Manual Process Management Interface for the Android Operating System

Authors: David Carson, Devin Denis, Sergey Vershinin

Date: April 12, 2013

Abstract

Developed by Google Inc., Android is a Linux-based operating system widely deployed on mobile devices such as smartphones and tablets. The stock task management utility provided with the Android operating system does not provide the user with convenient access to process management functionality. The ProcDroid application developed for this project enables the user to have greater control over process management by surfacing access to the application information page, and the kill process and force stop commands.

Table of Contents

Table of Figures

1.0 Introduction

This section provides an overview of the report, including a summary of background information, the problem statement, a brief description of the problem this project had set out to solve, as well as an outline of the main sections of the report.

1.1 Background

Developed by Google Inc., Android is an open-source, Linux-based operating system (OS) widely deployed on mobile devices such as smartphones, tablets, and other hardware platforms. The Android OS is based on the Linux kernel. However, there is some debate as to whether it constitutes a "proper" distribution of the Linux OS, or if Android's development has diverged far enough from the mainstream Linux OS so as to consider it to be a separate project altogether.

The main differences between Android and a typical Linux distribution is that Android does not make use of the X-Windows library, and instead relies on a modified Java API. The Dalvik process virtual machine (VM) provides the runtime environment for all of Android's end-user functionality, including most third party applications.

As of April 13, 2013, the latest release of the Android OS is at version 4.2, corresponding to API[1] level 17. Although due to the considerable effort involved in adapting the Android system to run on a specific hardware platform, many devices continue to make use of older versions of the OS.

1.2 Problem Statement

The focus of this project is the development of an improved process management application for legacy versions of the Android OS.

While the Android OS provides considerable task and process management functionality, the default task management application is limited to switching between tasks and viewing application settings.[2] This is by design, as Android implements a sophisticated memory management model that reduces the need for manual task and process management under most circumstances.

When considering features desired by an Android application developer, it is easy to see that the functionality provided by the stock task manager is insufficient. This project set out to develop an application that would enable an Android application developer to gain convenient access to the kill process, force stop, and application settings functions from a single integrated user interface.

1.3 Result

The ProcDroid application was developed using the Android Software Development Kit (SDK). The target API for this project was level 8 (Android OS v2.2).

---

[1] API is an acronym that stands for application programming interface.

[2] Viewing of the application information page is not possible using stock task managers included with the earlier versions of the Android OS.

The ProcDroid application addresses the problem outlined above by providing a graphical user interface that allows the user to access the kill process and force stop commands, as well as viewing an application's information page. The application interface consists of a single view that presents the user with a scrollable list of all of the processes currently running on the system, with the commands available via three buttons placed to the right of each entry in the list.

It is important to note that the kill process and force stop functions require ProcDroid to have super-user level of access.

1.4 Report Outline

This report consists of four main sections: Background, Description of the Result, Evaluation of the Result, and Conclusions. The report also contains three appendices: Contributions of Team Members, References, and the Evaluation Questionnaire. A brief description of the main sections of the report is provided below.

Background

This section provides information about the Android OS, its memory management model, as well as the current market share of legacy versions of the Android OS.

Description of the Result

This section provides information concerning the development and implementation of the ProcDroid application. The section includes an overview of the features of the application, as well as the approach used to implement them. Also provided are the detailed description of the graphical user interface and a discussion of the result's shortcomings.

Evaluation of the Result

This section contains an overview of the equipment used to test the ProcDroid application as well as the results of system testing conducted by the developer team. The section also includes the methodology and results of a small-N hybrid quantitative/qualitative evaluation of the project result with users who were not involved in the development of the application.

Conclusions

The Conclusions section presents an overview of the project result and provides a short discussion on the usefulness and viability of real-world use of the ProcDroid application.

2.0 Background

2.1 The Android OS

Developed by Google Inc., Android is an open-source, Linux-based operating system widely deployed on mobile devices such as smartphones, tablets, and other hardware platforms. The initial release of the Android OS was built around version 2.6 of the Linux kernel. Starting with version 4.0 also known as *Ice Cream Sandwich*, the Android OS uses code from 3.x versions of the Linux kernel.[3]

There is some debate as to whether it constitutes a "proper" distribution of the Linux OS, or if Android's development has diverged far enough from the mainstream Linux OS so as to consider it to be a separate project altogether. However, Linus Torvalds has said he believes that the Android fork will eventually be integrated into the main Linux codebase.[4]

Version	Codename	API	Distribution
1.6	Donut	4	0.1%
2.1	Eclair	7	1.7%
2.2	Froyo	8	4.0%
2.3 - 2.3.2	Gingerbread	9	0.1%
2.3.3 - 2.3.7		10	39.7%
3.2	Honeycomb	13	0.2%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	29.3%
4.1.x	Jelly Bean	16	23.0%
4.2.x		17	2.0%

Figure 1 - Percentage of devices running the various versions of the Android OS as of 2013-04-02. (Source: developer.android.com)

Aside from modifications to the kernel, the main difference between the Android OS and a typical Linux distribution is that Android does not make use of the X-Windows library. Instead, Android application software runs on an application framework which includes libraries based on the Apache *Harmony* implementation of the Java API.[5] The Dalvik process virtual machine (VM) provides the runtime environment for all of Android's end-user functionality, including most third party applications.[6]

As of April 13, 2013, the latest release of the Android OS is version 4.2, corresponding to API level 17.[7] Adapting the Android system to run on a specific hardware platform[8] involves considerable effort. Because of this, many devices continue to make use of older versions of the OS (see Figure 1).

---

[3] *Android (operating system)*, Wikipedia. (Source: http://en.wikipedia.org/wiki/Android_(operating_system), retrieved on 2013-04-14)
[4] *Linus Torvalds on Android, the Linux fork*, Vaughan-Nichols Steven J., ZDNet. [2011-08-18] (Source: http://www.zdnet.com/blog/open-source/linus-torvalds-on-android-the-linux-fork/9426, retrieved on 2013-04-14)
[5] *Supra* note 3
[6] *Dalvik (software)*, Wikipedia. (Source: http://en.wikipedia.org/wiki/Dalvik_(software), retrieved on 2013-04-14)
[7] *Jelly Bean*, Google Inc. (Source: http://developer.android.com/about/versions/jelly-bean.html, retrieved on 2013-04-14)
[8] *Supra* note 3

2.2 Android OS Memory Management Model and Process Lifecycle

When an application is executed, the Android OS starts a new Linux process. This behaviour may be modified if components of this application are already running in their own process. Once active, the process may create new threads or new processes. This behaviour may be desirable given that tasks requiring significant computation time can make the user interface appear frozen if they are run from the same thread.[9]

Starting with version 2.2 (API level 8) the Android OS implements an advanced memory management model that optimises memory usage and overall performance.[10] Under this memory management model, the Android OS may decide to kill a process if it requires additional memory resources. To decide which processes it can kill, the Android system uses an "importance hierarchy" based on the state of the components of an application in any given process.[11] The importance hierarchy consists of five levels:

1. Foreground process (highest priority, killed last)
2. Visible process
3. Service process
4. Background process
5. Empty process (lowest priority, killed first)

The exact characteristics that determine what priority is assigned to a given process are beyond the scope of this report.

2.3 Manual Process Management in the Android OS

The model described in the previous section makes manual termination of a process unnecessary for the end-user of an Android OS. However, the ability to manually terminate a process is a feature that may be desired in a number of use cases. One such case would be a power user running software known to be unreliable, who may find themselves in need of manually terminating a process. Another use case involves an Android application developer who may desire to manually terminate processes while carrying out testing of their application.

The stock version of the Android OS provides considerable task[12] and process management functionality. The default task management application provided by the Android OS is limited to switching between tasks and viewing application information. Manual task termination is available through the Settings application, by selecting the "Applications" menu item, followed by the "Manage Applications" option. To manually terminate an application the user would then need to select the "Running" tab, select the application of interest, and use the "Force Stop" option to terminate the application (see Figure 2).

---

[9] *Processes and Threads*, Google Inc. (Source: http://developer.android.com/guide/components/processes-and-threads.html, retrieved on 2014-04-14)

[10] *Android 2.2 Platform Highlights*, Google Inc. (Source: http://developer.android.com/about/versions/android-2.2-highlights.html, retrieved on 2014-04-14)

[11] *Supra* note 9

[12] A task in this context may refer to a single process or a number of related processes spawned by an Android application.

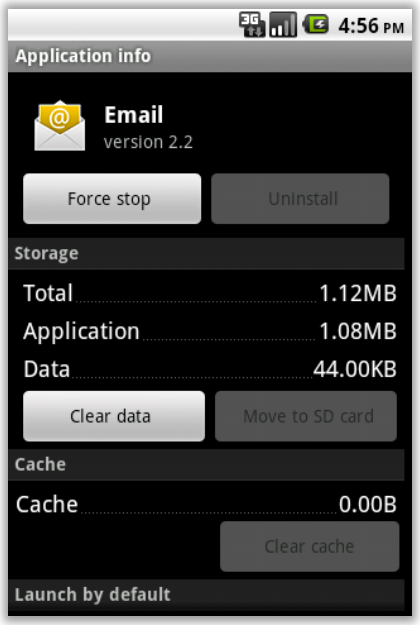

Figure 2 - A sample screenshot of an Android Application Information page

Starting with Android OS version 3.0, the stock task management application allows the user to access the "Application Settings" page from the list of the applications recently launched on the device. Older versions of the stock task switching application do not provide this feature.

As such, the functionality desired by the use cases described earlier in this section is not provided by the stock Android OS task management utility. The goal of this project is to create an application that would provide a single integrated interface to simplify manual process termination for users who may require such functionality.

3.0 Description of the Result

The ProcDroid application was developed to address the problem outlined at the end of the previous section. In this section of the report the reader is presented with the detailed description of the main features of the ProcDroid application, as well as the details concerning the design and implementation of the ProcDroid graphical user interface (GUI).

3.1 Main Features

There are four main features included in ProcDroid:

1. Displaying a list of currently running processes
2. Kill process command
3. Force stop command
4. Applications settings command

3.1.1 Process Listing

In order to implement an interface that would provide the desired functionality, we must first provide the user with a listing of the processes currently running on the system. To achieve this, we make use of an *ActivityManager* object. The *ActivityManager* is a system object which can be accessed using the *getSystemService(ACTIVITY_SERVICE)* call. The *ActivityManager* has a method, *getRunningAppProcesses*, which returns a list of *RunningAppProcessInfo* objects. Each object contains a variety of information about one of the processes running on the system, the most relevant being the process name and the process identifier (pid). To obtain a full listing of currently running processes, we store the list of *RunningAppProcessInfo* objects and display their names.

3.1.2 Kill Process and Force Stop Commands

The next two features are closely related: killing a process and executing a force stop command. In both cases, when the user selects the option, we get the corresponding *RunningAppProcessInfo* object. We then pass the *RunningAppProcessInfo object* to the *killProcess* method of the ProcDroid application. This method creates a separate process with super user level of access. In order to be able to execute the super user command ("su") the user must have root access to their device.[13] We then write the command "kill" or, in the case of a force kill, "kill -9", followed by the pid of the process.

The difference between a kill process and a force stop command is the signal sent to the process. The "kill" command sends the TERM signal, which can be caught by a process. This allows the process to terminate gracefully. It is also possible for an unresponsive process to catch and then ignore the TERM command. Using the force stop command sends the KILL signal, which cannot be caught, and will end execution of the process immediately. This can cause data loss, but will guarantee the process's execution is terminated.

[13] Typically, Android smartphones and tablets do not allow this level of access to the end-user. However, obtaining root access can be achieved through the use of a number of root-kit utilities provided by third-party developers. In most cases, the use of such utilities voids the manufacturer's warranty for the device.

On the Android operating system, sending a TERM command for a process that the OS has not itself selected for termination often results in the process restarting immediately. Sending the KILL command is more likely to prevent this, but the operating system may still restart the processes based on other considerations inherent in the memory management model (see section 2.2 of this report).

3.1.3 Viewing Application Settings

The final feature of ProcDroid is the ability to jump to a given application's "Application Information" page (see Figure 2 for a sample screenshot an Application Settings page). The method used to achieve this depends on the version of Android in question. On Android 2.3 and newer, this is a supported operation with a simple setup. We create an intent option, set its action to *Settings.ACTION_APPLICATION_DETAILS_SETTINGS*, and set the data to a uniform resource identifier (URI) object containing the process name. We then call *startActivity* with our intent object.

The Android OS version 2.2, which is the oldest version of the Android API we chose to support[14], does not natively support this operation. To get to the "Application Information" page we must manually construct the same intent that is used by the Settings application. We create the intent and set its action to *Intent.ACTION_VIEW*. We then have to call *setClassName* with some package data related to the Settings application. Finally we call *putExtra* with the string "pkg" and the process name, and then *startActivity*. Credit for clarifying this process goes to Stack Overflow user ZhengZhiren for his answer to the question concerning this operation.[15]

3.2 Graphical User Interface

All Android applications use a layout system for managing and interfacing with graphical elements. ProcDroid uses a number of different layouts that when combined form the final user interface (UI) that is presented to the user.

The Android UI layouts are written in XML[16] and contain entries for each UI element and their respective properties. These properties can include positioning, size, associated colors, and handler functions linked to interactive elements. Please use Figure 3 as a visual reference for the elements discussed below.

The ProcDroid application uses two main layouts:

The main application view layout (home layout) consists of a label for the list of processes, two buttons that link to the "Help" (2) and "About" (3) pages, another button for refreshing the list of processes (1), and finally the list of processes themselves.

---

[14] The decision to support Android 2.2 as the base target API for our project was based on the usage statistics featured in section 2.1 of this report. In short, the *Froyo* release (API level 8) is the oldest version of the Android OS still in use on a significant percentage of active devices (4.0%) as of 2012-04-02.

[15] *Start Android Application Info Screen,* StackOverflow. (Source: http://stackoverflow.com/questions/4421527/start-android-application-info-screen, retrieved on 2014-04-14)

[16] XML is an acronym that stands for extensible mark-up language

Figure 3 - Screenshot of the ProcDroid user interface

The second layout defines the presentation of the individual elements of the process list (process layout). This layout includes a label for the name of a process and three buttons used to view information about the process (4), execute the kill process command (5), and to execute the force stop command (6).

When the application is launched, it gets a list of all the running processes on the system. For each process, it creates a new process layout instance and sets the label to the name of that process. Once a layout for each process is created they are all added into the list in the home layout. At this point the UI is displayed to the user.

4.0 Evaluation of the Result

4.1 System Testing

During the development phase, the ProcDroid application was tested using both simulated and physical devices running the Android OS. The Android Virtual Device (AVD) is a software emulator provided as part of the Android Developer Tools bundle. The AVD can be set to run any version of the Android OS supported by the SDK (versions tested included 2.2 and 4.2.2). The physical device used for testing was an Acer Liquid E smart phone running Android OS version 2.2.2.

Given the limited scope of the functionality provided by ProcDroid, system testing largely consisted of verifying the correct execution of the main features of the application. In addition, usability testing was conducted to ensure correct rendering of the application's GUI in both portrait and landscape modes.

On the whole, the main features of ProcDroid were found to function as intended. The ProcDroid GUI was found to perform equally well when viewed in both portrait and landscape modes.

Two 'bugs' were identified during system testing:

1. Some processes are not associated with an Application Information page. This results in no action being taken when the application settings button is pressed. It may be possible to improve the way this feature is implemented in API level 9 and above. But given the method of implementation of this feature in API level 8, it is unlikely that improvement is possible under Android 2.2.
2. The kill process command sometimes appears do nothing. This behaviour could be explained by the system instantaneously re-launching the process as noted in section 3.1.2 of this report. Alternately, it is possible that the list of currently running processes is refreshed before the process terminates itself (as per graceful termination vs. force stop).

Given the likely explanations of the two 'bugs' identified above, it was deemed that their resolution was outside of the scope of the problem set identified for this project.

4.2 Evaluation methodology

In order to assess the result of this project a small-N hybrid qualitative/quantitative evaluation was conducted with individuals who were not involved in the development of the ProcDroid application.

In total four individuals were given an opportunity to use the ProcDroid application and subsequently filled out a short questionnaire about their experience. The questionnaire used for the evaluation can be found in Appendix III of this report.

Of the four participants, only one individual actually belonged to the target user group for the application (i.e. an Android application developer). Other participants self-identified as novice users of the Android operating system.

4.3 Evaluation Findings

Evaluation participants were asked to characterize their overall experience using the ProcDroid application on a 5-point scale ranging from completely unsatisfactory (1) to completely satisfactory (5). In response to this question, the ProcDroid application received an average rating of 4.25. This can be interpreted to mean that participants found their experience using the ProcDroid application to be mostly satisfying.

The next three questions dealt with the main features of the application.

Half of the participants (50%) found that the application settings button always functioned as expected, while the other half (50%) found that it only function as expected some of the time.

Three of the four participants (75%) found that the kill process button functioned as expected, while one participant (25%) found it to function as expected only some of the time.

Half of the participants (50%) found that the force kill button always functioned as expected, while of the two remaining participants one found it to function some of the time (25%), while another was unsure (25%).

From the data cited above it can be concluded that in aggregate, the ProcDroid application's main features functioned as expected only some of the time. This finding may be explained by the bugs uncovered during system testing (see section 4.1).

When asked to characterize their impression of the ProcDroid application's user interface on a 5-point scale ranging from completely unintuitive (1) to completely intuitive (2), participants gave the ProcDroid UI an average rating of 3.75. As such, it can be concluded that in aggregate, evaluation participants found the ProcDroid UI to be mostly intuitive.

At the end of the questionnaire evaluation participants were given an opportunity to provide additional comments. Three of the participants chose to do so. Most of the comments concerned a variety of shortcomings in the user interface including:

- *"The UI is too "busy", too many buttons that do exactly the same thing."*
- *"The crowding also seems to prevent the entire process name from being displayed in many circumstances."*
- *"Bomb symbol for force kill was less intuitive than other symbols."*

Finally, one participant pointed out that the application was better suited to users with advanced knowledge of the Android operating system.

In summary, the evaluation results suggest that participants had an overall positive experience using the application, but had noticed a variety of minor flaws in the application's main functionality as well as its user interface.

5.0 Conclusion

The default task management application provided with the Android OS provides limited functionality for manual management of processes currently running on the system. Android application developers and power users may wish to have more control over processes running on the system. This project set out to address this problem by providing the user with a single interface that allows access to the application information page, as well as the ability to issue kill process and force stop commands.

The ProcDroid application was developed to address the lack of manual process management capability in the stock distribution of the Android OS. ProcDroid features include listing of currently running processes, ability to access an application information page for a given process (if one exists), and two options for terminating the said process. The implementation of the process termination features requires that the user have root level access to the system.

To assess the result, both system testing and an evaluation with individuals not involved in the development process were conducted. System testing conducted by the development team uncovered two minor flaws in the application that did not impede its overall functionality. The evaluation found that on the whole, the ProcDroid application features functioned as expected, and that the user interface was mostly intuitive.

The evaluation results did point to a variety of shortcomings of the ProcDroid UI, some of which had to do with UI design, while others were related to the bugs discovered during system testing. The latter concerned the fact that the Android OS at times chooses to re-spawn a process that has been issued a kill signal, resulting in application behaviour that may seem confusing to the user.

In summary, the ProcDroid application developed for this project represents a viable solution to the absence of manual process management features in the stock distribution of the Android OS.

APPENDIX - I Contributions of Team Members

The ProcDroid application was developed by David Carson, Devin Denis, and Sergey Vershinin. The respective contributions of each of the team members were as follows:

David Carson

David participated in the initial brainstorming that yielded the idea to develop an Android process management tool. At the initial stages of the project David conducted research on the memory management model employed by the Android operating system. David's main contribution to the project was the implementation of the ProcDroid graphical user interface. He was also responsible for providing content for the sections of the final report that describe the GUI design and implementation.

Devin Denis

Devin participated in the initial brainstorming that yielded the idea to develop an Android process management tool. Once it was time to begin the development process, Devin took the lead on implementing ProcDroid's main features: the kill process and force stop commands, as well as the shortcut to the app settings page. Devin was also responsible for providing content for the sections of the final report that describe the design and implementation of ProcDroid's process management features.

Sergey Vershinin

Sergey led the initial brainstorming that yielded the idea to develop an Android process management tool. During the development process Sergey contributed ideas to the design and feature set of the ProcDroid application. Sergey was also responsible for procurement and design of the graphical resources used by the application UI. Sergey's main responsibility lay in documenting the project including the project proposal and the final report.

APPENDIX - II References

The references used throughout this report are presented below sorted by title, in alphabetical order:

1. *Android (operating system)*, Wikipedia.
   (Source: http://en.wikipedia.org/wiki/Android_(operating_system), retrieved on 2013-04-14)
2. *Android 2.2 Platform Highlights*, Google Inc.
   (Source: http://developer.android.com/about/versions/android-2.2-highlights.html, retrieved on 2014-04-14)
3. *Dalvik (software)*, Wikipedia.
   (Source: http://en.wikipedia.org/wiki/Dalvik_(software), retrieved on 2013-04-14)
4. *Jelly Bean*, Google Inc.
   (Source: http://developer.android.com/about/versions/jelly-bean.html, retrieved on 2013-04-14)
5. *Linus Torvalds on Android, the Linux fork*, Vaughan-Nichols Steven J., ZDNet. [2011-08-18]
   (Source: http://www.zdnet.com/blog/open-source/linus-torvalds-on-android-the-linux-fork/9426, retrieved on 2013-04-14)
6. *Processes and Threads*, Google Inc.
   (Source: http://developer.android.com/guide/components/processes-and-threads.html, retrieved on 2014-04-14)
7. *Start Android Application Info Screen,* StackOverflow.
   (Source: http://stackoverflow.com/questions/4421527/start-android-application-info-screen, retrieved on 2014-04-14)

APPENDIX - III Evaluation Questionnaire

ProcDroid Evaluation Questionnaire

This questionnaire is intended to record your first-hand impressions of the ProcDroid application. Prior to filling out this questionnaire you should have had an opportunity to use the ProcDroid application and test out its functionality in a manner of your choice. This questionnaire does not assume that you were provided with outside guidance on how to use the ProcDroid application.

Thank you for taking the time to fill out this questionnaire, your honest feedback is appreciated.

Q1: Which of the following best describes you level of experience using the Android operating system?

1. Novice
2. Casual user
3. Experienced user
4. Expert

Q2: On a scale of 1 to 5, where 1 means completely unsatisfactory, 3 means neither satisfactory nor unsatisfactory, and 5 means completely satisfactory, how would you characterize your overall experience using the ProcDroid application?

1 2 3 4 5 Unsure

Q3: Did you find the application settings button to function as expected?

Yes Sometimes No Unsure / Did not use

Q4: Did you find the kill process button to function as expected?

Yes Sometimes No Unsure / Did not use

Q5: Did you find the force stop button to function as expected?

Yes Sometimes No Unsure / Did not use

Q6: On a scale of 1 to 5, where 1 means completely unintuitive, 3 means neither intuitive nor unintuitive, and 5 means completely intuitive, how would you characterize your impression of the ProcDroid user interface?

1 2 3 4 5 Unsure

Q7: Do you have any comments you would like provide concerning your evaluation of the ProcDroid application?

Thank you!