email: egonzale@vub.ac.be
office: 10F736

5 Distributed Programming in Android: weScribble

Ambienttalk's tutorial and language reference are available at http://soft.vub.ac.be/amop/.
The lab session material is available at pointcarre andhttp://soft.vub.ac.be/amop/teaching/dmpp

5.1 Idea

The purpose of this exercise is to get you familiarized with distributed programming in AmbientTalk and its symbiosis with Java. To this end we will implement the weScribble application, a simple collaborative finger painting application [1]. The idea is to draw with one finger on a canvas. When other people running weScribble come into communication range, they can simultaneously draw on the canvas which is *virtually* shared between the different participants. Each user has its own color which can be changed at will. If the drawing sucks, just reset the canvas, and let the fun start again.

5.2 Implementing weScribble

We will implement weScribble starting from the skeleton code shown below (available in the session project under assets/atlib/demo/weScribble/weScribble.at) :

```
def makeWeScribble( myUserId := /.at.support.util.randomNumberBetween(0, 125)) {

  def otherPainters; // stores remote painters.
  def otherPaths; // stores the paths of remote users

  //interface for android gui.
  def localInterface := object: {
    //TODO
  };
  //interface for remote painters.
  def remoteInterface := object: {
 //TODO
  }; //end-remoteInterface

  def goOnline() {
   // setup peer-to-peer service discovery
  };
  goOnline();
  log("Hallo Android!");
};
self;
```

makeWeScribble is called by the main activity (WeScribble.java) after launching AmbientTalk. It takes as parameter the user identifier. To keep the exercise simple, users are identified with a random number. makeWeScribble consists of the local

[1] A demo of the weScribble application is available at http://www.youtube.com/watch?v=k0HYqRCxtHc

interface to communicate with the activity for GUI purposes, and the remote interface to communicate with other remote painters. Use the above skeleton to incrementally implement weScribble as follows:

(a) Complete the skeleton code to provide the following basic functionality:

  (a1) Complete the `goOnline()` method to be able to discover other painters in the network. As in the demo, assume that all painters have an empty canvas when they start collaboratively drawing.

  (a2) When a user draws on his canvas, his path should be also drawn in the canvas of other remote painters.

  (a3) If a user changes the color by means of the `Color` option menu, from then on, both the local and remote canvas will display his drawings with the new color.

  (a4) If a user resets the drawing by means of the `Reset` option menu, all remote canvas should be also erased.

  The `edu.vub.at.weScribble.interfaces` package contains three interfaces defining the necessary methods for the interactions between the Android GUI and the AmbientTalk application.

(b) Adapt your implementation so that the application displays the drawings made by a remote painter before the user joined the drawing session.

(c) Adapt your implementation to gray out drawing of remote peers which disconnect from the network.

  In order to easily test your code, you can use `Network Off/On` option is provided in the option menu. When the user first tabs on the option, the `disconnect()` method is called on the AmbientTalk object implementing the local interface. When the user tabs again the option, the `reconnect()` method is called, the next time the `disconnect()` method is called, and so on.

  First, you will need to uncomment the two last methods in the `ATWeScribble` interface and lines 212-220 in `WeScribble.java`. Then, adapt your local interface object to implement the necessary code to simulate network disconnections from your application.

(d) If you get here, it is time for fun! Try to have the largest collaborative drawing with your classmates. You may need to adapt your remote interface so that your weScribble application can talk to your neigbhour weScribble implementation.