Android Plant: the green robot revolution.

Francesca Madeddu
Computer Science MSc
Swansea University
SA2 8PP, UK
803623@swansea.ac.uk

Hua Cai
Computer Science MSc
Swansea University
SA2 8PP, UK
@swansea.ac.uk

Michael Waterworth
Computer Science MSc
Swansea University
SA2 8PP, UK
805956@swansea.ac.uk

Abstract

In this paper we describe a system for monitoring environmental conditions of a plant which allows monitoring of its health. The system uses Android and Phidgets to monitor a plant in-situ throughout the day. It provides a novel use for old mobile phones and tablets.

Introduction

We were asked to prototype a project using hardware controls - specifically Phdigets - with the details of the project being chosen by us in an initial project specification. We chose to implement a simple system of monitoring environmental conditions around a plant. This project was interesting to us as one of the group members keeps house plants but spends too much time at university so a simple monitoring system that could alert him/her to any issues had the possibility of solving a real-world problem. This builds on the research by Sandberg and Kokholm with iPlant [2] which details a system for making a plant self-sustaining by monitoring its own conditions, watering and feeding as necessary. Our system is an implementation of a subset of this functionality, just of monitoring, and proves that modern mobile phones can easily be put to use as monitoring devices.

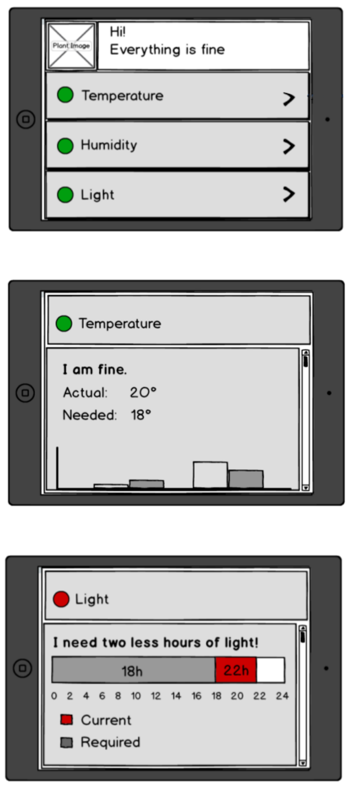


Figure 1: Figures show our initial idea for the Android application: a main activity which gives general information about the plant and from where each sensor detail section can be accessed.

Related works

As a intelligent device which is implemented in many aspects of daily life and research, sensors such as temperature sensor, humidity sensor , light sensor and its potential application has provided many inspirations for our project.
As is discussed by W Xue in 2009, The intelligent sensor and its networking could provide a flexible control for the plant disease. Their work is to use the wireless communication approach to realize their functions. But due to our limited project devices and background knowledge, it is a more feasible approach for us to realize the communication module by using cable. In their work, they designed both a data acquisition module and the control module. However, in our project, due to the reason that we do not have to control some devices, we have designed a data acquisition module and a data demonstration module.
In W Xue's paper, they use the DSP as their core digital processing devices which is not available for us. Because of this, we use more simpler and available device- phidgets devices as our project platform.
Another paper which provide us a inspiration is W S Lee's work in 2010, which is a overview of sensing technology in crop growth. In their project, they discussed the sensor application to detect the root information by image sensor which is not available for us. Because of this, we attempt to use humidity sensor to provide a basic information about the humidity of root which is important for a plant to grow. These two papers [1] [3] are all from UbiComp conference.

The final application

Requirements and Specification
According to the proposal defined at the beginning at the project, we needed to build a system which allows to look after a plant by checking at the current environmental variables such as temperature, humidity and received light and by displaying the monitored variables through a graphical interface.
Therefore our system consists in a Phidget station, composed by a temperature and humidity sensor, and a light sensor. The Phidget station can be directly connected to an Android mobile phone, where a custom developed application shows on a screen the current retrieved values. The Android application also allows to define a range of values within the current values should belong to. Notifications are sent by the application if the plant needs attention, that is the current value for a sensor is not between the range previously defined by the user.

Development Technologies
A number of different software technologies were employed in developing the prototype. These include:

- Balsamiq[1] was used during the earlier stages of the project to sketch mock-ups of possible graphical interfaces (figure 1). Baslamiq, in its trial version, offered all functionalities we needed to prototype our application, and it was also easy to learn and flexible to use.
- Android Studio[2] was the IDE / SDK used to develop the application across multiple platforms. It was chosen as it's custom-designed to work well with the Android platform, unlike the older Eclipse plug-in architecture. Android studio was used across all three major operating system architectures that we were developing on.
- The Phidget interface library[3] was integrated in the Android Studio project to allow the Android phone and the Phidget interface kit to directly communicate.

[1] https://balsamiq.com/.
[2] https://github.com/.
[3] http://www.phidgets.com/docs/OS_-_Android.

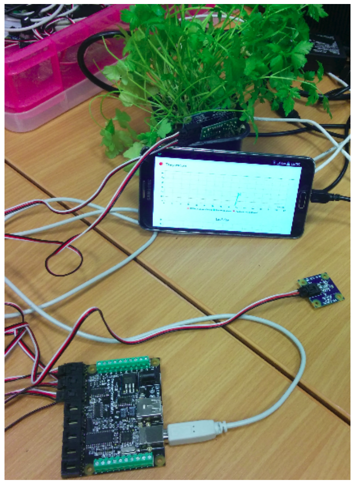


Figure 2: The complete system: the Phidget interface kit with sensors, and the Android application in action.

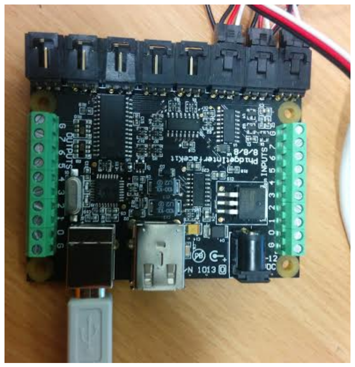


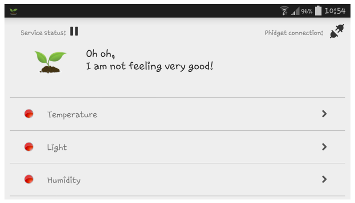
Figure 3: The Phidget interface kit.

- MPAndroidChart[4] was used to draw the graphs displaying the information acquired from sensors. It was chosen after a careful comparison with other libraries because of its great flexibility in terms of customization and its big amount of features.
- Ghost Log[5] was used during the last stage of the development of the Android application for debugging purpose. In fact, because the mobile phone was directly connected to the Phidget, the debugging was made very difficult, since it was impossible to read the logcat buffer directly. The Gost Log application has proven to be a good solution since it allowed to print the application long in a system overlay window during the execution of the application.
- Git / GitHub[6] was used on development machines to synchronise code and to version control it effectively. The source run in to dozens of files and so having a version control system was very useful. GitHub was used to synchronise these repositories using pull requests.
- Dropbox[7] was used to synchronise files that weren't code. It was also used to share binary files like PDF and word documents to allow simple project management.

To build the current system we used also several hardware components:

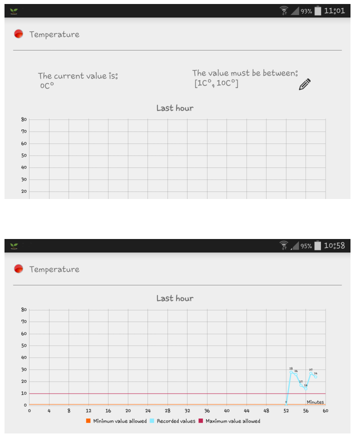
- Phidget Interface Kit 8/8/8[8]: the interface kit which was used to connect three sensors to the Android phone using a USB cable.
- Phidget 1127 - Precision Light Sensor[9]: a precision light sensor was used to retrieve the amount of light was absorbed by the plant. The sensor is capable to measure approximately between 0 and 1000 lux, allowing to distinguish between the different environment condition such as if the plant is exposed to direct sunlight or not. The light sensor was attached close to the plant, specifically on its pot surface, to guarantee precision; also attention was paid so that the plant did not cover the sensor with its leaves, making the measured values incorrect.
- Phidget 1125 - Humidity / Temperature Sensor[10]: a combined temperature and humidity sensor was used to discern between the different enviromental conditions in terms of temperature and humidity, that is for example the plant is receiving enough water. The two sensors are capable of measuring respectively values between the range of 10%RH[11] and 95%RH, and -30C°and 80C°. To make the measurement more precise we needed to put the sensors into the soil, to that we had to design and develop a protector case made with plastic, which allowed the air to pass through but protected it by soil and water.
- An Samsung Note 3 has been used as main testing device. However the application runs on any Android

[4]https://github.com/.
[5]https://github.com/jgilfelt/GhostLog.
[6]https://github.com/.
[7]https://www.dropbox.com/.
[8]http://www.phidgets.com/products.php?product_id=1018.
[9]http://www.phidgets.com/products.php?product_id=1127.
[10]http://www.phidgets.com/products.php?product_id=1125.
[11]Relative humidity.



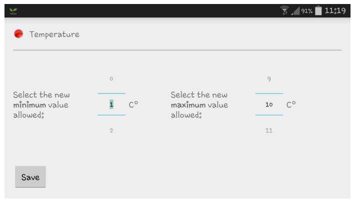
Oh oh,
I am not feeling very good!

Figure 4: The main screen of the final application.



The current value is:
0C°
The value must be between:
[1C°, 10C°]

Figure 5: The sensor details activity of the final application.



Select the new **minimum** value allowed: 1 C°
Select the new **maximum** value allowed: 10 C°

Figure 6: The edit range values activity of the final application.

smartphone above Android 2.3.3[12] which support Android USB OTG [13].

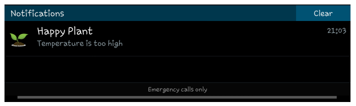
- A female USB-A to micro USB cable and a USB-A to USB-B cables were used to connect the Android phone to the Phidget interface kit.

Delivered Prototype
The Android application needs to be uploaded to the device as it is not available on any of the Android App Stores. This can be done from USB an by side-loading the application using a file explorer. Once the application is installed on the device it is started by tapping the icon. Once open the application will show that the service isn't running and that the Fidgets aren't connected. To start the service a user taps in the top right on the play button. This starts the service which collects data and allows interfacing with the Phidgets. Once started it will attempt to connect to the Phidget Controller board. When it sees this the Android system will prompt the user to allow hardware access to the application [14]
The application will then start to display live data coming in from the device

Appearance and Implemented functionalities
The mobile phone application is composed by three different activities as explained in the following paragraphs.
The main activity (figure 4) gives a panoramic view about the general condition of the plant. The status of each sensor is made clearly visible using a led images which turns green or red if the current values of the specific sensor is comprised in the user defined range or not. The general status of the plant is shown on the top, where another led image turns green only when all sensor are green too. Also a generic explicative message is given. On the top of the main activity are also displayed two icons: on the top left is the service icon, which shows if the service is actually running, that is the application is retrieving the values from the sensors, saving them in the database, and updating the application graphical interface with the new values; on the top right another icon shows if the interface kit is connected to the Android phone or not.
The second activity (figure 5) presents details regarding a specific sensor: it shows the last acquired values, the minimum and the maximum values previously set by the user, and two graphs displaying values acquired during the last hour, with a minutes granularity, and the last day, with a hours granularity. Both graphs can be expanded or reduced by pinching and they display values acquired in the last hour (or the last day) and they are constantly updated with the new values. Also two additional function are drawn in each graph to show the minimum and the maximum threshold the current values should be comprised in. The values acquired from sensor are row, so that before being displayed they need to be converted[15] and filtered[16].
The third activity (figure 6) is accessible from the second activity by touching the edit icon next to the required values, and it allows to modify the range of the required values for the current sensor. Values can be chosen by interacting with a number wheel where values are limited according to the physical limitation of the Phidget sensor as detailed in the *Development Technologies* section. The numbers

[12]Corresponds to API Level: 10.
[13]USB On The Go OTG. This is supported on many tablets and an increasing number of phones.
[14] Required every time hardware access is requested.

[15]Values were converted according to the specific sensor specification, as detailed of the cited website, except for the light sensor which already gave the right values.
[16]All values were filtered using the following low pass filter: (New value) = (Last value) + xi * a - (Last value) * a, with a=0.01.


Notifications
Clear
Happy Plant
Temperature is too high
21:03
Emergency calls only

wheel also implements some constraints to make sure that the user cannot actually chose the wrong values[17].

Finally a notification system is created to notify the user that the plant needs attention. When any of the sensor spots that the current value is not in the user defined range, a notification pops up on the notification manager with a customized message for each sensors (figure).

Limitations of the delivered prototype
The delivered prototype provides a reasonable solution to the proposal designed, but it does have a couple of issues that would have to be overcome before the next iteration is created. These include the limitation of a mobile device to charge whilst in USB OTG mode. As our prototype is designed to be run whilst permanently connected this would give a maximum amount of time it could run before the battery fully discharges and the user would have to unplug and charge their device. This could be solved using a number of solutions, including inductive charging of the device 4 or by moving the Phidget interface to using Bluetooth.

Roles
As can be expected the work carried out was split in to a number of tasks that could be accomplished separately. This worked out well because of the limited number of practicals available. We came up with the plant idea as a group and before starting to implement it continuing to develop the idea in to a concrete project specification. From here development of the application could continue at a rapid pace, with the report documentation tapering in near the end. A detailed breakdown of tasks and roles in presented in the following table:

[17]For example the user cannot input a minimum values which is bigger than the maximum value.

Task	H	F	M
Develop concept in to project plan		√	√
Create assets for application		√	√
Create a prototype application		√	
Create the graphical interface		√	
Investigate project feasibility		√	√
Decide on key technology to be used	√	√	√
Investigate graphing libraries	√	√	√
Choose graphing library		√	√
Create graphing code		√	
Write up sections of the report	√	√	√
Design holder for the sensors	√	√	√
Create database and associated classes			√
Create service for application			√
Implementing communication between activities		√	√
Create notifications for application			√

Table 1: Task achieved by person where H = Hua Cai, F = Francesca Madeddu and M = Michael Waterworth

In particular, for the report, each section title is marked with a footnote which mention the initial letter of the contributor name.

Conclusion
In this paper we presented a system which, combining a simple Android mobile phone with few Phidget sensors, allows to look after a plant by monitoring its basic environmental variables such as received light, humidity and

temperature. At the present time the developed system is limited by some constraints: some of them are easy to fix but for some others a more structural change is needed. For example, as mentioned before, because the mobile phone is directly connected to the Phidget interface, there is no way to charge the phone except with a wireless charger. Or again, even if we are storing values without any limit of time, we are only displaying information about values recorded during the last day, with a specific attention to the last hour: however, because the values are actually stored, it would be easy to implement new functionalities to allow an user to manage data from days or even month of recordings. Also, notifications to make the user aware that the plant needs attention are actually displayed on the device itself, but would be easy to implement addition features to send a text, or an e-mail to notify the user even when she/he is in a different location. Finally the application is actually monitoring only one plant, mainly because of the lack in the number of sensors which were available, but would be easy to extend the project to allow monitoring more than one plant. One of the constraint which would be less immediate to address would be a more automatic management of the plant, that is to build a system to cover the plant if it is receiving too much light, or to automatically water it when the humidity level is too low. However achieving this purpose lies outside the prefixed purpose of this project. Moreover we believe that people buy plant because they like the idea of looking after them, so that what they need is a support to remind them what the plant needs, more than something which completely automate the job for them.

References

[1] Kuznetsov, S., Odom, W., Pierce, J., and Paulos, E. Nurturing natural sensors. In *Proceedings of the 13th international conference on Ubiquitous computing*, ACM (2011), 227–236.

[2] Sandberg, J., and Kokholm, T. iplant: Inteligent plant system.

[3] Wang, X., Zhang, C., and Yang, S. Design of wireless video communication system used to monitor and control plant disease. In *Networks Security, Wireless Communications and Trusted Computing, 2009. NSWCTC'09. International Conference on*, vol. 2, IEEE (2009), 584–587.