AARHUS
UNIVERSITET

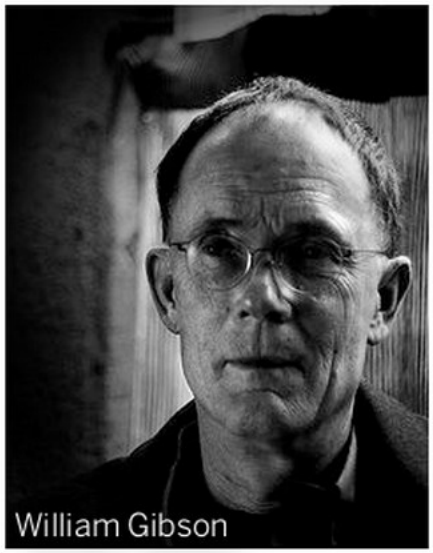
ADVANCED PERVASIVE COMPUTING

Lecture 3: Intelligent Environments and UbiComp User Interfaces

Stefan Wagner
sw@iha.dk

AGENDA

- Intelligent Environments
- Smart Cities
- Smart Homes
- Smart Spaces
- User interfaces
- AAL Smart Homes
- ADL
- IE Challenges
- Focus for Exam



"The future is already here, it's just not evenly distributed."

INTELLIGENT ENVIRONMENTS

Intelligent environments are spaces in which computation is seamlessly used to enhance ordinary activity (Steventon &Wright 2006)

One of the driving forces behind the emerging interest in highly interactive environments is to make computers not only genuine user-friendly but also essentially invisible to the user (Steventon &Wright 2006)

SHAPES AND SIZES …

- Smart Cities
- Smart Homes
- Smart Spaces
  - Factories
  - Pig pens
  - Offices
  - Desks
  - White boards
  - Patient beds
  - Healthcare measurement stations

- Ubicomp User Interfaces (UUI)
  - Supplement or replace existing interaction device types?

AARHUS
UNIVERSITET

SMART SPACES

AARHUS
UNIVERSITET

SMART SPACES

AARHUS
UNIVERSITET

ENABLING TECHNOLOGIES

AARHUS
UNIVERSITET

ENABLING TECHNOLOGIES

AARHUS
UNIVERSITET

ENABLING TECHNOLOGIES

AARHUS
UNIVERSITET

ENABLING TECHNOLOGIES

Activity tracking

UBICOMP USER INTERFACES

- Present day user interfaces inadequate
- - fixed computers with keyboards and mouse
- - smart phones and tablets also have their limitations

- Do you always carry your phone or tablet?
- - in the shower? … or to the toilet?
- - in the kitchen while cooking? To the pig pen?
- - while driving, eating, training, sleeping, relaxing?

- New class of alternative UI devices needed
- - smart watches, voice recognition, intelligent surfaces (walls, doors, tables)
- - face/eye tracking, hand gestures
- - implicit interaction (getting up from the couch or the bed will turn on the light), intelligent floor, PIR

- Challenges with new UI devices
- - neither developers nor users have experience, states multiplied by several magnitudes as compared to traditional computing
- - more field studies needed to understand the setting / device context over time and space
- - frequent, rapid, and longitudinal prototyping is needed to test the effects in changing situations
- - consider the need for multi modal interaction (allowing for several UI)

AARHUS
UNIVERSITET

NOVEL CLASSES OF UUI'S

- Tangible User Interfaces (TUI)
- Interact with physical objects, input and output
-
- Surface User Interface (SUI)
- Utilizes a surface to act as a display and/or input,
- Is either a screen or a projected image

- Ambient User Interfaces (AUI)
- Ambient technologies are ignorable or glanceable, speech, lights
- They are in the periphery of our attention (peripheral vision)

- Context-aware User Interfaces (CUI)
- Interfaces that reacts on context but are otherwise invisible
- Implicit interaction, medication cabinet opens when relevant

- Artificial Reality Interfaces (ARI)
- Produces an overlay of information on real objects
- Google glasses, projected image on milk with expiry date

EXERCISE 3A

› Create a Ubicomp User Interface

- › Must support Context Sensitive Reminder Dialogs – e.g. Medication Reminders
- › User arrives in the bathroom, remind user to take medication, if not taken within 2 minutes
- › Consider skipping the medication detection part for first prototype

› Use Context sensors to react to user presence

- › We shall use Phidget sensors to detect user presence and medication moved
- › Download drivers and source code from http://phidgets.com

› Use Artificial Speech Technology to interact with the user

- › We shall use System.Speech to interact with the user

› Time

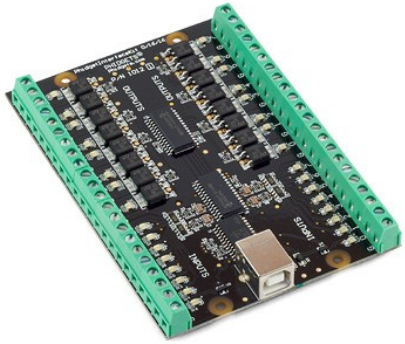
- › 1 hour (including break)

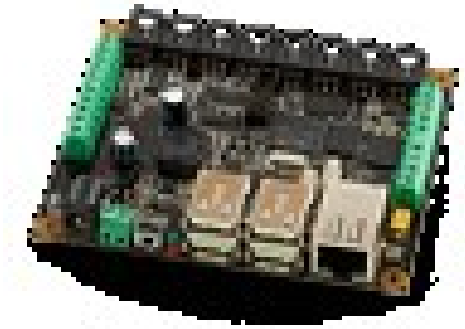
SYSTEM.SPEECH

```
using System;
using System.Speech.Synthesis;

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

      // Initialize a new instance of the SpeechSynthesizer.
      using (SpeechSynthesizer synth = new SpeechSynthesizer())
      {

        // Configure the audio output.
        synth.SetOutputToDefaultAudioDevice();

        // Speak a string synchronously.
        synth.Speak(“Good morning. Please remember to take your medication. Thank you!");
      }

      Console.WriteLine();
      Console.WriteLine("Press any key to exit...");
      Console.ReadKey();
    }
  }
}
```

PHIDGETS

- 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

AARHUS
UNIVERSITET

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");
    }
```