Teaching UbiComp with Sense

Conference Item

How to cite:


For guidance on citations see FAQs

© 2010 ACM
Version: Accepted Manuscript

Copyright and Moral Rights for the articles on this site are retained by the individual authors and/or other copyright owners. For more information on Open Research Online’s data policy on reuse of materials please consult the policies page.

oro.open.ac.uk
Teaching UbiComp with Sense

Mike Richards
Centre for Research in Computing
The Open University, Milton Keynes, UK
m.richards@open.ac.uk

Neil Smith
Centre for Research in Computing
The Open University, Milton Keynes, UK
n.smith@open.ac.uk

ABSTRACT
Modern computer science education has to take account of the recent changes in computing towards smart ubicomp devices. In addition, existing programming languages are needlessly difficult for novice programmer to learn concepts.

Author Keywords
Education, programming, graphical languages, ubiquitous computing.

ACM Classification Keywords (’pick one’)
D.3.2 Programming Language Classifications (Very high level languages).
K.3.2 Computer and Information Science Education.
H.5.2 User Interfaces.

INTRODUCTION
Computer science and technology teaching have suffered from declining enrollment across the UK over the last few years. Meanwhile, computing has changed massively with ubiquitous and highly connected devices becoming commonplace. The new module My Digital Life is intended to reinvigorate and widen participation in computer science and engineering. If computers are going to be everywhere and ‘everyware’, we should attempt to teach the subject to appeal to as large an audience as possible.

The Open University (OU) is Britain’s largest university with more than 250,000 active students. All OU undergraduate students study at a distance using self-study materials with extensive support from a tutor, termed associate lecturer (AL). Unlike most universities, the OU does not require any previous educational achievements as a condition of entry. Instead we offer a range of introductory courses that teach a basic grounding in a subject and key study skills. OU student results compare favourably with comparable students from conventional universities.

The OU has always been at the forefront of innovation. Not only was it Britain’s first distance education institution; but it went on to pioneer the use of educational television and radio programming through a long-standing relationship with the BBC. During the 1990s the OU trialed many of internet technologies, such as electronic delivery of materials, computer conferencing and computer marking, which are now commonplace in other universities [1].

RATIONALE FOR SENSE
Current first level OU computing students are taught JavaScript. This is a well-supported, relatively powerful, conventional language that has proved unsatisfactory. Whilst many students successfully develop JavaScript programs, a significant proportion of users either withdraw from study or do not progress to more advanced programming courses. Common complaints include JavaScript’s pedantic syntax, relatively poor debugging support and that the time spent learning the language was not reflected in the relatively simple and unengaging programs developed. Student interviews and feedback from ALs suggested that most students could design an algorithm to solve a problem, but lacked the confidence to turn that algorithm into an executable program.

The OU’s connections to the well-developed RoboFesta movement [3], and experience with a robotics course for novices Robotics and the Meaning of Life, [4][5] indicated another direction. Both of these used the LEGO Mindstorms™ kit that could be programmed using the drag-and-drop RCX Code environment. Children and adults found drag-and-drop programming extremely intuitive to use and were able to build relatively complex programs with rich behaviours.

For My Digital Life we decided to adopt and extend the Scratch [6] language from the Lifelong Kindergarten Group at the MIT media Lab. Scratch is a media-rich programming environment which is especially notable for its clear programming structure. Individual program blocks (e.g. if-else statements, logical operators and variables) can only be assembled in meaningful (not necessarily correct) ways, and this remove ones of the major frustrations of JavaScript: syntax and logic in block-based programming becomes explicit rather than implicit.
Although Scratch is not a language in industrial use, it gives
students all of the basic skills needed to succeed in most
common programming languages. It also builds students'
confidence in their own abilities. Unlike most languages,
with Scratch ‘you don’t need to know a lot to do a lot’. Scratch has proved extremely popular with educators and
students alike.

Scratch was not ideal for our purposes. First, it has clearly
been designed for children and we were concerned that
adults may find it patronizing. However, when we tested
Scratch with adults we were surprised by their enthusiasm. Scratch’s soft shapes and bright palette were welcoming and
the toy-like appearance suggested that the program was
every bit as resilient as physical children's toys. Second, Scratch lacked some of the richer programming concepts
(such as lists) required by the computing curriculum. Finally, Scratch could not talk to the outside world. If we
were going to use it to teach ubiquitous computing, Scratch
was going to need network support.

Fortunately, Scratch can be modified. Over the last two
years we have been building our own programming
environment, Sense, which is more suited to adult learners
exploring ubiquitous computing. Sense is an open-source
project and it will be released under a suitable licence. We
are in discussion with the Scratch community about
merging some of Sense's extensions back into the Scratch
code base. Sense runs on Windows, OSX, and Linux.

EXTENSIONS
Sense extends Scratch in three main areas: the Sense board
allows diverse input and output; support for reading and
writing data over the internet allows collaboration; and lists
allow more complex programs to be developed.

Sense board
The core of the Sense extension is the SenseBoard (Figure
1), a tethered device based around the Arduino
microcontroller. The concept of a programmable hardware
device for novices is not new; the PicoBoard [10] was
developed precisely for this purpose and can be used with
Scratch. However, the PicoBoard has no outputs (LEDs or
motors) which limits its use for designing ubiquitous
devices and is relatively expensive in the UK.

An alternative was to use Phidgets [9], which are widely
used in ubiquitous applications and electronics courses.
However, their high unit cost and intimidating ‘breadboard'
appearance precluded their use with novice students.

The design of the SenseBoard was a collaboration between
the Open University and an external consultant. The aim
was broadly to replicate the functionality of the PicoBoard
as well as adding output devices. The Sense board had to be
sufficiently robust to survive postage, transport and
repeated use in a home environment with minimal technical
support, yet cheap enough that it need not be returned to the
OU. After several iterations of the design, the final cost of a
SenseBoard, as well as plug-in sensors, actuators, and case
is approximately US$50 when ordered in bulk.

The SenseBoard has a number of devices on board, including
a slider, a pushbutton switch and a bank of 6
LEDs. Additional sensors, such as a thermistor and a
motion detector, can be plugged in, as can a stepper motor
and a battery pack (needed when using the motors). The
board connects to a PC via a USB cable.

Internet access
A key feature for developing ubiquitous devices is their
ability to communicate with the outside world. We added a
number of features to Sense to allow it to read and process
data from the internet. Sense is able to read the contents of
web pages and place the HTML in a variable for further
processing. Support for RSS is more extensive: specific
blocks allow RSS feeds to be read, the items processed
individually, and the elements of each item read (Figure 2).
This allows for sophisticated processing of data and the use
of RSS data to act as an input to ubiquitous devices. For instance, the weather clock (described below) uses a weather forecast RSS feed to display the predicted weather on a clock face.

Potentially more powerful is the ability of Sense to push data to a dedicated server, which can be read back as an RSS feed. This allows students to both store data ‘in the cloud’ for later use, but also to programmatically interact with each other by using a common RSS feed for reading and writing data. The server also provides the same functionality via HTML forms, so students can interact with, monitor, and test their systems with a web browser, which gives them a clearer view of how their program is working. In this way, Sense supports both interactive and social computing. This feature has been used to implement presence indicators for students and develop a quick Twitter-like status update alert system.

A specific feature of this system is that all students have full read-write access to all feeds, and that any student can create any feed at any time. We decided that this flexibility would prompt students to explore novel ways of using this feature, without having to ask for permission. Time will tell if additional control over feeds is needed. The server is behind an access-control mechanism and is only accessible to students enrolled on the module.

Richer types: lists
The richer feature set of Sense, and the pedagogic demands of an introductory Computer Science module, required students be exposed to and use more complex data structures than the scalars of numbers and strings. Hence lists were introduced, complete with a set of blocks to manipulate them, such as inserting, deleting, and selecting items.

PEDAGOGY
Students will engage with Sense through a large number of activities, most of which last less than an hour. The activities are designed to reinforce the learning found in the other course materials. The activities are designed to be relevant and useful to the students, allowing them to show how smart devices and ubiquitous technology can be used in authentic situations. The sample activities are tightly constrained and come with sample solutions. Students can choose to complete an activity without assistance, or use one of a number of methods of support:

- Step-by-step instructions on completing the activity;
- Partial programs: complex or tedious parts of the program are provided by the OU;
- Exploded programs: the program is broken into individual blocks, the student must assemble them in the correct order;
- Sample solutions: where the complete program is available for examination;
- Screencasts: a narrated video showing how to complete an activity.

Activities constitute teaching material rather than assessment. Student progress is regularly monitored using formal, marked assessment for which no assistance is provided.

An ideal solution is provided by the course for each activity. If a student has failed to complete the current activity for any reason (most commonly lack of time rather than lack of skill) they can begin the next activity using the provided solution, rather than spending yet more time trying to catch up with their peers.

All activities come with suggestions for extension and improvement, which will guide interested students to further develop their skills. The final assessment point in the module is an extended piece of coursework, one part of which is the development of a smart device to solve a problem in the student’s own situation. A prize for the best project will be awarded in each presentation of the module.

EXAMPLES
We now present some sample activities to show how Sense is used in the module to develop ubiquitous computing devices. As well as the examples shown here, students develop simple games, cryptographic programs, and musical ‘instruments.’

Cyber coaster
A simple application of the environment sensors on the SenseBoard is a smart drinks mat, termed ‘the cyber coaster’. The objective is to create a device that reminds the user that they’ve forgotten their tea (or coffee) and it’s in danger of getting cold. The temperature sensor, supplied on a lead, is threaded through a hole made in a coaster so that it will be in contact with a mug placed on top. A Sense program regularly polls the temperature sensor and uses a list to keep a record of its temperature over time. If the temperature is falling slowly and drops below a specified threshold temperature, Sense plays a sound and pops up an alert dialog to remind the user about their tea. Sudden temperature changes indicate that the cup has been lifted, so no reminder is necessary.

Presence and Status
The RSS push and read features have been used to develop simple devices that mediate and prompt social interaction. Students in a tutor group are allocated to smaller groups of six students and each student is allocated one of the six LEDs on the SenseBoard. A shared RSS feed holds the status of the students. The push button on the board is used to toggle the student’s presence indicator to the rest of their group, indicating whether they are studying and available for interaction. When the button is pushed, Sense sends the
updated presence information to the shared RSS feed. Each
student's presence program periodically polls the RSS feed
and sets the LEDs on the board to reflect the reported
presence status of the students.

An extension activity to this project involves the students
making their own pressure sensor, using the resistance
meter input on the SenseBoard, to detect when they are
sitting in their chair and to automatically update their
presence.

Weather and whereabouts clock
The stepper motor that comes with the sense board can be
used to create large, highly visible displays by moving a
clock hand. Two applications of this are the weather clock
and the whereabouts clock.

The weather clock reads weather forecast data from a
public RSS feed (currently, the localised weather from the
BBC) and parses it to extract the predicted weather
conditions for the next day. Once the weather type is
detected, a weather symbol is displayed on the Sense stage
area. The program also moves the motor so that an attached
hand moves to the correct position on a clock-face like
display. The display is printed by the students on a simple
template.

A similar idea underlies the whereabouts clock, inspired by
the Harry Potter books. Again, each group of students
shares their own RSS feed. As each user changes their
status (at work, unavailable, etc.), they press the relevant
button displayed on the Sense stage and that status change
is posted to the shared RSS feed. As with the presence
notifier, each user's program periodically polls the RSS feed
and indicates each person's status using a moving pointer.
As the SenseBoard comes with only one motor, the physical
display is split into two, with one section indicating people
and the other their status. The pointer moves to a person,
then their status, then the next person, and so on.

Weather station
British people are obsessed about the weather and it was
only natural that the SenseBoard's light and temperature
sensors would be used to develop a simple weather station.
This device uses all the extensions developed for Sense, as
well as additional features such as presenting data in a
Google Maps mash-up (developed by the OU).

The weather station is one of the first ubiquitous devices
developed by students. Novice students may find this an
extremely challenging build, so the construction is broken
into a number of separate activities.

The final development of the weather station is to add
location information and publish the data over the Internet.
Since we cannot guarantee every student will possess a
computer that can determine its own location, position
information is obtained by the student entering their post
(ZIP) code or geographical coordinates. The location
information is appended to their readings and published on
an RSS feed.

The same OU server which handles student feeds extracts
weather readings and location information and publishes
the results on a Google Map. Eventually, the map will fill
with data from thousands of students as they complete the
activity and begin to publish data.

Although the student weather stations are relatively crude,
the intent is not to produce accurate maps so much as to
demonstrate the potential of ubiquitous technologies. A
corresponding benefit is that the activity serves to build a
community of contributors from a disparate, widely-
scattered population of students.

Polling
Students are expected to develop skills in evaluating and
summarising conflicting sources of information. The course
contains an activity centred on a televised debate about the
role of intellectual property in the information age.

Students are expected to record their opinions of the debate
speakers using a simple handheld device that registers
agreement or disagreement at regular intervals. Students
construct a simple application that records the value of the
SenseBoard's slider and posts changes to a central RSS
feed. As the debate progresses, the student moves the slider
to reflect their opinion. Another Sense program downloads
the contents of the RSS feed and writes them to a local
spreadsheet file. The student can examine the file and see
their changing responses on a timeline, either individually,
or compared to their cohort.

RECEPTION
My Digital Life is still in production and has not yet been
released to students. However, we have tested the teaching
approach and Sense with a variety of representative learners
and experience ALs The response has been uniformly
positive and enthusiastic. Even programming novices can
quickly develop programs and devices. Feedback has
prompted us to strengthen support for debugging Sense
programs, such as including a variable-speed stepper to
show how the program proceeds, step by step.

CONCLUSIONS
TU100 is a typically ambitious project for the Open
University and has required the development of a number
of new technologies to address the particular needs of
distance learning students. We have developed a
programming environment for adult learners with no prior
experience of computer programming, and a
complementary piece of hardware that will allow them to
begin experimenting with ubiquitous computing.

A particular challenge has been the development of
activities that are sufficiently straightforward that it is
realistic to expect novices to complete them, and that demonstrate genuine principles in ubiquitous computing.

REFERENCES


3. RoboFesta Europe (Britain) http://www.robofesta-europe.org/britain/


The columns on the last page should be of approximately equal length.