DISTRIBUTED & PERVASIVE SYSTEMS

CONTEXT AWARENESS

STEFAN WAGNER
SW@ECE.AU.DK

AGENDA

CONTEXT AWARENESS

A key concept for achieving **calm technology** as envisioned by Weiser

Context-aware systems are aware of their context and must be able to differentiate where, they are, what is going on, with who, and why

Achieved by **equipping embedded devices with sensors and intelligence** including the ability to: Sense, model and reason on sensor data using relevant filtering and classification methods: Kalman, Monte Carlo Markov, Bayes classifiers, KNN, Decision trees, Hidden Markov Models, Neural Networks …

Sensor fusion is often needed for the getting the full picture

CONTEXT

Context is derived from the **Latin word: contextus**

- from con- “together” and texere “to weave”.

Context is fundamental to human cognition, which is the act or process of knowing, including perceiving, recognizing, conceiving, and reasoning. Words, sentences, images, and experiences can be fundamentally differently interpreted when served to the user in a different context
McCracken & Wolfe (2004)

Context is the circumstances that form the setting for an event, statement, or idea, and in terms of which it can be fully understood
Oxford Dictionay (2012)

CONTEXT

Plenty of definitions of context exists in pervasive computing:

1. A context describes a situation and the environment a device or user is in.
2. A context is identified by a unique name
3. For each context a set of features is relevant.
4. For each relevant feature a range of values is determined (implicit or explicit) by the context.

Schmidt (1999)

A set of environmental states and settings that either determines an application's behavior or in which an application event occurs and is interesting to the user
Chen & Kotz (2000)

Any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and the applications them-selves
Dey (2010)

The five W's (Who, What, Where, When, Why) is the minimum information that is necessary to understand context.
Abowd and Mynatt (2010)

CONTEXT AWARENESS

Again, several definitions exists:

Context-aware computing is the ability of a mobile user's applications to discover and react to changes in the environment they are situated in. In our system mobile users run software that is constantly monitoring, or subscribing to information about the world around them.
Schilit & Thiemer (1994)

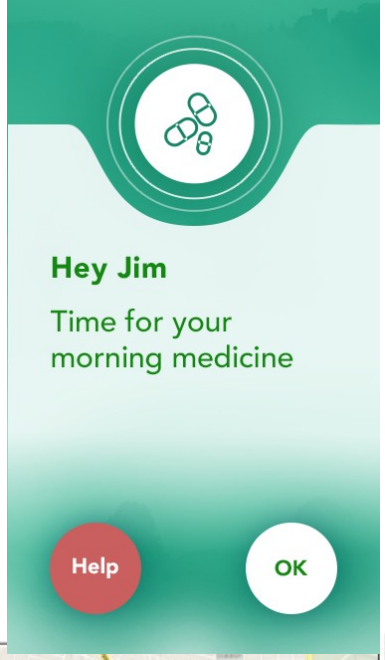
A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user’s task.
Dey (2010)

CONTEXT AWARENESS

Three features that context-aware applications may support:

1) Presentation of information and services to a user
2) Automatic execution of a service
3) Tagging of context to information for later retrieval

Abowd et al. (1999)



Hi you have just received the following results from the Adherence Logger

Patient id:mia
Start date: Thu Oct 25 2012 Start time: 22:39
End date: Wed Oct 24 2012 End time: 23:03
Email sent to: dk.iha.sensorlogger@gmail.com

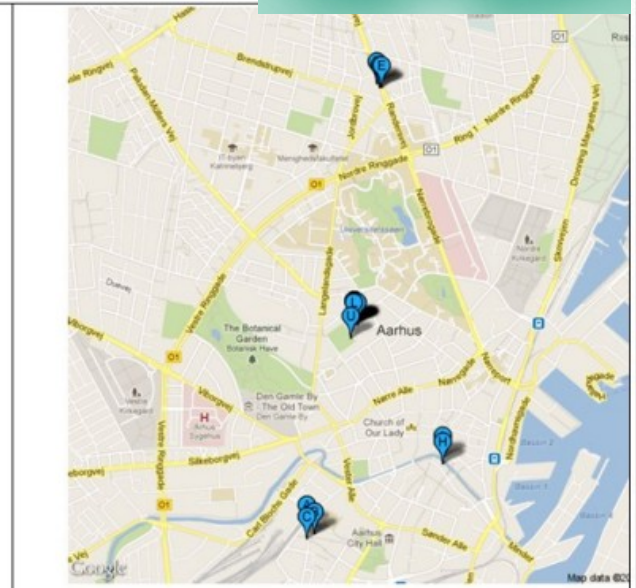
The table below shows your movement for the past 24 hours

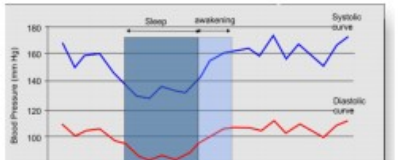
Time	Battery	Movement	Position
22:39	9	Low	NA,NA
22:24	9	Low	NA,NA
22:09	10	Low	NA,NA
22:03	12	Low	56.1632,10.20001

EDITED TO REDUCE SIZE

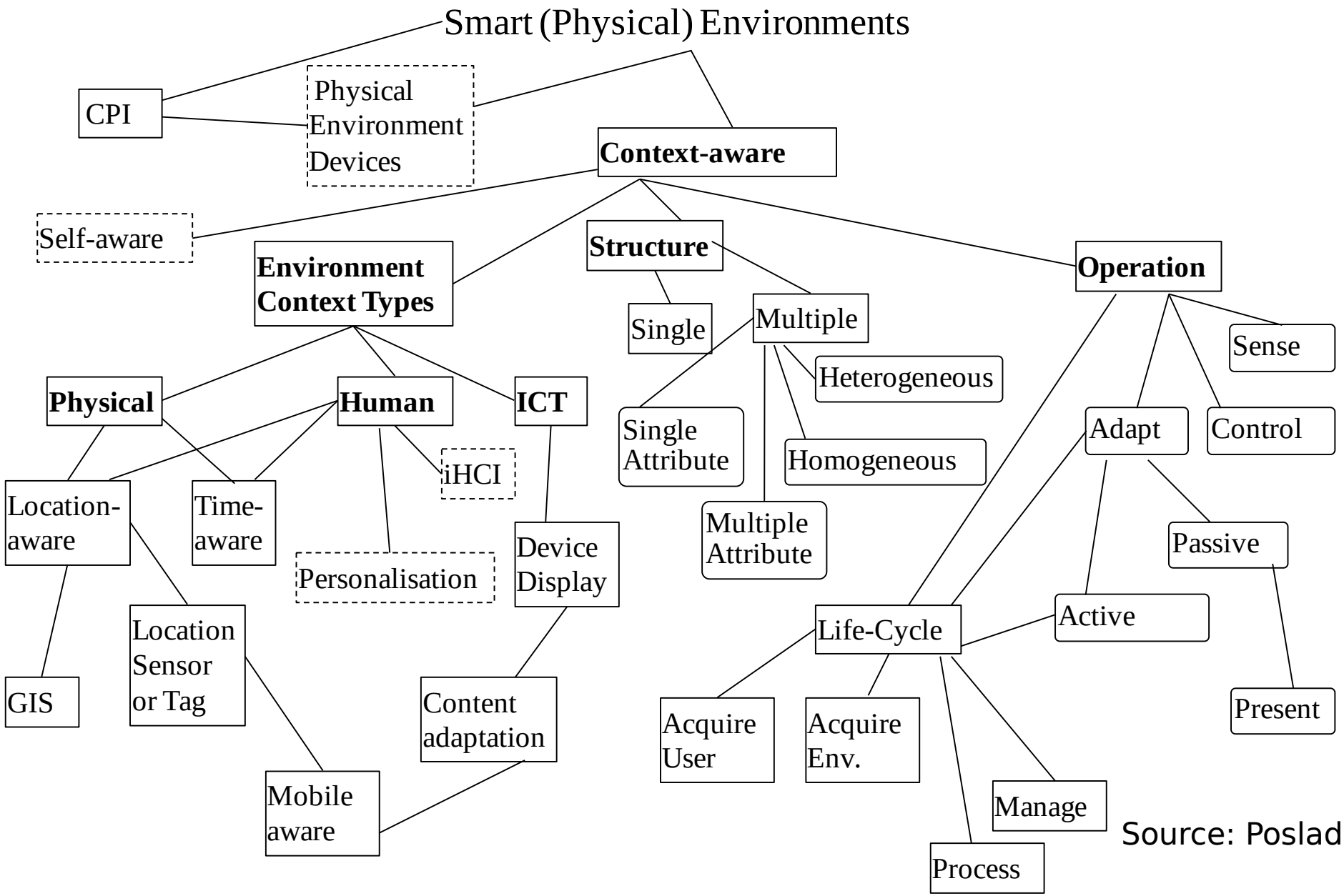
00:24	58	Low	NA,NA
00:09	59	Low	NA,NA
00:03	61	Low	56.16329,10.19996
23:39	61	Low	NA,NA
23:24	62	Low	NA,NA
23:09	64	High	NA,NA
23:03	65	High	56.16331,10.19985

Click on below link for map view of your movement for the past 24 hours:
http://maps.googleapis.com/maps/api/staticmap?&zoom=auto&size=600x300&maptype=roadmap&markers=color:blue%7Clabel:A%7C56.1632,10.20001&markers=color:blue%7Clabel:B%7C56.16329,10.20035&markers=color:blue%7Clabel:C%7C56.16319,10.20012&markers=color:blue%7Clabel:D%7C56.16332,10.19989&markers=color:blue%7Clabel:E%7C56.16312,10.19999&markers=color:blue%7Clabel:F%7C56.16329,10.19998&markers=color:blue%7Clabel:G%7C56.16323,10.21116&sensor





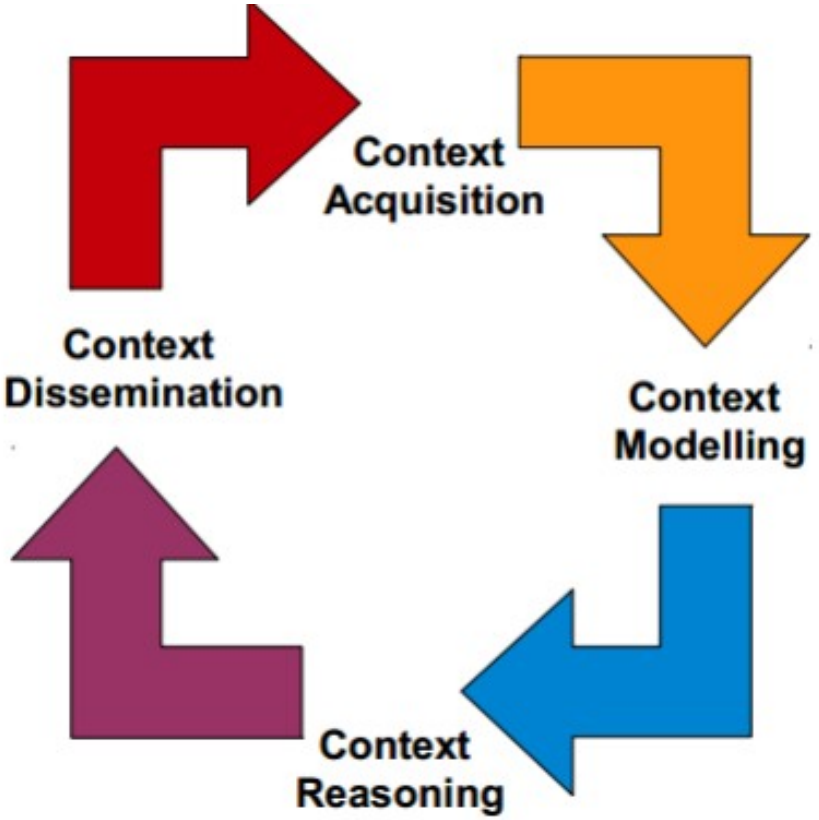
Smart (Physical) Environments



Source: Poslad 2009

LIFE-CYCLE FOR CONTEXT AWARENESS

Capture Physical Context
Capture User Context
Context Processin
Adapt to Context
Manage contexts



CONTEXT-AWARENESS: CHALLENGES

1. User Contexts may be incorrectly, incompletely, imprecisely determined or predicted ambiguously
2. Environment Contexts may be incorrectly, incompletely, imprecisely defined, determined or predicted.
3. Contexts may exhibits a range of spatial-temporal characteristics
4. Contexts may have alternative representations
5. Contexts may be distributed and partitioned, composed of multiple parts that are highly interrelated
6. Contexts may generate data huge volumes
7. Context sources and local processes often need to embedded in a low resource infrastructure
8. Context use can reduce the privacy of humans
9. Awareness of context shifts can distract users

CONTEXT COMPOSITION: CHALLENGES

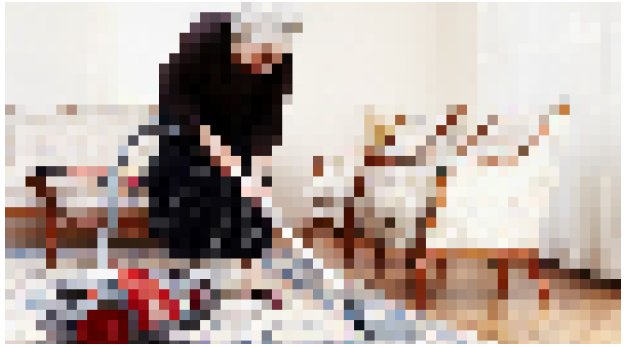
1. Handling heterogeneity of representation
2. Handling heterogeneity of meaning
3. Mediating and coordinating context aggregation
4. Ordering the adaptation to individual contexts
5. Different weightings for combining contexts
6. Handling uncertainty in combining contexts
7. Cost of sensors acquisition and maintenance

ACCURACY VS COSTS

A major challenge for context-aware systems are false positives – which degrades confidence in the systems – and often results in systems being “turned off”

On the other hand – sufficient sensors to avoid misclassification can be very expensive

A) Ambient sensor only (PIR)
Low acquisition and maintenance



B) Ambient + wearable
Medium acquisition and maintenance

C) Multiple ambient+wearables
High acquisition and maintenance

CONTEXT ADAPTATION: PASSIVE VS ACTIVE

Passive context adaptation system

- Context is presented to users
- Context-based tagging (chapter 6)
- System is not active in terms of adapting

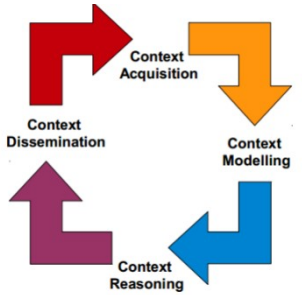
Active context-adaptation system

- Adaptation to context performed by the UbiCom system, not human users.

Hybrid context adaptive system

- Human user guides or corrects the automatic adaptation
- Hybrid context adaptive system can help reduce costs

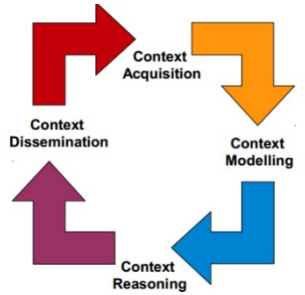
CONTEXT MODELS



A context model identifies a concrete subset of the context that is realistically attainable from sensors, applications and users and able to be exploited in the execution of the task. The context model that is employed by a given context-aware application is usually explicitly specified by the application developer, but may evolve over time.

A context attribute is an element of the context model describing the context. A context attribute has an identifier, a type and a value, and optionally a collection of properties describing specific characteristics
Henricksen (2003)

CONTEXT MODELS



Key-Value models

These models represent the simplest data structure for context modeling. They are frequently used in various service frameworks, where the key-value pairs are used to describe the capabilities of a service. Service discovery is then applied by using matching algorithms which use these key-value pairs.

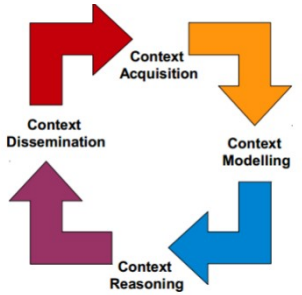
Markup scheme models

All markup based models use a hierarchical data structure consisting of markup tags with attributes and content.

Object oriented models

Modeling context by using object-oriented techniques offers to use the full power of object orientation (e.g., encapsulation, reusability, inheritance). Existing approaches use various objects to represent different context types (such as temperature, location, etc.), and encapsulate the details of context processing and representation.

CONTEXT MODELS



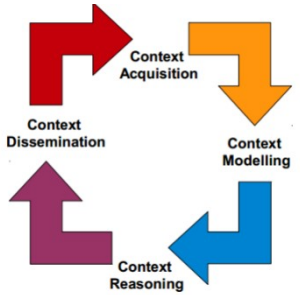
Logic based models

Logic-based models have a high degree of formality. Typically, facts, expressions and rules are used to define a context model. A logic-based system is then used to manage the aforementioned terms and allows to add, update or remove new facts. The inference (also called reasoning) process can be used to derive new facts based on existing rules in the systems. The contextual information needs to be represented in a formal way as fact.

Ontology based models

Ontologies represent a description of the concepts and relationships. Therefore, ontologies are a very promising instrument for modeling contextual information due to their high and formal expressiveness and the possibilities for applying ontology reasoning techniques. Various context-aware frameworks use ontologies as underlying context models

CONTEXT MODELS



Using AMML with ASET

```
<Rules>
    <Rule>
        <CollectedDataOnly>false</CollectedDataOnly>
        <EvaluationValueInIntervalOnly>true</EvaluationValueInIntervalOnly>
        <EvaluationValueNotInIntervalOnly>false</EvaluationValueNotInIntervalOnly>
        <RuleName>Time of day</RuleName>
        <EvaluationValues>
            <EvaluationValue>
                <ValueKey>Time of day</ValueKey>
                <ValueData>6.34908935E+17</ValueData>
                <ValueDataFormatted>07:00:00</ValueDataFormatted>
                </EvaluationValue>
                <EvaluationValue>
                <ValueKey>Time of day</ValueKey>
                <ValueData>6.34909E+17</ValueData>
                <ValueDataFormatted>10:00:00</ValueDataFormatted>
```

```
    </Rule
</Rules>
```

```
        static void Main(string[] args)
        {
            //Create an instance of the AMML Conversion Utility
            var converter = new AMMLConversionUtil();
            //Validate an existing document
            if (converter.ValidateAMMLDocument(@"RAM.xml", @"AMMLSchema.xsd", "AMML"))
            {
                //Convert it to an object model in order to illustrate the tool
                var aset = converter.DeserializeFromXML(@"RAM.xml");
                //And convert the object model back to AMML XML
                converter.ConvertToAMML(aset, @"RAM3.xml");
            }
```

CONTEXT REASONING

Context Reasoning and Interpretation

- Raw context data is often too noisy and too vague to provide the true context state of an entity
- Combining current context event data with historic events from one or multiple context sensors will allow us to determine a ”hidden state” based on the sensor fusion of one or more ”visible states”

Various approaches are possible:

- Rule based logic, Hidden Markov Models, Bayesian Networks, Artificial Neural Networks, Support Vector Machines, Ontology based, Fuzzy reasoning, k-

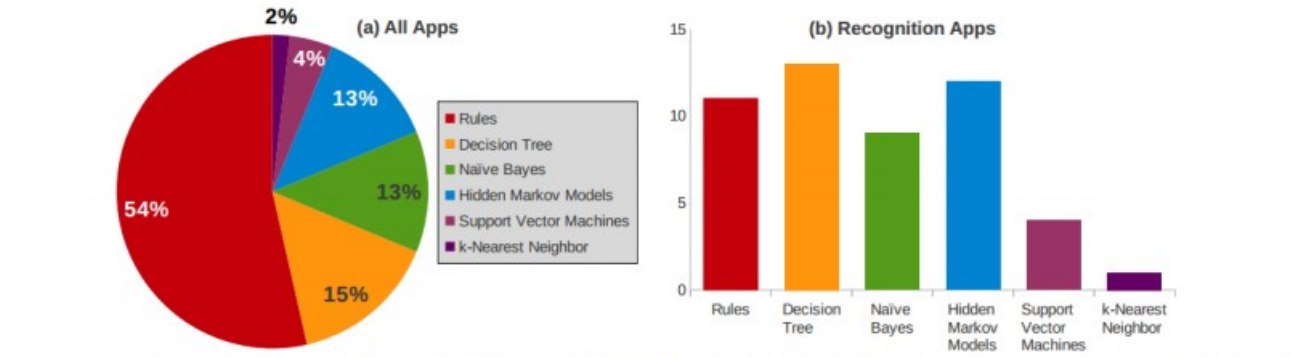
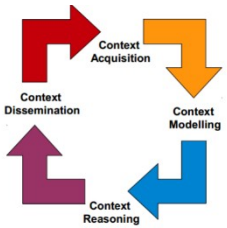
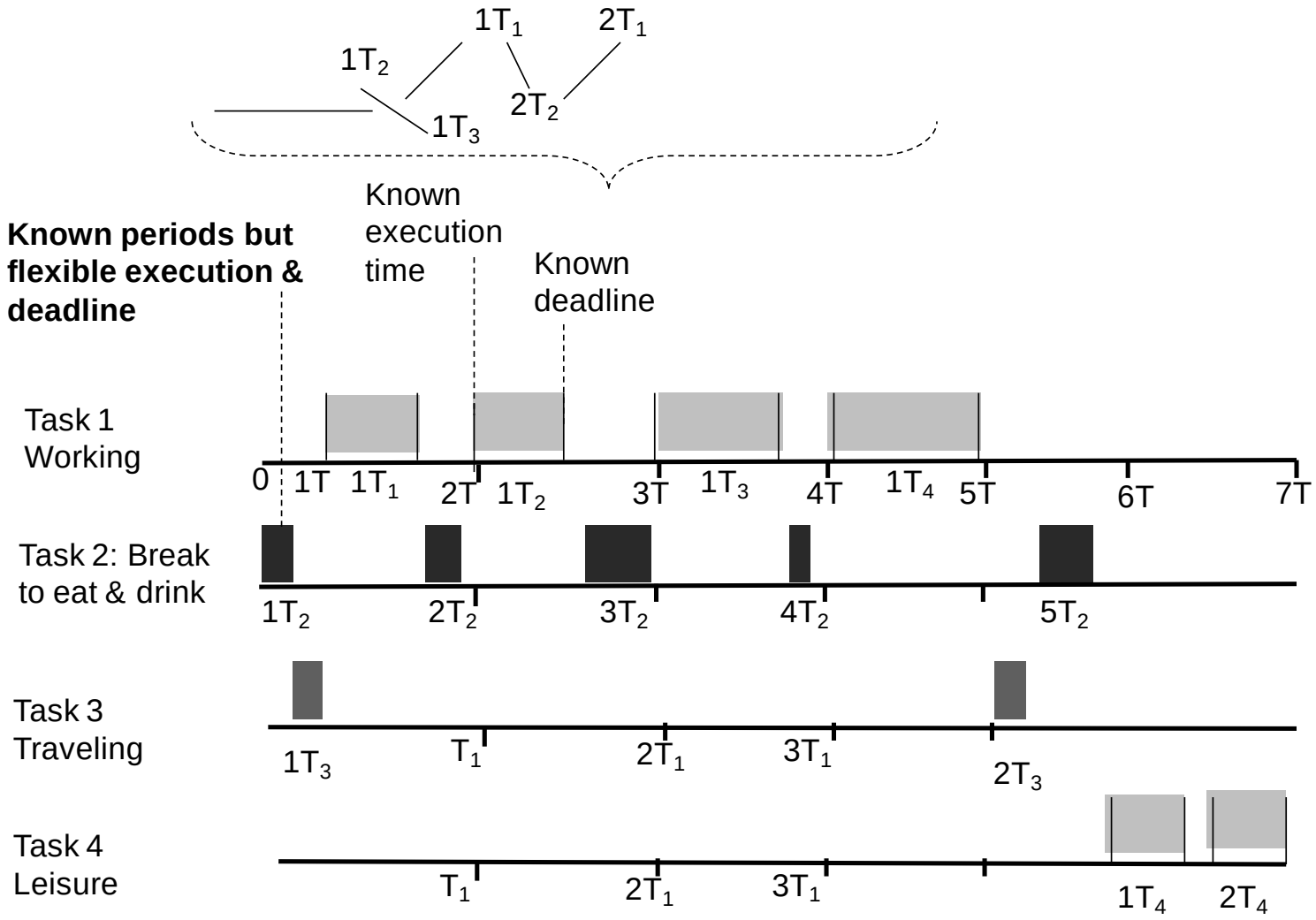


Fig. 7. (a) Counts of model types used in 109 of 114 reviewed context-aware applications. (b) Counts for 50 recognition applications; classifiers are used most often for applications that do recognition [108].



TIME AWARENESS: SCHEDULING



Source: Poslad (2009)

DECLARATIVE LOGIC PROGRAMMING

Sometimes – things are simple & we can use simple expert knowledge to encode situations

```
if in_meeting_now(E) then
   with_someone_now(E) ,
   has_entry_for_meeting_in_diary(E) .
if with_someone_now(E) then
   location*(E, L) , people_in_room*(L, N) , N > 1.
if has_entry_for_meeting_in_diary(E) then
   current_time*(T1) ,
   diary*(E, 'meeting', entry(StartTime, Duration)) ,
   within_interval(T1, StartTime, Duration) .
```

```
IF at_kitchen_on ANDlater tdRK_on ANDlater no_movement_detected
THEN assume the occupant has fainted
```

INTUITVE & SIMPLE CONTEXT REASONING

A RANGE OF EVENTS CAN BE UNCOVERED DIRECTLY FROM SENSORS:

If (toothbrush_active) => Brushing teeth

If (bed_sensor_active) => Sleeping or resting

If (TV is on && no_movement_in_room && couch_sensor_active) => Watching TV

If (Bathroom_presence && Bathroom_humidity_high) => Showering

But not all states can be inferred directly. Also, with more than one resident, problem arises.

Techniques for multi-resident activity recognition are generally composed by two parts: data association, and activity recognition. Data association (which user is generating the data) can be achieved with simple wearables - e.g. using presence sensors (e.g. Bluetooth or ZigBee RSSI tracking) – and then combining with ambient sensor data

INDRECT CONTEXT REASONING METHODS

Sometimes – we cannot directly infer the context out of sensor data.

Hidden Markov Models – rely on observed sensor events to uncover hidden activities

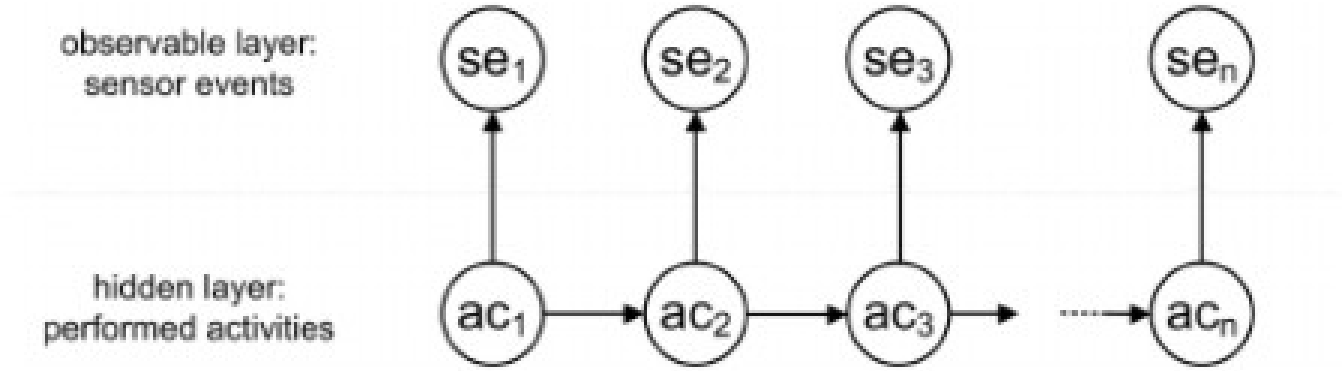


FIGURE 5. Hidden Markov Model for activity recognition.

Source: [Riboni and Murru (2020)]

HMM REASONING WITH CASAS

Source: Riboni and Murru (2020)

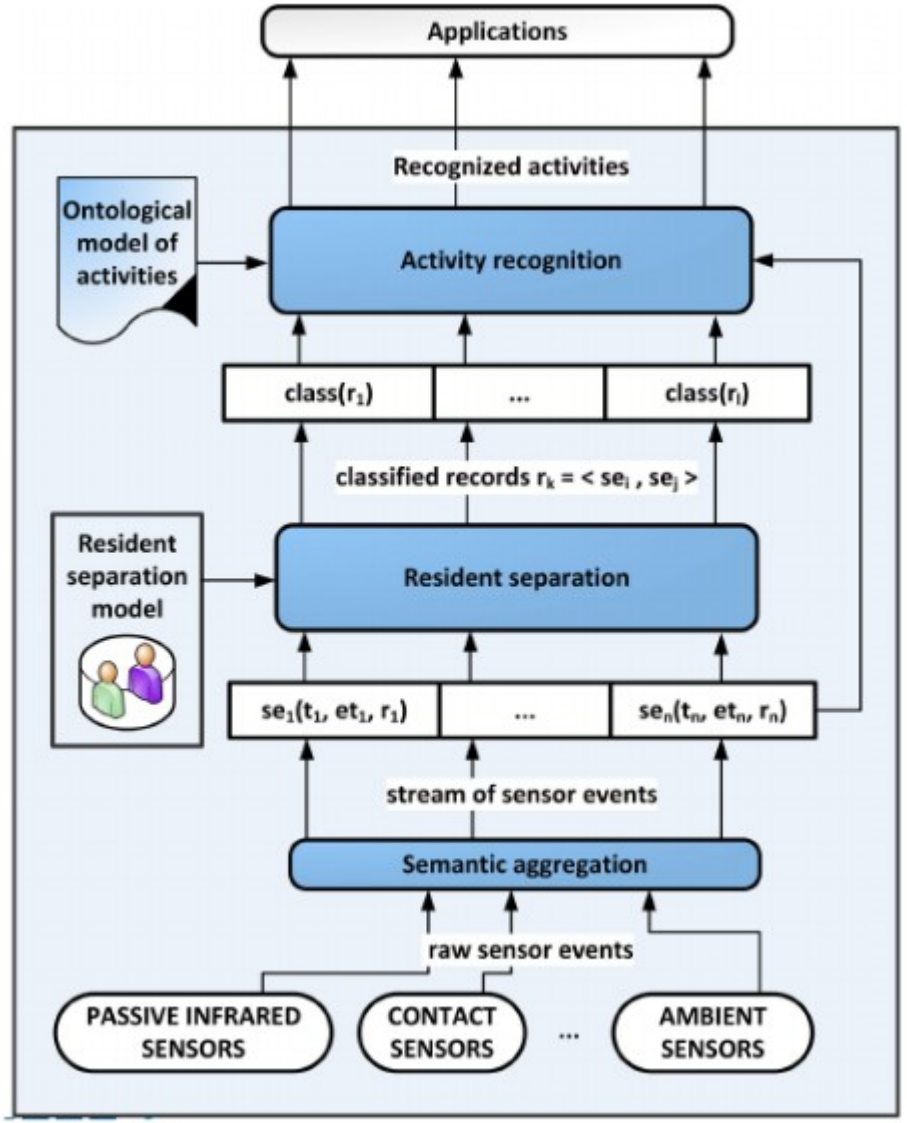


FIGURE 2. Architecture for multi-inhabitant activity recognition.

Algorithm ***Consecutive events***(T):
Input: A temporal sequence of sensor events T
Output: A set R of records with associated statistics

1: **for each** couple of consecutive sensor events $\langle se_1, se_2 \rangle \in P$ **do**
2: $c = \langle se_1, se_2 \rangle$
3: $c.occurrences = c.occurrences + 1$
4: $R.add(c)$
5: **end for**
6: **for each** $c = \langle se_1, se_2 \rangle \in R$ **do**
7: $n_{se_1} = |\{se \in T : se = se_1\}|$
8: $c.frequency = \dfrac{c.occurrences}{n_{se_1}}$
9: **end for**
10: **return** R

FIGURE 3. Extraction of the resident separation model according to the *consecutive events* approach.

ID	Description	Actor(s)
1	Fill a medication dispenser and return items	Resident A
2	Hang up clothes in the hallway closet	Resident B
3	Move the couch and coffee table to the other side of the living room	Resident B asking help from Resident A
4	Sit on the couch and read a magazine	Resident B
5	Water plants	Resident A
6	Sweep the kitchen floor	Resident B
7	Play a game of checkers	Resident A and Resident B
8	Set out ingredients for dinner	Resident A
9	Set dining room table	Resident B
10	Sit on the couch and read a magazine	Resident A
11	Pay an electric bill	Resident A asking help from Resident B
12	Prepare a picnic basket	Resident A
13	Retrieve dishes	Resident B asking help from Resident A
14	Pack supplies in the picnic basket	Resident B
15	Pack food in the picnic basket	Resident A

CASAS

[CASAS Data s](#)

Center of Advanced Studies In Adaptive System

home / about / research projects / tools / datasets / publications / classes / opportunities

TOOLS

WSU CASAS Tools

- Multi-resident tracking
- Mobile activity recognition
- Extract digital behavior markers from activity labeled mobile sensor data
- Activity recognition from ambient sensor data
- CASAS smart home dashboard (updated daily)

CASAS DATASETS

CASAS Smart Home Data Sets

Dataset	*Testbed*	*#Residents / #Participants*	Description	Annotated	Last Updated
1	Kyoto	20	ADL Activities	Yes	4/27/2010
2	Kyoto	20	ADL Activities with Errors	Yes	7/7/2009
3	Kyoto	20	Interweaved ADL Activities	Yes	7/7/2009
4	Kyoto	40	Multiresident ADL Activities	Yes	7/7/2009
5	Kyoto	1	Daily life, 2007	Partial	7/10/2009
6	Kyoto	2	Daily life, 2008	Yes	5/24/2010
7	Kyoto	2	Daily life, Spring 2009	Yes	7/8/2014
8	Kyoto	2	Daily life, Summer 2009	Yes	5/13/2010
9	Tulum	2	Daily life, 2009	Yes	1/18/2011

SINGLE SENSOR ACTIVITY RECOGNITION

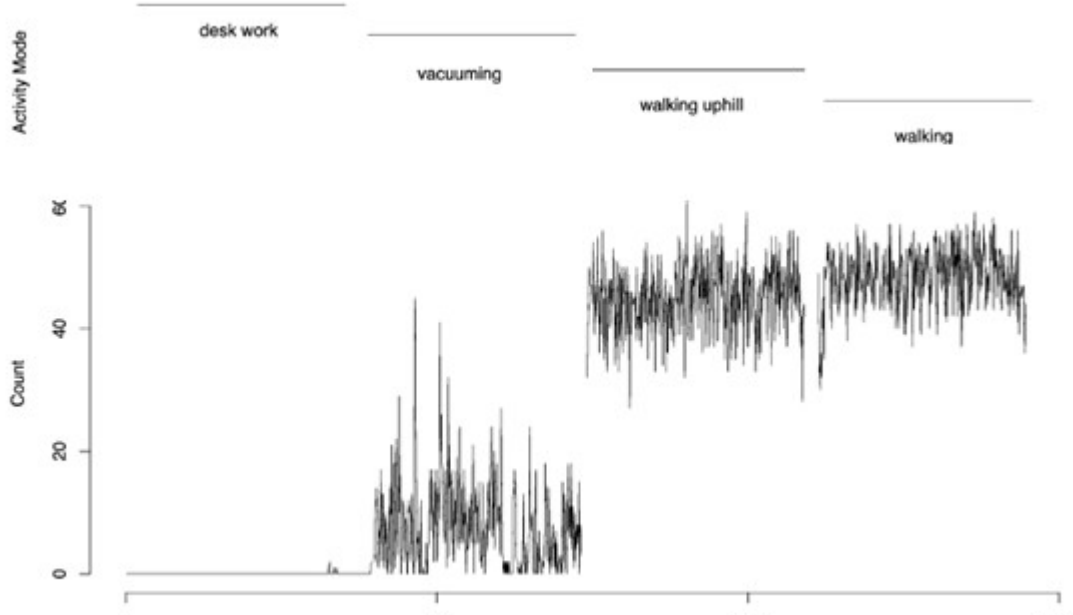


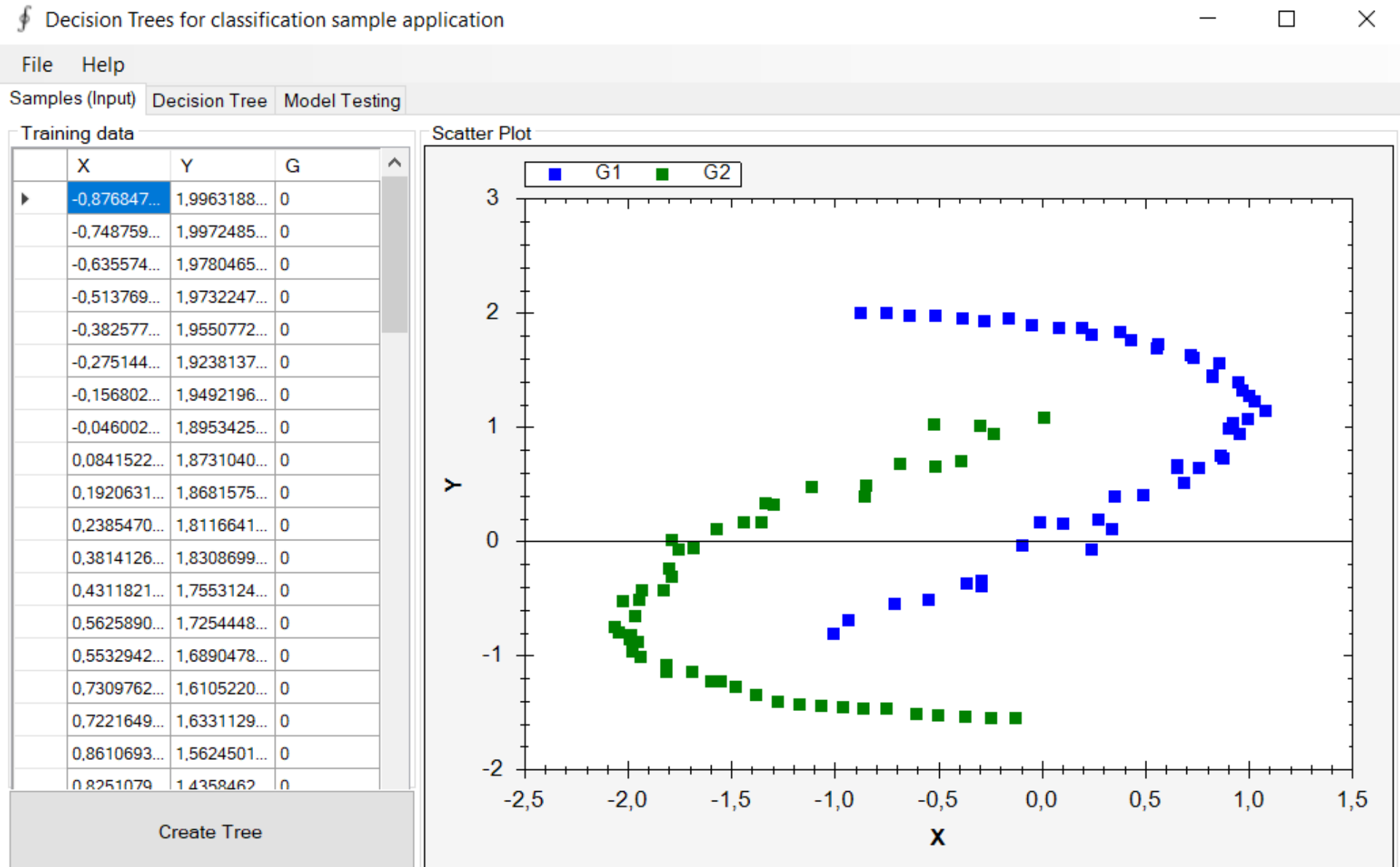
TABLE 4. Confusion matrix for the HMM.

	Truth			
Classification	**Walking (%)**	**Walking Uphill (%)**	**Vacuuming (%)**	**Computer Work (%)**
Walking	62.6	37.3	0.0	0.0
Walking uphill	36.9	62.5	0.0	0.0
Vacuuming	0.3	0.2	98.8	2.7
Computer work	0.2	0.0	1.2	97.3

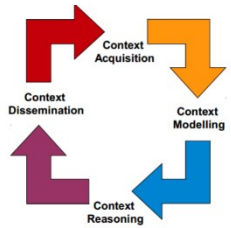
The ith row and jth column of this table contains the mean percentage of seconds in which activity j was classified as activity i by the model.

Pober et al. 2006

DECISION TREE EXAMPLE

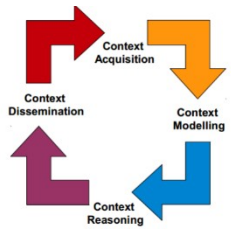


Source: Accord Framework
http://accord-framework.net/samples.htn



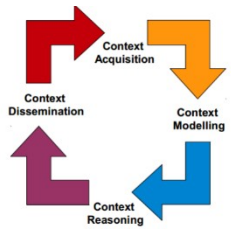
DECISION TREE CODE

```
// Get only the input vector values (first two columns)
double[][] inputs = table.GetColumns(0, 1).ToJagged();

// Get only the output labels (last column)
int[] outputs = table.GetColumn(2).ToInt32();


// Specify the input variables
DecisionVariable[] variables =
{
    new DecisionVariable("x", DecisionVariableKind.Continuous),
    new DecisionVariable("y", DecisionVariableKind.Continuous),
};

// Create the C4.5 learning algorithm
var c45 = new C45Learning(variables);

// Learn the decision tree using C4.5
tree = c45.Learn(inputs, outputs);
```



DECISION TREE CODE

```
// Creates a matrix from the entire source data table
double[][] table = (dgvLearningSource.DataSource as DataTable).ToArray(out columnNames)

// Get only the input vector values (first two columns)
double[][] inputs = table.GetColumns(0, 1);

// Get the expected output labels (last column)
int[] expected = table.GetColumn(2).ToInt32();


// Compute the actual tree outputs
int[] actual = tree.Decide(inputs);


// Use confusion matrix to compute some statistics.
ConfusionMatrix confusionMatrix = new ConfusionMatrix(actual, expected, 1, 0);
dgvPerformance.DataSource = new [] { confusionMatrix };

// Create performance scatter plot
CreateResultScatterplot(zedGraphControl1, inputs, expected.ToDouble(), actual.ToDouble(
```



EXERCISE – GROUPS OF 2-3

Discuss how YOU would:

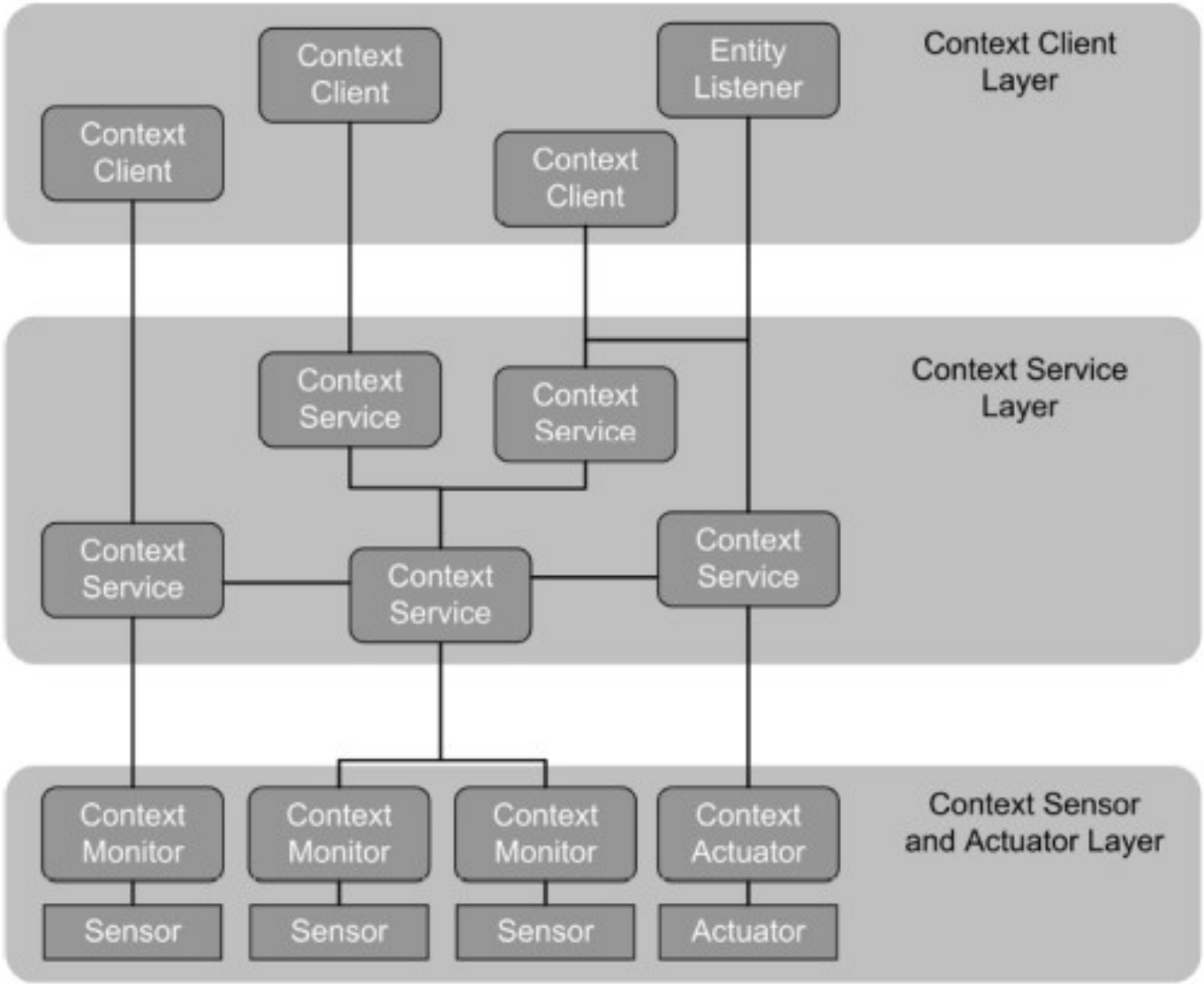
1) Model the context of a user in his/her home (e.g. rooms, devices, occupancy, activity)

2) Use code to reason what type of activity the user is doing and where it is going on

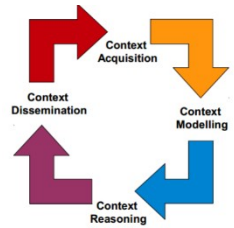
3) Share information gathered from sensor data and context services

Feel free to be very specific about any specific sensors, code, etc.

JCAF CONTEXT ARCHITECTURE



Bardram 2008



CONTEXT AWARENESS CHALLENGES

Sensors

- Granularity, accuracy, precision vs cost
- Sensor drift over time and need for recalibration
- Power consumption vs precision
- Maintenance costs
- Communication interface features vs costs
- Detection of failing sensors

Networking

- Wired vs. wireless
- Protocol(s)
- Bandwidth vs power consumption
- Range vs interference
- Topology

System

- 24 -7 system designs vs low cost & low power hardware
- Privacy and security
- Agent-based vs. monolithic vs service orchestration
- Open vs. closed
- Volatility when adding and removing sensors and clients

QUESTIONS?

UBICOMP METHODOLOGY

Main research approach used in early ubiquitous computing projects:

“the construction of working prototypes of the necessary infrastructure in sufficient quality to debug the viability of the system in daily use; ourselves and a few colleagues serving as guinea pigs” [Weiser]

Used today in pervasive computing research

PROTOTYPING

Prototype derives from the Greek word *prototypon* meaning "primitive form", and *prototypos*, "original, primitive", from *protos*, meaning "first" and *typos*, "impression".

Prototyping is a Systems Engineering method

Part of the “simulation” and “modelling” families

CONTEXT SENSOR CHALLENGES

Which context sensors are relevant?

How many sensors are needed to get the nessecary fidelity?

Amibent and wearable hybrids?

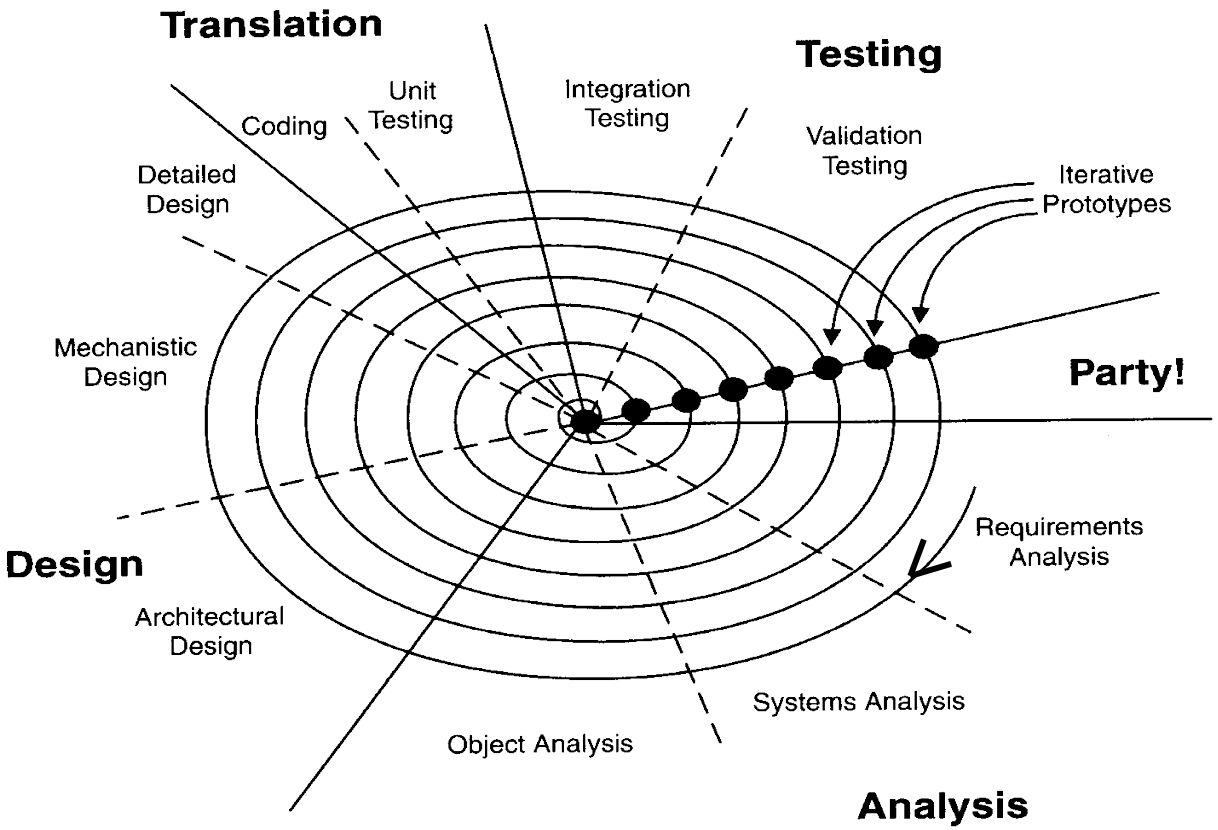
How do the sensors work in longitudinal scenarios?

How are they perceived by usrse?

WE NEED TO EXPERIMENT BEFORE INVESTING IN THE FULL SYSTEM

ITERATIVE PROTOTYPING USING SOFTWARE AND HARDWARE

ITERATIVE PROTOTYPING PROCES



(Bohm 1988)

CHALLENGES IN PROTOTYPING

Identifying and involving stakeholders:

- users, managers, developers, customer reps?, union reps?, shareholders?

Involving stakeholders:

- workshops, interviews, workplace studies, co-opt stakeholders onto the development team

Who are the ‘real’ users?

- not managers, not marketing, not expert users, not engineers

Political problems within the organisation

Dominance of certain stakeholders

Economic and business environment changes

Balancing functional and usability demands

Rosenthal and hawthorne effects (and other confirmation bias)

FIELD STUDIES & OBSERVATIONS

Direct observation:

- Gain insights into stakeholders' tasks
- Good for understanding the nature and context of the tasks
- Will provide an insight into which sensors are needed
- Requires time and commitment from design team, and it can result in a huge amount of data

Indirect observation:

- Log activities (using early prototypes: movement sensors, sound etc, wearables)
- Good for understanding current tasks

TECHNIQUES FOR PROTOTYPING

Storyboards / mock-ups / sketches

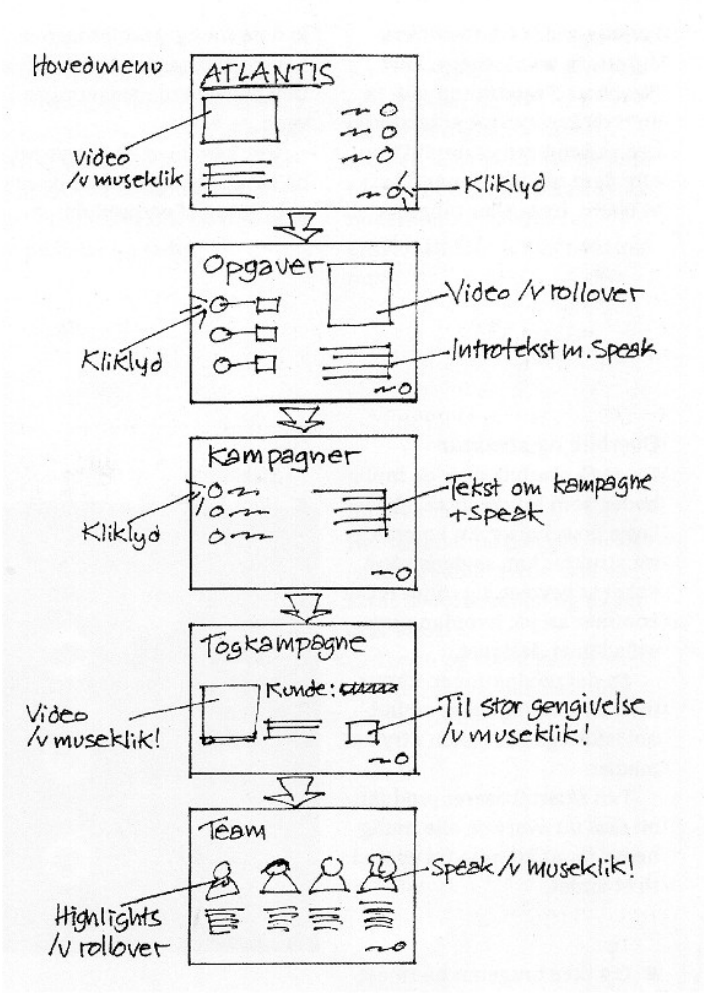
- prototypes with low fidelity (= precision)
- need not be computer-based – paper mock-ups
- AKA paper prototypes

Limited func simulations = scenarios

- One part of functionality provided by designers
- RAD tools may be used for these (Visual Studio)
- Most often mock-ups are ok
- Wizard of Oz technique
- Video prototypes

Horisontal / Vertical advanced prototypes

- Depending on what needs to be tested
- RAD tools are common for these
- Throw-away, incremental, evolutionary



LONGITUDINAL PROTOTYPING

Traditional prototyping often within confined timespan

Sometimes more feasible to test concepts over time

Wizard of oz and video prototypes not feasible

Higher demands:

- Must be easy to install and activate
- Must be sufficiently easy to use
- Must be sufficiently robust (running e.g. 1 day, 1 week, 1 month …)
- Must be able to store or forward test data
- Must (often) be able to distribute data (or at least status) via a network
- Must have sufficient power supply

LONGITUDINAL PERVASIVE PROTOTYPING

My research group uses the following types of longitudinal prototype technology

1) Phidgets for early & small-scale longitudinal prototyping
2) Zigbee devices for large scale longitudinal prototyping
3) Android smart phone & smart watch API’s for wearable prototyping

PiCom, Ardunio, and a host of alternatives exists

WHAT ARE PHIDGETS?

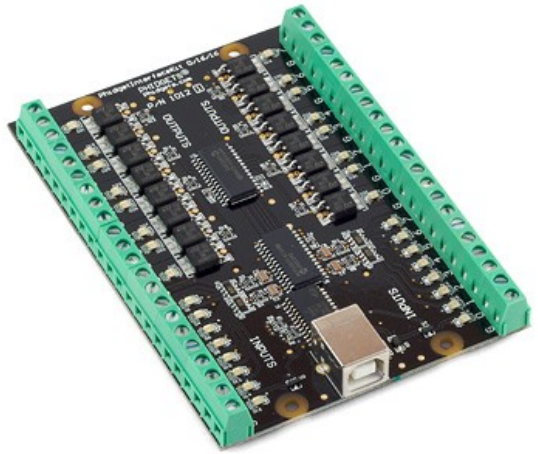
Rapid Prototyping for ICT Engineers

Pervasive Computing Building Blocks

- Sensors
- Actuators

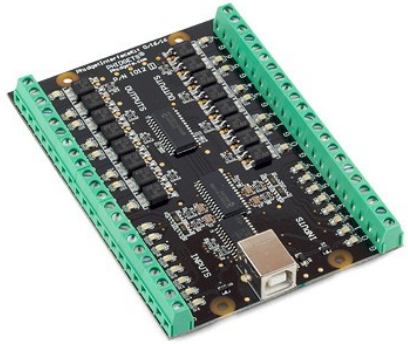
Simplifies prototyping

Easy to use API’s



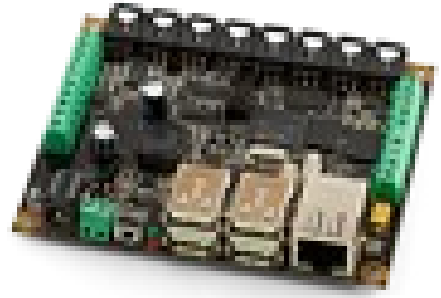
I/O BOARDS

Connects easily to PC using USB



Basic Analog & Digital I/O

I/O with USB Hub



Advanced Single Board Computer

- Running Linux
- C++ / Java on-board
- USB / LAN / Wifi Connection

SENSOR TYPES

Distance
Sonar
Inductive proximity
Pressure
Light
Humidity
Magnetic
Temperature
Accellerometer
Current/Voltage

MAN MACHINE INTERFACE

Circular touch
Liniear touch
Point touch
Joystick
Slider
Displays

RFID

Phidget RFID

ACTUATORS

Liniear actuators

Stepper motors

Servo motors

HOW TO DEVELOP PROTOTYPES?

Connect the physical devices to I/O boards

Connect the I/O Boards to a PC / OMAP / WCE device

Install the drivers

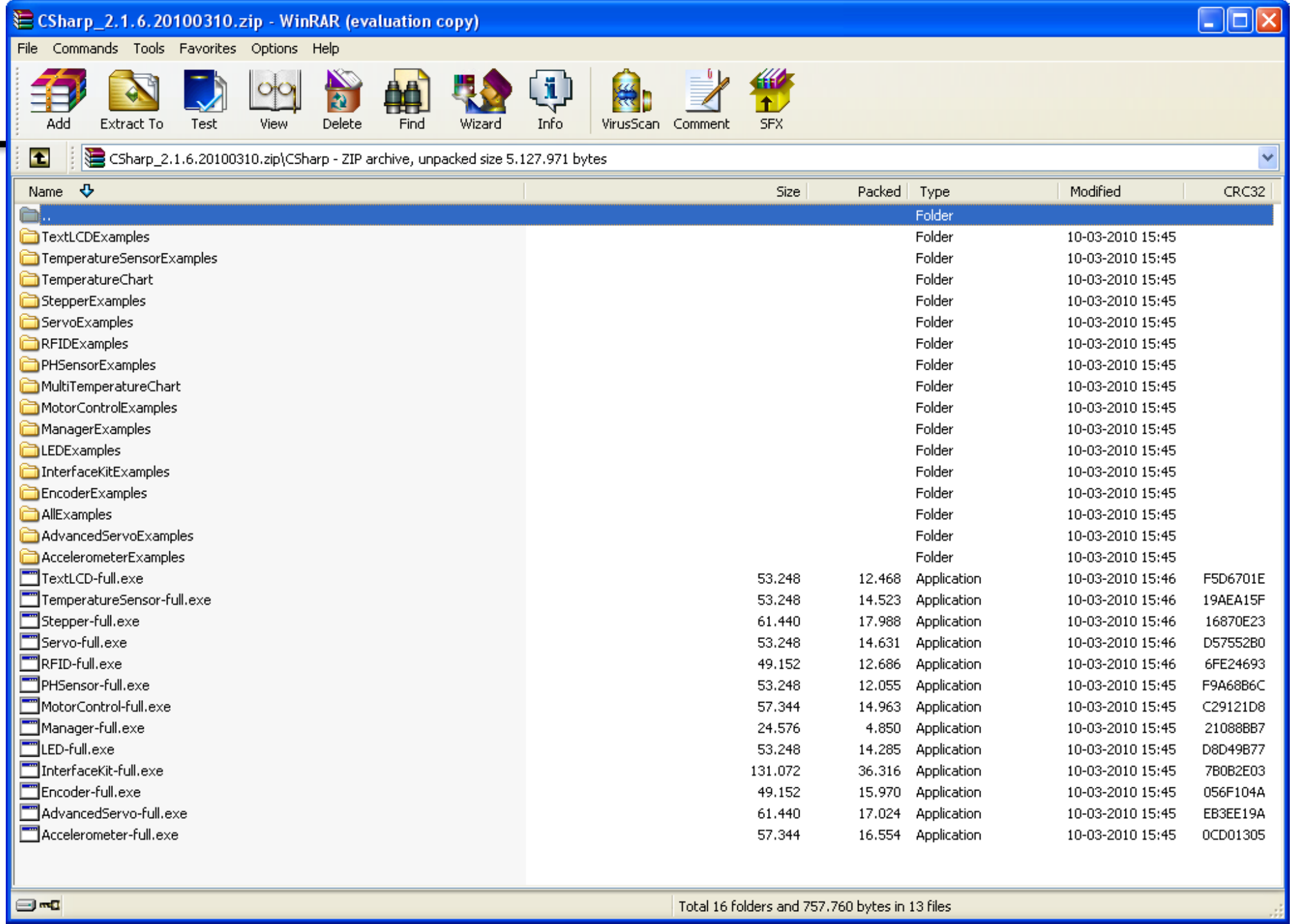
Start programming!

PROGRAMMING

Huge support

- Platforms: Windows, Linux, OSX, Android, iOS,
- Languages: C, C++, Python, JavaScript, C#, Java, VB, Max/MSP, LabView

Core Languages	Mobile Languages	Other Languages
C#	Objective C	LabVIEW
C/C++	Swift	Max/MSP
Python	Android Java	
Java		
Visual Basic .NET		
JavaScript		


CSharp_2.1.6.20100310.zip - WinRAR (evaluation copy)
File Commands Tools Favorites Options Help
Add Extract To Test View Delete Find Wizard Info VirusScan Comment SFX
CSharp_2.1.6.20100310.zip\CSharp - ZIP archive, unpacked size 5.127.971 bytes
Name Size Packed Type Modified CRC32
.. Folder
TextLCDExamples Folder 10-03-2010 15:45
TemperatureSensorExamples Folder 10-03-2010 15:45
TemperatureChart Folder 10-03-2010 15:45
StepperExamples Folder 10-03-2010 15:45
ServoExamples Folder 10-03-2010 15:45
RFIDExamples Folder 10-03-2010 15:45
PHSensorExamples Folder 10-03-2010 15:45
MultiTemperatureChart Folder 10-03-2010 15:45
MotorControlExamples Folder 10-03-2010 15:45
ManagerExamples Folder 10-03-2010 15:45
LEDExamples Folder 10-03-2010 15:45
InterfaceKitExamples Folder 10-03-2010 15:45
EncoderExamples Folder 10-03-2010 15:45
AllExamples Folder 10-03-2010 15:45
AdvancedServoExamples Folder 10-03-2010 15:45
AccelerometerExamples Folder 10-03-2010 15:45
TextLCD-full.exe 53.248 12.468 Application 10-03-2010 15:46 F5D6701E
TemperatureSensor-full.exe 53.248 14.523 Application 10-03-2010 15:46 19AEA15F
Stepper-full.exe 61.440 17.988 Application 10-03-2010 15:46 16870E23
Servo-full.exe 53.248 14.631 Application 10-03-2010 15:46 D57552B0
RFID-full.exe 49.152 12.686 Application 10-03-2010 15:46 6FE24693
PHSensor-full.exe 53.248 12.055 Application 10-03-2010 15:45 F9A68B6C
MotorControl-full.exe 57.344 14.963 Application 10-03-2010 15:45 C29121D8
Manager-full.exe 24.576 4.850 Application 10-03-2010 15:45 21088BB7
LED-full.exe 53.248 14.285 Application 10-03-2010 15:45 D8D49B77
InterfaceKit-full.exe 131.072 36.316 Application 10-03-2010 15:45 7B0B2E03
Encoder-full.exe 49.152 15.970 Application 10-03-2010 15:45 056F104A
AdvancedServo-full.exe 61.440 17.024 Application 10-03-2010 15:45 EB3EE19A
Accelerometer-full.exe 57.344 16.554 Application 10-03-2010 15:45 0CD01305
Total 16 folders and 757.760 bytes in 13 files

SAMPLE C#

Phidgets21: [Language - C Sharp - Phidgets Legacy Support]

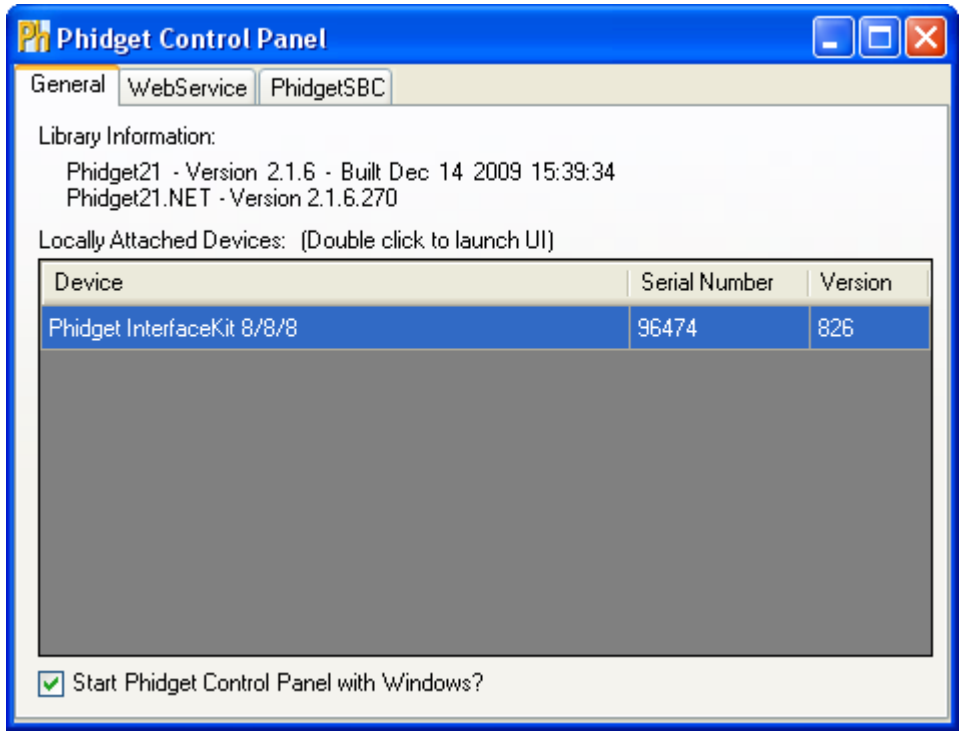
```
using Phidgets;
using Phidgets.Events;
// using.....

namespace Program {
    class Code {
        static void Main(string[] args) {
            try {

                // Declare a TemperatureSensor object
                TemperatureSensor device = new TemperatureSensor();

                // Hook in any event handlers
                // ...

                // Open the device
                device.open();

            } catch (PhidgetException ex) { Console.WriteLine(ex.Description); }
        }
    }
}
```

```
ifKit.SensorChange += new SensorChangeEventHandler(ifKit_SensorChange);
```

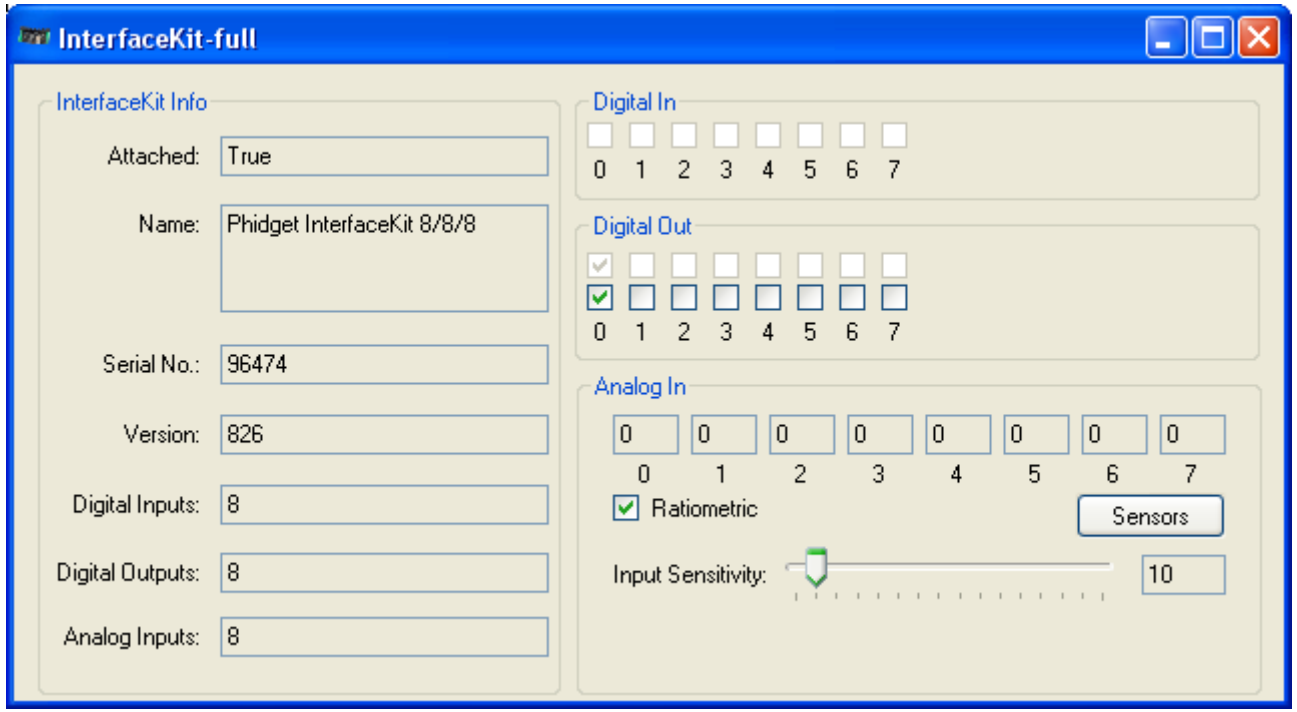
The ifKit_SensorChange method is defined as follows:

```
void ifKit_SensorChange(object sender, SensorChangeEventArgs e) {
   textBox1.Text = "Index " + e.Index + " Value: " + e.Value;
}
```

NEED TO ATTACH THE I/O BOARD



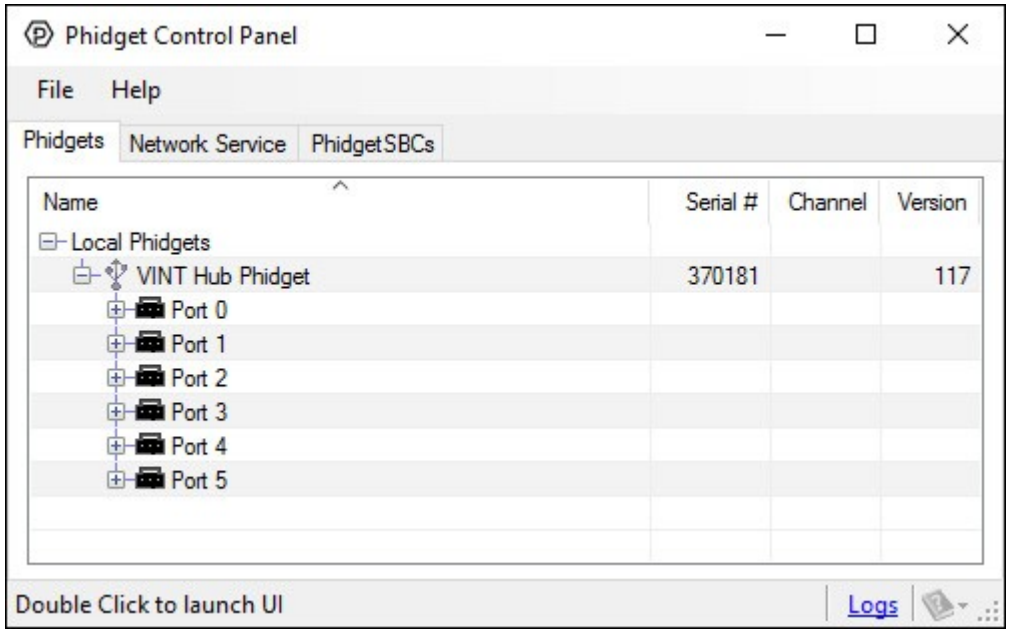
GETTING A LED RUNING ...

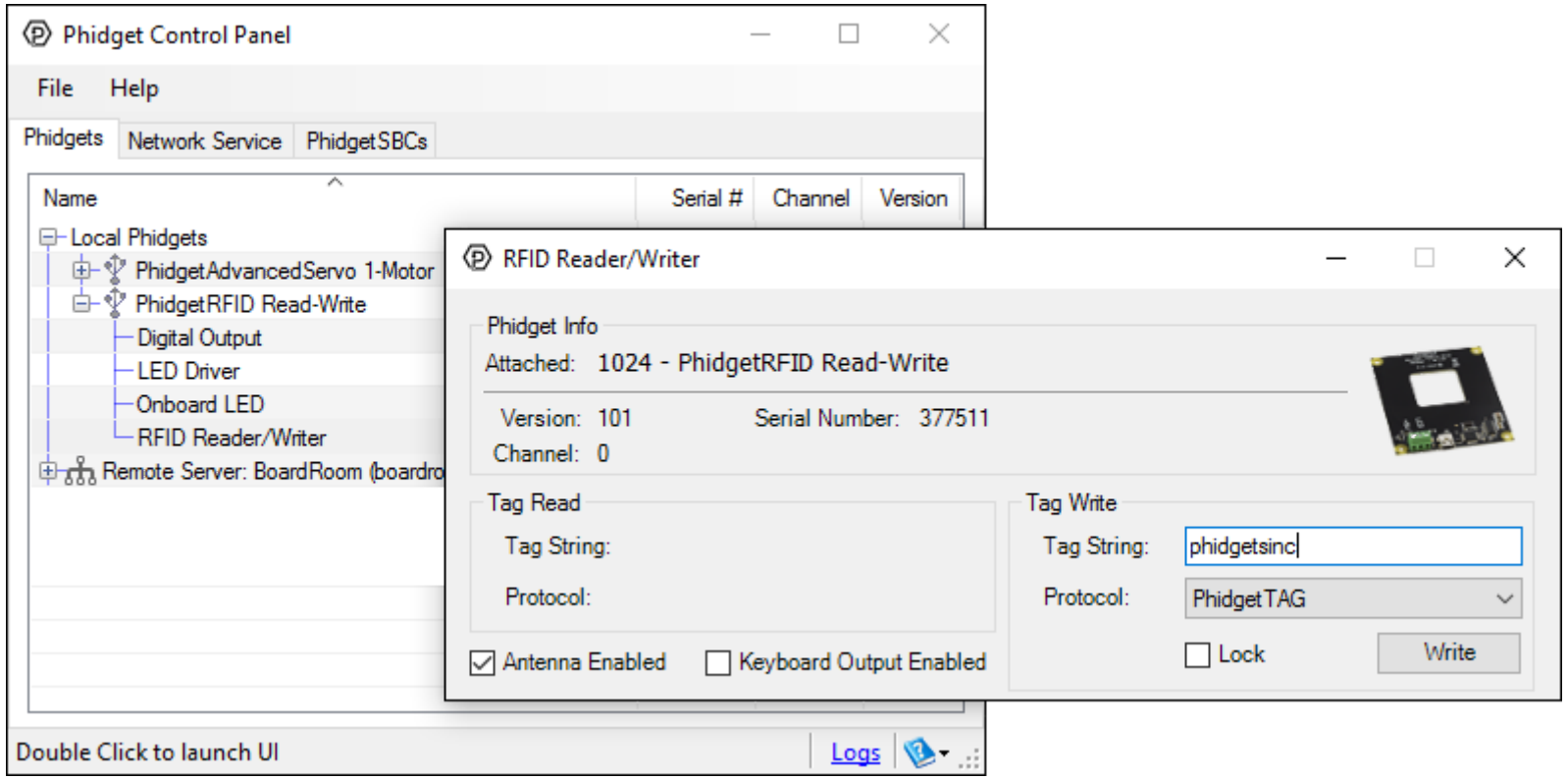


HOOK IT UP – AND YOU ARE READY!

```
//Declare an InterfaceKit object
static InterfaceKit ifKit;

static void Main(string[] args)
{
    try
    {
        //Initialize the InterfaceKit object
        ifKit = new InterfaceKit();

        //Hook the basica event handlers
        ifKit.Attach += new AttachEventHandler(ifKit_Attach);
        ifKit.Detach += new DetachEventHandler(ifKit_Detach);
        ifKit.Error += new ErrorEventHandler(ifKit_Error);

        //Hook the phidget spcific event handlers
        ifKit.InputChange += new InputChangeEventHandler(ifKit_InputChange);
        ifKit.OutputChange += new OutputChangeEventHandler(ifKit_OutputChange);
        ifKit.SensorChange += new SensorChangeEventHandler(ifKit_SensorChange);

        //Open the object for device connections
        ifKit.open();

        //Wait for an InterfaceKit phidget to be attached
        Console.WriteLine("Waiting for InterfaceKit to be attached...");
        ifKit.waitForAttachment();

        //Wait for user input so that we can wait and watch for some event data
        //from the phidget
        Console.WriteLine("Press any key to end...");
        Console.Read();

        //User input was rad so we'll terminate the program, so close the object
        ifKit.close();

        //set the object to null to get it out of memory
        ifKit = null;

        //If no expcetions where thrown at this point it is safe to terminate
        //the program
        Console.WriteLine("ok");
    }
```

PHIDGET22 UPGRADE

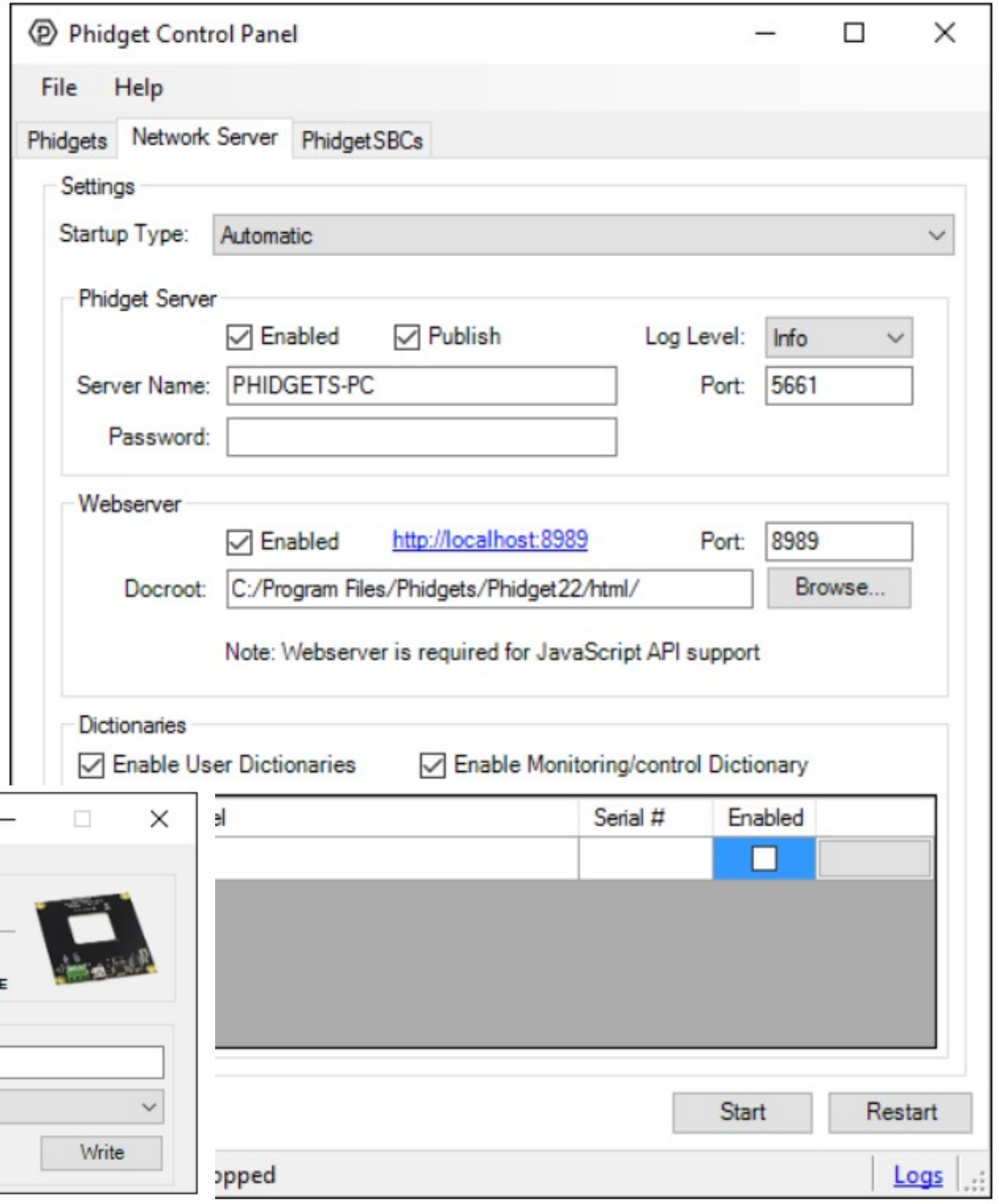


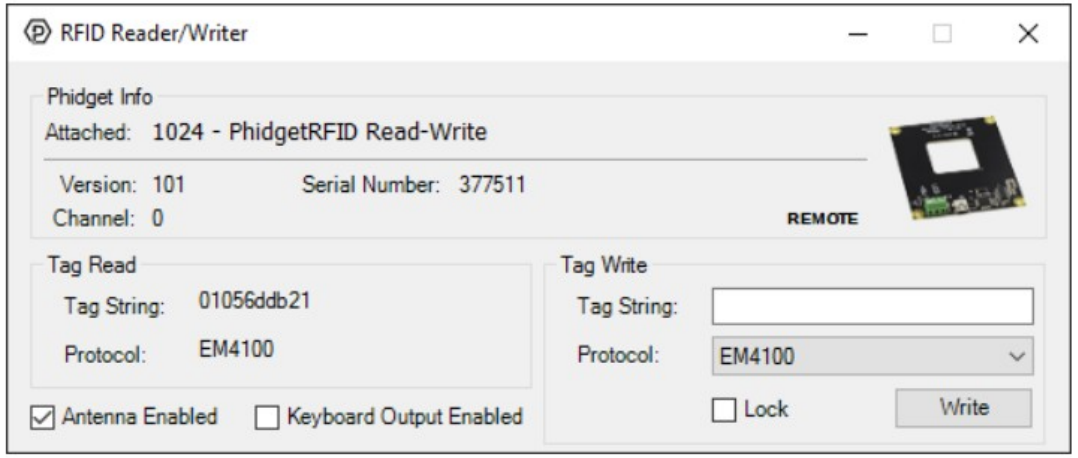


https://www.phidgets.com/docs/OS_-_Windows#Quick_Downloads

NETWORK SERVE

Turn your PC
into a network server
Turns your system into
a distributed system






AARHU
S
UNIVER
SITY