011 : Executing 'NOAA Weather sensor' action with payload: nothing
Sun Jul  3 18:03:40 2011 :Station ID        : KMSY
Sun Jul  3 18:03:40 2011 :Station State     : LA
Sun Jul  3 18:03:40 2011 :Station Name      : New Orleans International Airport
Sun Jul  3 18:03:40 2011 :Station Latitude  : 29.98
Sun Jul  3 18:03:40 2011 :Station Longitude : -90.25
Sun Jul  3 18:03:40 2011 :Station HTML URL  : http://weather.noaa.gov/weather/current/KMSY.html
Sun Jul  3 18:03:40 2011 :Station RSS URL   : http://weather.gov/xml/current_obs/KMSY.rss
Sun Jul  3 18:03:40 2011 :Station XML URL   : http://weather.gov/xml/current_obs/KMSY.xml
Sun Jul  3 18:03:40 2011 :Station RFC time  : Sun, 19 Jun 2011 00:53:00 -0500
Sun Jul  3 18:03:40 2011 :Station Weather   : Fair
Sun Jul  3 18:03:40 2011 :--------------------------------------------------------------
```

Demos:

1. Standalone hardware sensor:
http://www.youtube.com/watch?v=78YGGzUHwDw

2. All weather sensors in USA
http://www.youtube.com/watch?v=Sklkz4zf9II

3. Sample auto selection of weather sensors based on levee's location:
http://www.youtube.com/watch?v=ql0NAJhETTI

What next?

- An alert generation and destributiuon system
- Log rotation and reporting
- Real-time xml processing engine
- Build and release package (probably .deb)
- Extended knowledge base
- Advance cronjobs!
- Real-time event handling and processing over D.A.
- Serial Over Internet | 802.11 | 802.11s

Current source code

Code repository:
http://code.google.com/p/levee-health-monitoring-and-surveillance/

Latest issues:
http://code.google.com/p/levee-health-monitoring-and-surveillance/issues/list

Revisions:
http://code.google.com/p/levee-health-monitoring-and-surveillance/source/list

Melange Time line:
http://www.google-melange.com/gsoc/project/google/gsoc2011/q1w2e3r4/12001