Model-based Wireless Health System Design Tool

IMPACT lab http://www.impact.asu.edu
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University, Tempe

1. *Model-based Wireless Health System Design Tool*

The tool automatically generates sensor and smart phone code from a given high-level specification of Wireless Health System (WHS).

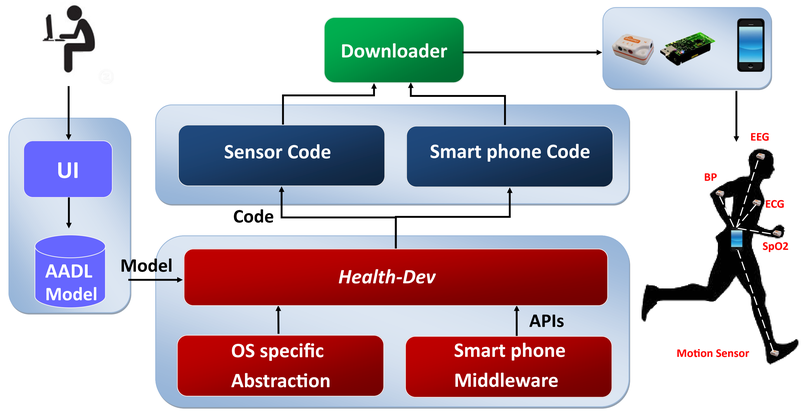
Problem

- Diversity in hardware and software of medical devices and smart phones makes rapid development of WHS challenging
- Manual implementation of WHS can be error prone and time consuming
- Requires programing background to implement WHS

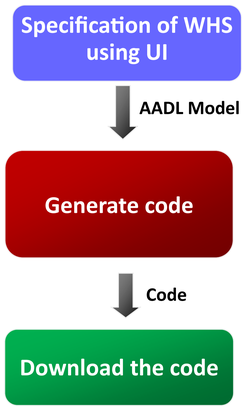
Solution—Model Based Engineering

A automated code generator for Wireless Health Systems is developed that takes high level models specified in Architecture Analysis and Design Language (AADL) as input and outputs platform specific code.

Wireless Health Systems Design Tool



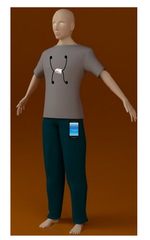
Design Steps



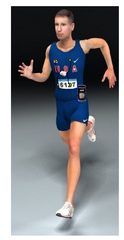
WHS Design Tool Usage

Use Case 1: Developing new health apps

Use Case 2: Provide physiological data to existing health apps

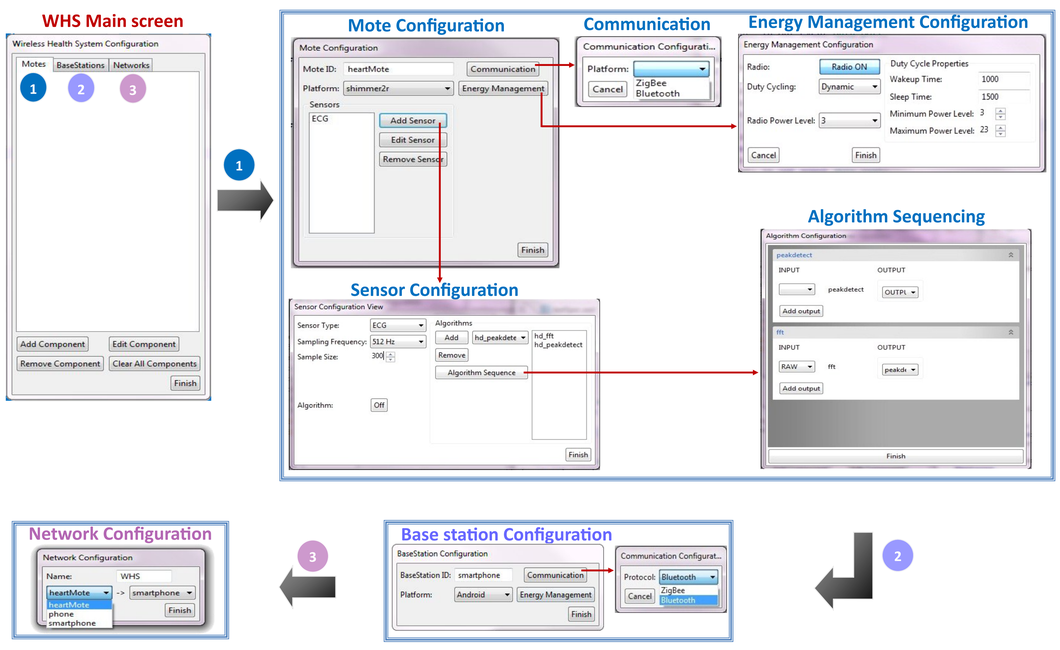


Use Case 1



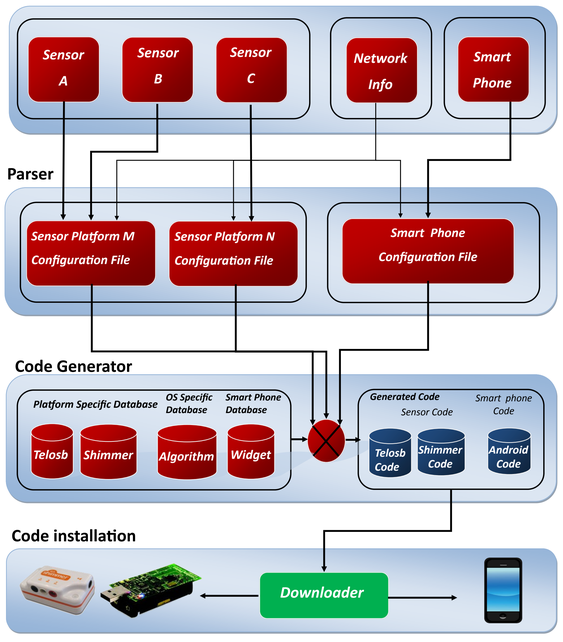
Use Case 2

2. *User Interface*



3. *AADL Model to Automatic Code Generation*

AADL Specification to Automatic Code Generation



AADL Sensor Specification

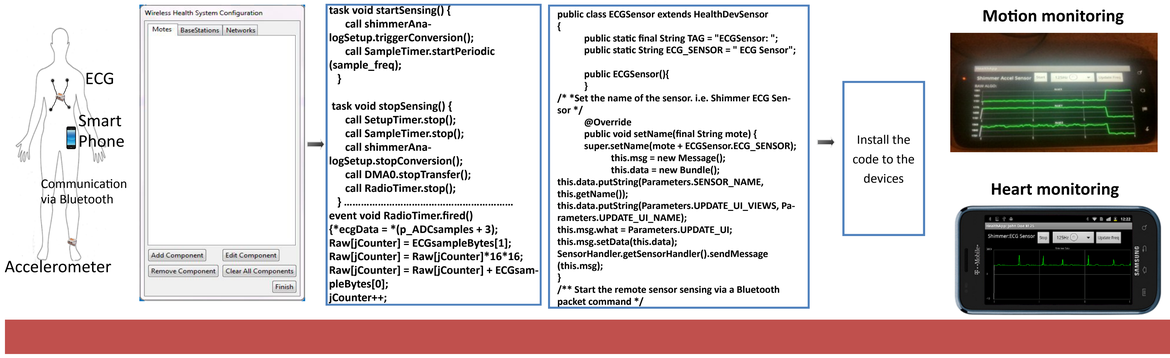
```
System Mote
      features
   input: in event data port;
   output: in out event data port;
end Mote;
System Sensor
      features
   input: in event data port;
   output: in out event data port;
end Sensor;
System implementation Mote.one_imp
subcomponents
   ECGSensor: system Sensor.ECG_one_imp;
   CommComp: system Communication.Mote_imp;
connections
   SendData: event data port output ->
   CommComp.input;
properties
   HD_MoteProperties::identifier => "439A07";
   HD_MoteProperties::platform => shimmer2r;
end Mote.one_imp;
System implementation Sensor.ECG.one_imp
subcomponents
   Algo1: process Algorithm.HD_FFT256_imp;
   Algo1: process Algorithm.HD_PeakDetection_imp;
properties
   HD_SensorProperties::sensor_type => ecg;
   HD_SensorProperties::sampling_frequency => 125;
   HD_SensorProperties::sample_size => 250;
connections
   Algorithm_C1: event data port input -> Algo1.input;
   Algorithm_C2: event data port Algo1.output -> Algo2.input;
   Algorithm_C2: event data port Algo2.output -> output;
end Sensor.ECG.one_imp;
```

TinyOS Code

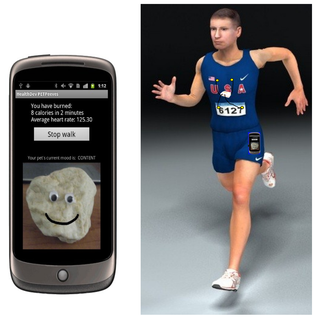
```
Header File
#define ECG      //Sensor type
#define BLUETOOTH  //Communication protocol

#define BUFFER_SIZE = 250;      //Sensor properties
enum{
SAMPLING_INTERVAL=8;          //Sensor properties
} .....

Header File
implementation {
#ifdef BLUETOOTH                          //Communication protocol
call BluetoothInit.Init();                // Mote property
call Bluetooth.setName("439A07");
.....

if (reading ==BUFFER_SIZE)
{
#ifdef ECG
post ecg_data_compute();                  //Algorithm handling
ecg_data_compute() {
Array* hd_fft256_var = make_AArray(512);
Array* hd_peakDetect_var= make_AArray(50);
hd_fft256(hd_fft_var, raw_data);
hd_peakDetect(hd_peakDetect_var, hd_fft_var);
.....

Algorithm File
void hd_fft256(AArray* hd_fft256_var, AArray* input)
{
.....                                     //Algorithm handling
}
void hd_peakDetect(AArray* hd_peakDetect_var, AArray* input)
{ .....

Wiring File
implementation {

components new TimerMilliC as SampleTimer;
#ifdef BLUETOOTH                          //Communication protocol
components RovingNetworksC;
```

Mote properties ***Sensor properties*** ***Communication handler*** ***Algorithm handler***

4. *Use Case 1: Developing new health apps*



5. *Use Case 2: Physiological data provider to existing app*



- We demonstrate an Android app which uses physiological data from WHS Base Station app.
- A user can create heart-rate monitoring application to send heart-rate to an Android application called PETPEEVES which is developed to promote exercise in a fun way.
- The facial expression of the pet is proportional to how much user exercise.

WHS Base Station

PETPEEVES (existing app)

The two apps run on same phone