Info@ITU

Egil Hansen
IT University of Copenhagen
ekri@itu.dk

Jonas Rune Jensen
IT University of Copenhagen
jrur@itu.dk

INFO@ITU

We decided to base our Info@ITU system on three main components, a Info@ITU Monitor (monitor), which tracks the user while inside ITU and displays information to the user when the user is close to a display, a Info@ITU Proxy (proxy) which sits between the monitor and the Android devices and keeps track of the devices that entering and leaving ITU, and an Info@ITU App for tracking outside of ITU. This provices loose coupling between the components and in theory will allow us to scale up more easily.

Info@ITU Proxy

The proxy provides the following three service methods for the phones to use when interacting with the Info@ITU system:

- `entering?<Phones Bluetooth MAC address>` This method is used by the phone to tell the system that the phone is moving into ITU and to start tracking the it.
- `leaving?<Phones Bluetooth MAC address>` This method is used by the phone to tell the system that the phone has left ITU and to stop tracking it.
- `ping?<Phones Bluetooth MAC address>` This method should be used every XX minutes, after the phone has entered into ITU, to indicate to the system that the phone is still turned on and inside ITU.

The reason why we added the `ping` method is to handle the common scenario where the phone is out of range of ITU's BLIP sensors for a longe time, i.e. when the owner is in a meeting room or lecture hall, or if the owner has turned of his phone or the phone has simply run out of power. If the proxy stops receiving pings from the phone, we assume that the phone should no longer be tracked and can save resources on the monitor.

The proxy, which is runs on Google's App Engine cloud service, also requires the user to log in with his Google ID when starting the Info@ITU App on his phone. That helps us tie a real person to a Bluetooth MAC address.

Future Work

A related feature to the `ping` method, which is not yet implemented, is a scheduled clean-up job, that searches through the list of 'checked in' phones on the proxy and removes those who have not 'pinged in' for XX minutes.

Push notificaitons from the proxy to the monitor would also be natural extension, since it would lower the resources required and allow the monitor to be notified almost instantly when a new phone 'checks in'. In the current solution, the monitor pulls from the server every XX seconds to see if there are any new phones or phones that have left.

Info@ITU App

For our Info@ITU App we wanted to solve the issue of identification (who is actually using the Android device), battery life, volatile state, and privacy.

Identification

This is an issue even if mobile phones are often very personal, since they could be shared between people, or more likely, a person could have a private profile and a public profile. So to identify the user of the phone, we ask the user to select one of his Google accounts when signing in to Info@ITU from his device.

Battery life

This can quickly become an issue when both WiFi, GPS and Bluetooth radios are turned on. So we try to be as effective as possible, only turning on Bluetooth when the device is actually inside ITU.

Volatile state

The volatile state of mobile devices must also be considered – the user can choose to turn it of, the device can run out of battery, the user can turn of Bluetooth or GPS, other applications can turn of Bluetooth or GPS, all things we need to consider when building an app like this that relies on these.

Considerations for the app:

- We require users to log in with (one of) their Google accounts - Dont turn on Bluetooth discoverable mode before entering ITU (Battery and privacy) - Be unobtrusive when enabling/disabling the Bluetooth/GPS radios. Other Apps

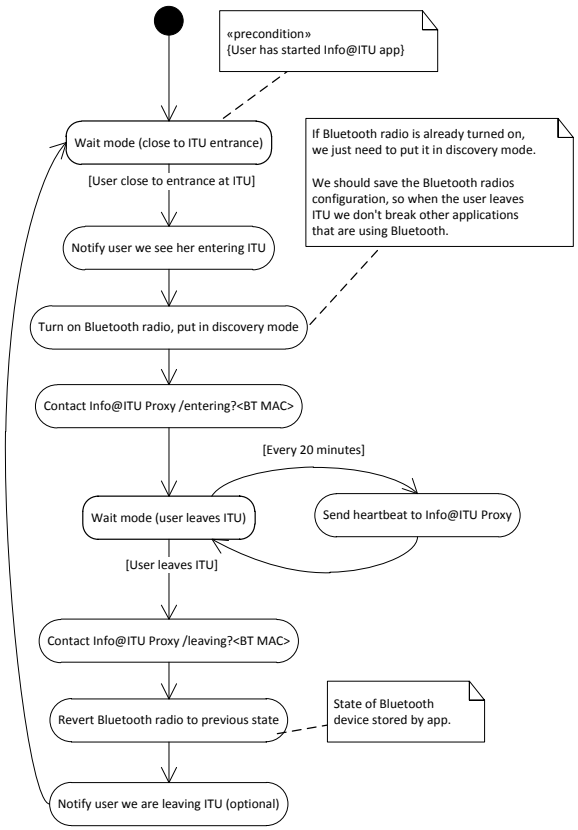

Figure 1. Activity diagram of the main service in the Info@ITU App.

might be using those radios so we should log the state and not - Using vibration, sound, and notification popups when entering and leaving ITU, so user knows we are turining on/off active tracking. - Handling normal conditions such as phone running out of batter, being turned of or loosing network connectivity (reliability)

The workflow of our Android App is illustrated in figure 1.

REFERENCES