1. Using Fragments allows us to handle display diversity by reusing our code. It allows us to display different versions of the application depending on the type of display (phone, tablet, etc.) and orientation (portrait and landscape).
   For handling different orientations, we can place the different XML layouts in different folders in the layouts folder. Each folder will be named accordingly (layouts-port or landouts-land).
   For the tablet, we could have the Settings and the Buddies views together, and the Show and Map views together, instead of just displaying one activity at a time.
2. Some options we could use:
   - List and Grid List: We could have a list of the four activities and ask the user to pick anything in the list.
   - Tabs: We could have the four activities as different tabs on the top of the activity, and the user would successively choose from these four tabs.

   Both of the above two interfaces would be equally suitable as they allow the user to switch between the activities he or she requires without having to go through the activities that he or she does not wish to see.

   - Horizontal Paging The user could swipe on the screen (left or right) to move back and forth among the screens. This would be less suitable as the user does not necessarily have to go through all the screens while using the application.
3. The Buddies and Settings page have only one thread:
   - The Main UI thread controls all the operations in these threads since they are relatively quick and will not make the UI unresponsive.

   The Maps and Show pages however have multiple threads. They have a common structure:
   - A Main UI thread that displays the form in Show and the MapView in Maps.
   - Once the start button is clicked, a Handler produces a message once every time interval that fetches updates from the server, fetches updates from the GPS unit on the device and calls the Handler to update the User Interface. This ensures that the UI is responsive at all times. Once the Stop button is clicked, the Handler is destroyed to stop the update process.
4. To use GCM, we would have to do the following steps:
   - Register the API key.
   - Include the permissions for the GCM service.
   - Install the GCM libraries.
   - Write the application to include the references to the GCM whenever we wanted to contact the server. This would involve using an Intent to send the information.
   - Set up a Broadcast receiver to receive the GCM message. The payload can contain all the information we need (Since we will need less than 4 kB of information.)
   - Write the servlet to contact the GCM servers form the server side.

   We should use GCM rather than direct polling because it will optimise power consumption, reduce bandwidth consumption as the Google servers are highly optimised, and will reduce the size of our application.

5. ACTION_DIAL tel:123
   We need to send an Implicit Intent with the action ACTION_DIAL and the Telephone number expressed as a URI. This will call any application that has a broadcast receiver set up to handle this kind of Intent, and will dial the number that we sent through the URI.