Design Patterns

MSc in Computer Science

Produced by

Eamonn de Leastar (edeleastar@wit.ie)

Department of Computing, Maths & Physics
Waterford Institute of Technology

http://www.wit.ie

http://elearning.wit.ie

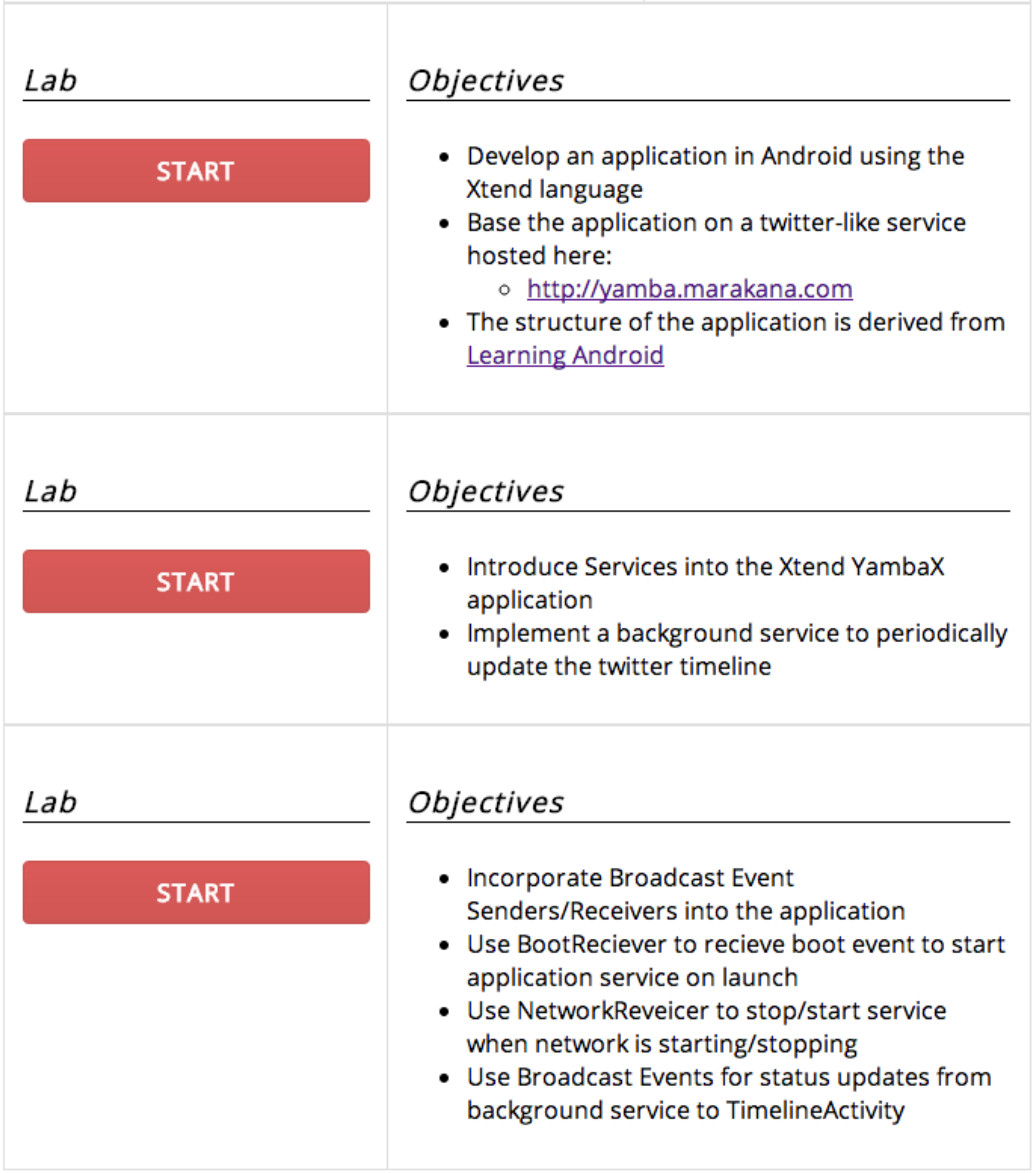
Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

eLearning support unit

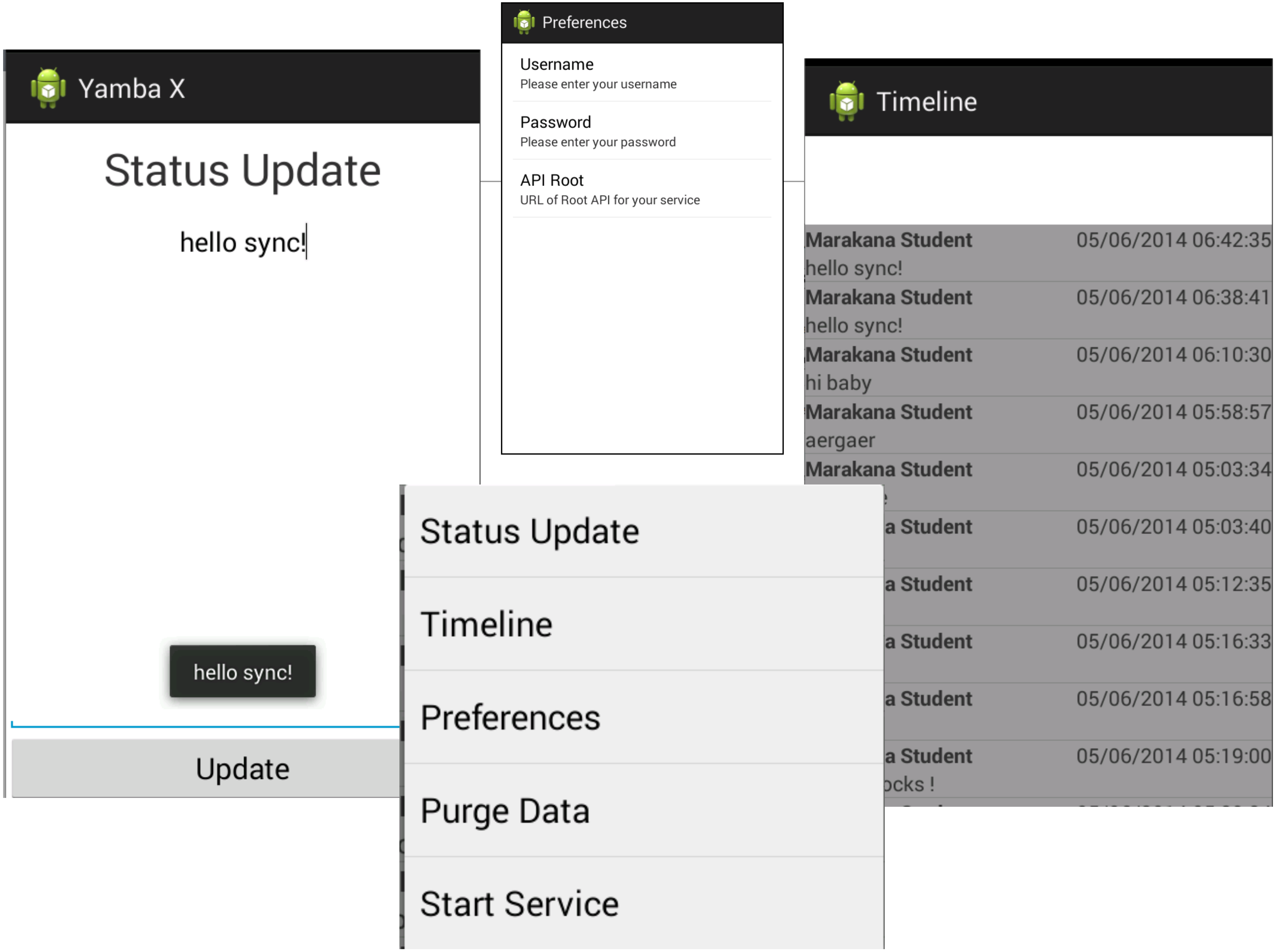
Yamba

Xtend version Encapsulated as 3 Labs

- A - Enable Simple Tweet + timeline update on background thread
- B - Move background thread to an Android Service + restructure application to use Lambdas + Command pattern
- C - replace custom event mechanism with generic Broadcast Receivers



Lab	*Objectives*
START	• Develop an application in Android using the Xtend language • Base the application on a twitter-like service hosted here: ◦ http://yamba.marakana.com • The structure of the application is derived from Learning Android
START	• Introduce Services into the Xtend YambaX application • Implement a background service to periodically update the twitter timeline
START	• Incorporate Broadcast Event Senders/Receivers into the application • Use BootReciever to recieve boot event to start application service on launch • Use NetworkReveicer to stop/start service when network is starting/stopping • Use Broadcast Events for status updates from background service to TimelineActivity



Username
Please enter your username

Password
Please enter your password

API Root
URL of Root API for your service

```
Marakana Student          05/06/2014 06:42:35
hello sync!
Marakana Student          05/06/2014 06:38:41
hello sync!
Marakana Student          05/06/2014 06:10:30
hi baby
Marakana Student          05/06/2014 05:58:57
aergaer
Marakana Student          05/06/2014 05:03:34
a Student                 05/06/2014 05:03:40
a Student                 05/06/2014 05:12:35
a Student                 05/06/2014 05:16:33
a Student                 05/06/2014 05:16:58
a Student                 05/06/2014 05:19:00
ocks !
```

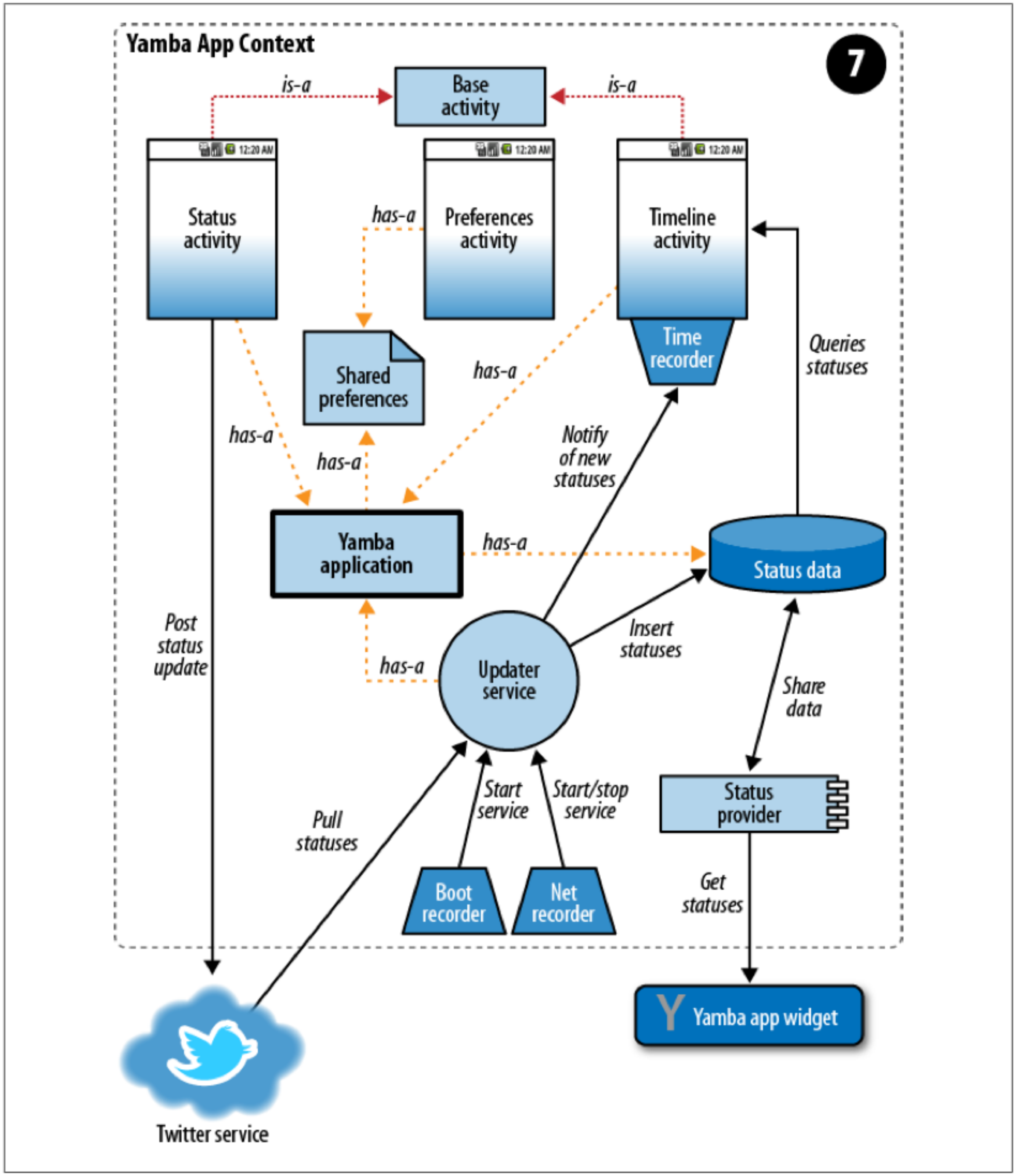


Figure 12-1. Yamba completion

BroadcastReciever

- A broadcast receiver is a component that responds to system-wide broadcast announcements.
- Many broadcasts originate from the system—for example, a broadcast announcing that the screen has turned off, the battery is low, or a picture was captured.
- Apps can also initiate broadcasts—for example, to let other apps know that some data has been downloaded to the device and is available for them to use.
- A broadcast receiver is implemented as a subclass of BroadcastReceiver and each broadcast is delivered as an Intent object.

LocalBroadcastManager

- Helper to register for and send broadcasts of Intents to local objects within your process. This is has a number of advantages over sending global broadcasts with sendBroadcast(Intent):
    - You know that the data you are broadcasting won't leave your app, so don't need to worry about leaking private data.
    - It is not possible for other applications to send these broadcasts to your app, so you don't need to worry about having security holes they can exploit.
    - It is more efficient than sending a global broadcast through the system.

BootLoader

- Start the service as soon as the application phone is turned on.

```
class BootReceiver extends BroadcastReceiver
{
  override onReceive(Context context, Intent intent)
  {
    context.startService(new Intent(context, typeof(UpdaterService)))
  }
}
```

- Permissions and receiver must be declared in manifest

```
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
      <receiver android:name="com.marakana.yambax.BootReceiver">
        <intent-filter>
          <action android:name="android.intent.action.BOOT_COMPLETED" />
        </intent-filter>
      </receiver>
```

NetworkReceiver

```
class NetworkReceiver extends BroadcastReceiver
{
  override onReceive(Context context, Intent intent)
  {
    val isNetworkDown = intent.getBooleanExtra(ConnectivityManager.EXTRA_NO_CONNECTIVITY, false);

    if (isNetworkDown)
    {
      if (YambaApplication.serviceRunning)
      {
        Log.d("YAMBA", "onReceive: NOT connected, stopping UpdaterService");
        context.stopService(new Intent(context, typeof(UpdaterService)))
      }
    }
    else
    {
      if (!YambaApplication.serviceRunning)
      {
        Log.d("YAMBA", "onReceive: connected, starting UpdaterService");
        context.startService(new Intent(context, typeof(UpdaterService)))
      }
    }
  }
}
```

- Triggered whenever network is enabled/disabled
- Use this event to start/stop UpdaterService

UpdaterService - Change to use BroadcastReciever

- Define some identifiers for the events

```
class UpdaterService extends BackgroundService
{
  public static final String NEW_STATUS_INTENT                = "com.marakana.yamba.NEW_STATUS"
  public static final String SEND_TIMELINE_NOTIFICATIONS      = "com.marakana.yamba.SEND_TIMELINE_NOTIFICATIONS";
  public static final String RECEIVE_TIMELINE_NOTIFICATIONS = "com.marakana.yamba.RECEIVE_TIMELINE_NOTIFICATIONS"
```

- sendBroadcast (just the event, not the data)

```
  override def void doBackgroundTask()
  {
    ...
      val List<Twitter.Status> timeline  = twitter.getFriendsTimeline
      newTweets = if (app.timeline.size == 0) timeline else timeline.filter [it.id > app.timeline.get(0).id]
      ...
      app.updateTimeline(newTweets)
      sendBroadcast(new Intent(NEW_STATUS_INTENT), RECEIVE_TIMELINE_NOTIFICATIONS);
    ...
  }
```

TimelineReceiver BroadCastReciever

- This receiver will be triggered when new status updates arrive

```
class TimelineReceiver extends BroadcastReceiver
{
  var TimelineActivity timelineActivity

  new (TimelineActivity activity)
  {
    timelineActivity = activity;
  }

  override onReceive(Context context, Intent intent)
  {
    timelineActivity.timelineAdapter.notifyDataSetChanged
  }
}
```

TimelineActivity

```
class TimelineActivity extends BaseActivity
{
  @Property TimelineAdapter timelineAdapter
  var TimelineReceiver receiver
  var IntentFilter     filter

  override onCreate(Bundle savedInstanceState)
  {
    ...
    receiver = new TimelineReceiver (this)
    filter   = new IntentFilter( UpdaterService.NEW_STATUS_INTENT )
  }

  override onResume()
  {
    super.onResume
    super.registerReceiver(receiver, filter, UpdaterService.SEND_TIMELINE_NOTIFICATIONS, null);
  }

  override onPause()
  {
    super.onPause();
    unregisterReceiver(receiver)
  }
```

- When TimelineActivity resumes, register to receive UpdaterService events
- When paused, unregister…

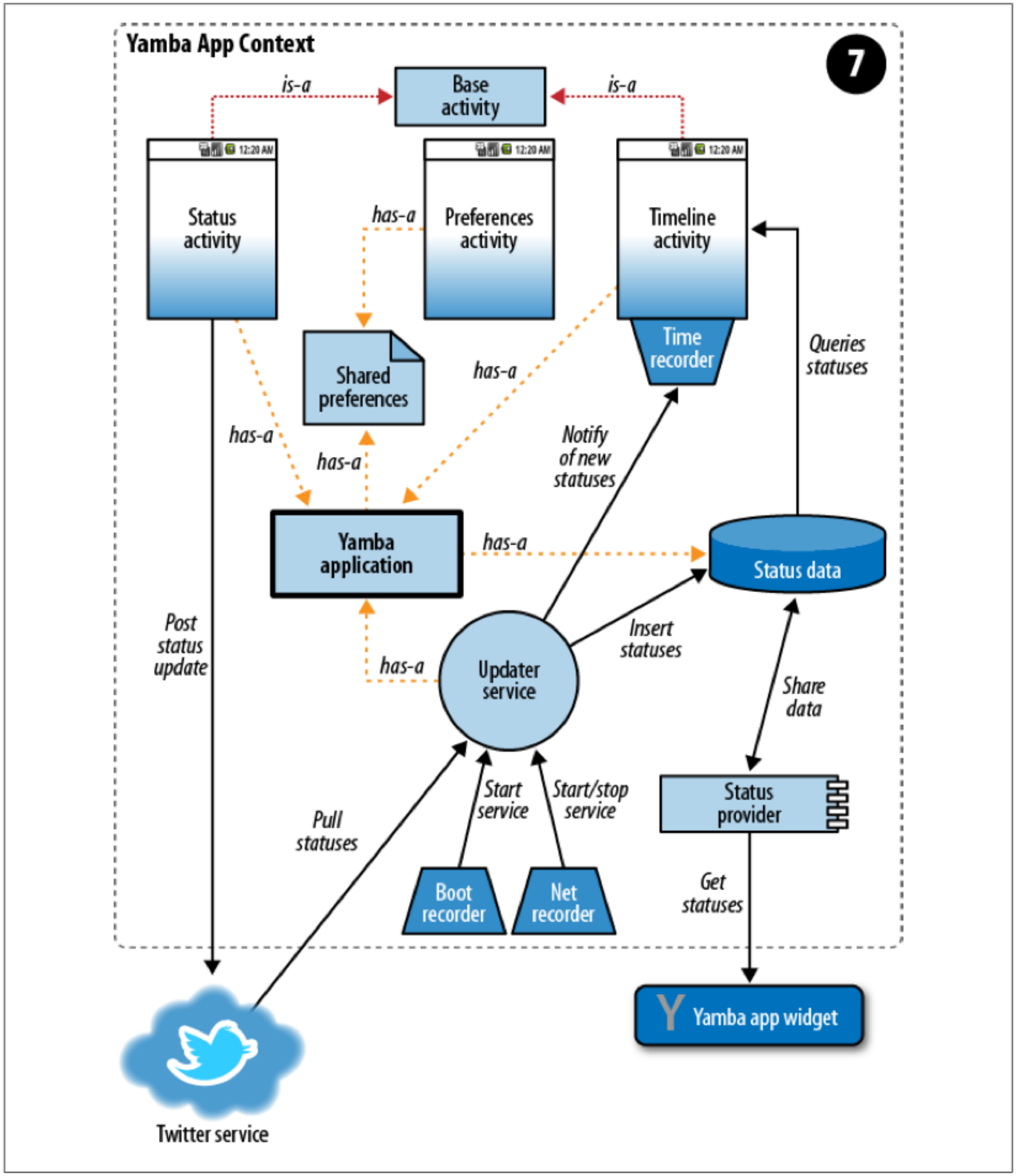
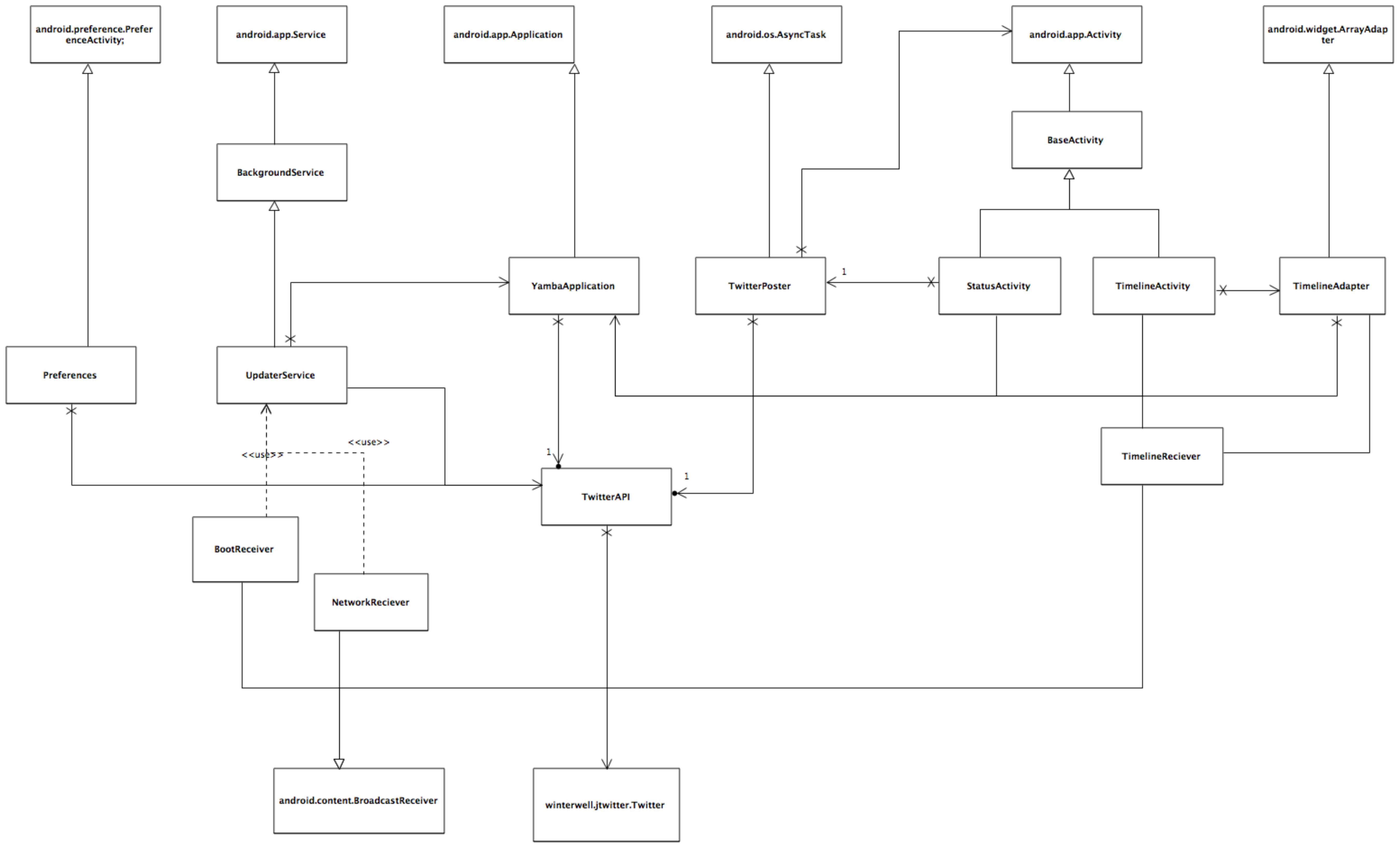


Figure 12-1. Yamba completion


android.preference.PreferenceActivity;
android.app.Service
android.app.Application
android.os.AsyncTask
android.app.Activity
android.widget.ArrayAdapter
BaseActivity
BackgroundService
YambaApplication
TwitterPoster
StatusActivity
TimelineActivity
TimelineAdapter
Preferences
UpdaterService
TimelineReciever
TwitterAPI
BootReceiver
NetworkReciever
android.content.BroadcastReceiver
winterwell.jtwitter.Twitter
<<use>>
<<use>>
1
1
1