Vol.2011-SE-172 No.3
Vol.2011-EMB-21 No.3
2011/5/17

Android における OS プロセス可視化環境の開発

中川裕貴[†]　早川栄一[††]　西野洋介[†††]

Android は，Linux カーネルと DalvikVM の二つの階層で構成されているので，複数のプロセスの動作を理解することが困難である．そこで，本報告では Android における OS プロセス可視化環境の開発を行った．本システムでは，ftrace を用いた低オーバヘッドのシステム情報取得環境を構築し，プロセス生成や切換えに関する情報を取得可能とした．この情報を元に Web ブラウザで可視化環境を開発することで，ユーザが容易に利用可能な環境を構築した．本環境ではプロセスの状態遷移図，時間変化グラフ，プロセスのツリー構造を表示することにより，利用者がプロセスの実行時間や状態の遷移，プロセスの関連を容易に把握できるようにした．

Development of Operating System Process Visualization Environment in Android

Yuuki Nakagawa[†]　and Eiichi Hayakawa[††]　and Yousuke Nishino[†††]

Android is difficult to understand the behavior of processes, because it is built from Linux kernel and Dalvik VM. In this report we describe the development of the visualization about operating system process in Android. Data about context switch and process creation is available to acquire from ftrace based low overhead mechanism. Visualization environment is also developed that it executes on Web browser to be easily available to users. The environment consists of three components: state transition diagram of processes, time transition diagram and process tree diagram. User can understand the process execution time, the process state transition and the relationship among processes from these diagrams.

1. はじめに

近年，Android を携帯電話や組込み OS として利用する機会が増加している．その背景としては，Android を搭載するスマートフォンを作る上で開発時にかかる OS のライセンス料金がかからないということや，Eclipse による統合開発環境が提供されており，一般の開発者が無償で Android のアプリケーションを作成することができるということが挙げられる．つまり，端末はコストの削減，オープンソースなので手を加えやすいことや，アプリケーションについては Eclipse による統合開発環境が提供されており，開発しやすいことから Android の利用者が増加している．そして，Android 利用者の増加に伴い，Android のアプリケーションの開発や基盤となるオペレーティングシステムに関する学習をする者も増加している．

学習する方法として，Android のソースコードが公開されており，誰でも見ることが可能である．しかし，Android の学習には問題点がある．Linux カーネルのソースコードが 450 万行近くあり，Linux のプロセスの動作と DalvikVM のプロセスの動作の複数あるプロセスの動作を理解することは困難である 1)．

また，学習する際に PC に学習用のアプリケーションをインストールする必要がある．複数の PC にアプリケーションをインストールする場合，学習者または教授者の手間が増えてしまう．このことが，OS 学習を行う際の学習者にとって余計な負担になる．

また，Android の学習をするには，まず基本として OS の各機能の仕組みを理解する必要がある．Android を組込みシステムで用いる場合でも，デバイスの関する OS の各機能の仕組みを理解しなければならない．特に組込みシステムでは，機器の追加や省資源における性能保証が不可欠であり，デバイスドライバの追加やネイティブコードでの実装や協調動作などが必要になる 2)．このことから，OS のプロセスの振る舞いを理解することは重要となってきている．

本研究の目的は，可視化対象の OS を Android とし，その動作を可視化するシステムの開発を行うことである．その中でもプロセス管理を対象とし，プロセスの状態遷移やプロセス実行時間，プロセス同士の関連を可視化していく．その方法として，プロセスの切替えの構造のアニメーションやプロセスのシーケンス図による状態遷移の変化，ツリー構造による可視化を行う．これにより OS プロセスの動作を容易に把握することが可能になる．また，複数のプロセスの動作も把握しやすくなる．これによ

† 拓殖大学　大学院　電子情報工学専攻
Takushoku University
†† 拓殖大学　工学部　情報工学科
Faculty of Engineering Takushoku University
††† 東京都立　八王子桑志高等学校
Hachioji Soshi High School

Vol.2011-SE-172 No.3
Vol.2011-EMB-21 No.3
2011/5/17

り，ソースコードだけではプロセスの動的な状態の遷移や実行時間による変化などが理解できるようになる．

2. 従来のシステムとの比較

Android のプロセスを可視化するアプリケーションは複数存在している．TraceLogVisualizer 3) ではマルチプロセッサのトレースログを可視化して表示をするツールがある．このツールは様々な形式のトレースログに対応しており，タスクの状態遷移が表示される．可視化の表示部分をアプリケーションしにており，本研究はブラウザで表示をしている点が異なる．可視化結果をブラウザ上で表示することにより，実行画面のアプリケーションのインストールが不要であることや，ネット環境があれば使用できるので学習者が利用しやすいという利点がある．

Vzet 4) では Linux システムの CPU のプロセスを可視化するアプリケーションがある．このアプリケーションは ftrace トレースデータを使用して，プロセス名や状態，優先度，CPU，CPU 使用率，実行時間などをグラフで表示している．プロセスの切替わりやマルチコアの振舞いが見えるようになっている．しかし，このアプリケーションはグラフだけの表示をしているが，本研究はグラフと状態遷移図，ツリー構造の可視化を表示していることが異なる．また，可視化の表示はアプリケーションで表示をしている．しかし，アプリケーション表示は使用できる OS が決められているので，使用する場合はまず OS の確認が必要である．しかし，本研究は可視化の表示をアプリケーションで表示するのではなくブラウザ上で表示することにより，どの OS でもネット環境があれば使用できる．

3. 設計

3.1 全体構成

Android における OS プロセス可視化環境の開発の全体構成を図 1 に示す．

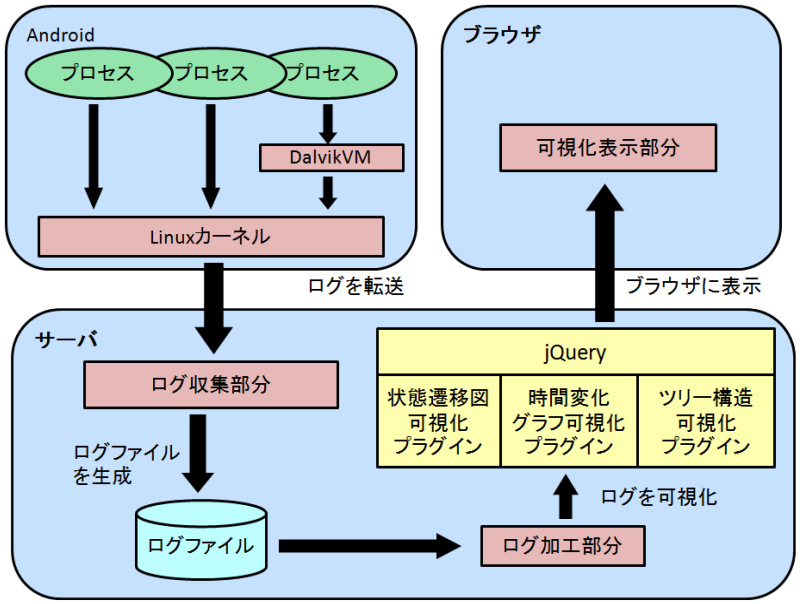

図 1　Android における OS プロセス可視化環境

Android 内でプロセスの動作ログを収集し，サーバに送り，ログファイルを生成する．ログファイルを可視化するために加工する．加工したログファイルを jQuery を使用して可視化をする．そして，可視化したログファイルをブラウザ上で表示する．

可視化環境は，ログファイルのデータサイズが大きくなるので，ログファイルの保存領域を Android 内ではなく，サーバ内部に保存する．Android 内で保存する場合は大きなサイズのログファイルの保存領域を取ることは難しい．しかし，Android 内にログファイルを保存できる保存領域を持っていない場合でも，サーバに保存することによりログファイルを生成し，保存することが可能である．

ブラウザ表示には jQuery を使用する．jQuery は JavaScript と HTML の両方で使用できる JavaScript ライブラリである．さらに jQuery はプラグイン機構を提供していて，既存のプラグインを使用すること以外にも自分で新しいプラグインを JavaScript の知識があれば，作成することが可能である．本システムでは，既存のプラグインではなく新しいプラグインを作成した．JavaScript を使用した理由としては，Flash を使用する場合は AdobeFlashPlayer をインストールしなければ見ることができないことや，読み込みが遅いなどのユーザビリティの低い点があげられる．これに対して JavaScript

Vol.2011-SE-172 No.3
Vol.2011-EMB-21 No.3
2011/5/17

の場合は，環境に左右されず見ることができることや，HTML の操作が容易であるという利点がある．

3.2 プロセス情報取得機構

プロセス情報取得では，Android の Linux カーネル内にある ftrace 5) を利用することでプロセス情報の取得を行う. この機能を利用することにより, プロセスの生成や, コンテキストスイッチの様子をトレースすることができる．プロセス情報取得機構では, プログラム内で ftrace に存在する生成されたログデータをトレースする. そして, トレースした結果を動作ログとして出力し，動作ログを可視化できる形式で動作ログを出力する．また，学習者は実行させるプログラムの内容に関係なく，そのプログラムが Android 内部でどのようにプロセスとして生成され状態遷移をすることで実行されているのか理解できる．

Android には DalvikVM と Linux カーネルがあり，Linux カーネルの C 言語と DalvikVM の Java 言語の二つの実行環境で動作している．この二つの言語の間でプロセスがどのように対応付けられているのかを知るために，DalvikVM から情報を取得し，DalvikVM 上で Java プロセスと Linux のプロセスがどのように変化しているかを対応づけて表示する．

3.3 取得する情報内容

Android の動作ログを取得する上で ftrace を利用することで得られるプロセス情報は次のものである．

- 実行中のプロセス
- 実行中のプロセス ID
- 実行中のプロセス優先度
- 割り当てられている CPU
- 実行開始時間
- トレースしている状態
- スイッチ後に遷移する状態
- スイッチ先のプロセス名
- スイッチ先のプロセス ID
- スイッチ先のプロセス優先度

これらを取得することにより, プロセスに割り当てられている CPU や実行中のプロセスの名前や ID，実行時間，優先度，トレースしている状態，スイッチ先のプロセスの名前や ID 優先度を知ることができる．トレースしている 6 種類の状態がある．それぞれの状態と意味を表 1 に示す．

表 1　トレースしている状態

トレースしている状態	意味
sched_wakeup_new	新たにプロセスが生成された
sched_wakeup	待ち状態から実行可能状態への遷移
sched_switch	実行可能状態から実行状態への遷移
sched_process_fork	新たにプロセスが生成された
sched_process_wait	子プロセスが停止するか死ぬまで待つ
sched_process_exit	終了し，親プロセスに戻る

これらの状態を知ることにより，プロセスの生成，消滅やプロセスの状態遷移の様子が理解できる．sched_wakeup_new と sched_process_fork はプロセスの生成としては同一だが, sched_process_fork は fork システムコールによって生成された場合のプロセスである．これに対して，wakeup_new はスレッド生成などの場合に用いられる．
また，ftrace で取得しているプロセスの状態は 7 種類あり，状態と意味を表 2 に示す．

表 2　プロセスの状態

ftrace での表記	状態	意味
R	Running	実行中状態もしくは実行可能状態
S	Sleep	待ち状態（入出力とは無関係に遷移する状態）
D	Disk sleep	ディスクの読み書きによる待ち状態
T	Stopped	停止状態
t	traced	トレース中
Z	zombie	ゾンビ状態
X	unknown	存在しない

表 2 において実行状態と実行可能状態に明確な区別がないのは，Linux では実行可能状態の中から最も優先度の高いプロセスが選ばれて CPU に割り当てられるためである．

ログファイルはブラウザと JSON で通信し，動作ログの情報を取得することで可視化をする．JSON は XML などと同様のテキストベースのフォーマットであり，記述が使用者にとって容易で理解しやすいデータフォーマットになっている．JSON を利用してブラウザとログファイルとで通信を行って可視化のログファイルを読み込んで可視化の表示をしている．

Vol.2011-SE-172 No.3
Vol.2011-EMB-21 No.3
2011/5/17

3.4 可視化画面

可視化画面をブラウザ上で表示させており，JavaScript で書いている．そして，JavaScript のプラットフォームである jQuery を用いて可視化を表示させていく. jQuery は既存のプラグインを使用すること以外にも自分で新しいプラグインを作成することが可能である．3 種類の可視化を行うので，それら三つのプラグインを作成した．それぞれを状態遷移図可視化プラグイン，時間変化グラフ可視化プラグイン，プロセスツリー可視化プラグインとして可視化を行う際に使用した. 表 3 は, 今回使用した API の種類と詳細を示す.

表 3　API の名称と詳細

API	詳細
move	指定したものを上下左右に移動するアニメーションの呼び出す
color	指定したものの色を徐々に変化させるアニメーションを呼び出す
show	指定したものを非表示から表示に変更するアニメーションを呼び出す
hide	指定したものを表示から非表示に変更するアニメーションを呼び出す

これらの API は 3 種類のプラグインで使用している．状態遷移図可視化プラグインは move と color の API を使用している．move はプロセスの状態遷移時のアニメーションで使用している. color はプロセスの状態遷移時の色の変化のアニメーションに使用している. 時間変化グラフ可視化プラグインは hide と show の API を使用している．

可視化の実行画面をブラウザで表示させており，図 4 に示す.

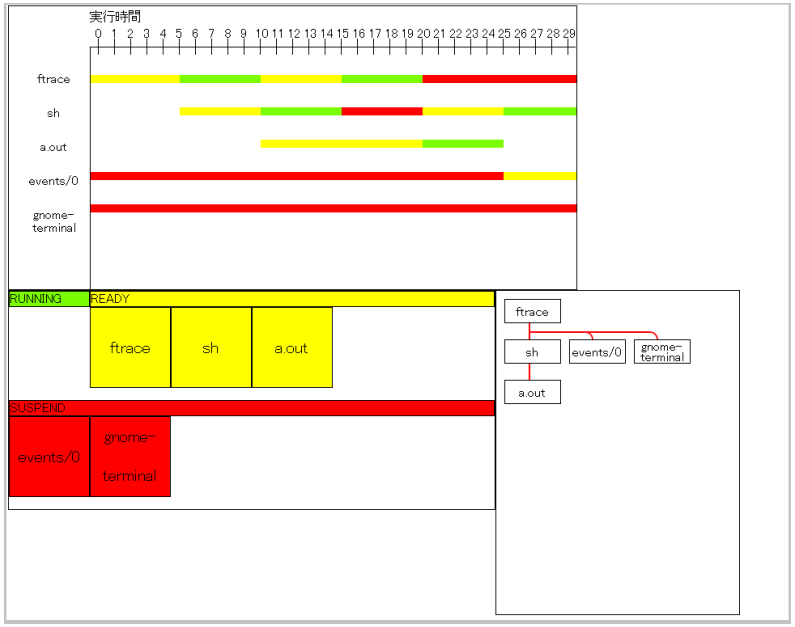

図 4　可視化の実行画面

図 4 の左上にプロセスの状態遷移図を可視化したものを表示している．左下にプロセスの時間変化グラフを可視化したものを表示している．右下にプロセスのツリー構造を可視化したものを表示している．この 3 種類で Android の可視化を行う．可視化の実行画面をアニメーションで表示する．DalvikVM 上で生成されたプロセスは名前の部分を灰色にして表示をする．次にこれら可視化の詳細な説明をする．図 5 にプロセスの状態遷移図を可視化した実行画面を示す.

Vol.2011-SE-172 No.3
Vol.2011-EMB-21 No.3
2011/5/17

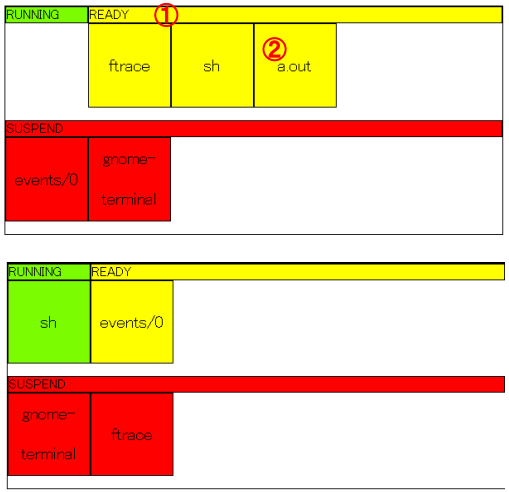

図 5　状態遷移図の実行画面

①はプロセスの状態を表している. プロセスの状態は 3 種類あり, 緑色の場所が CPU に割り当てられている状態，黄色の場所がスリープ状態，赤色の場所が待ち状態となっている．②では，プロセスの名前を表している．色についてはプロセスの状態と同じ色になっている．プロセスの状態が遷移すると②が遷移する状態の場所へ移動し，色を変化する．実行可能状態の場合，並び方が他の状態と違い，プロセスの優先度の高い順番に左側から並んでいく. プロセスの状態遷移時の移動と色の変化は jQuery を使用して，アニメーションで表示している.

状態遷移図を表示することにより，どのタイミングでプロセスが状態遷移するかわかるようになる. また, プロセスの優先度やプロセスの状態を把握することができる. また，状態遷移図をアニメーションで表示することにより，プロセスの状態の遷移を動的に表現することにより学習者が視覚的に理解できるようになる.

次に図 6 にプロセスの時間変化グラフを可視化した実行画面を示す.

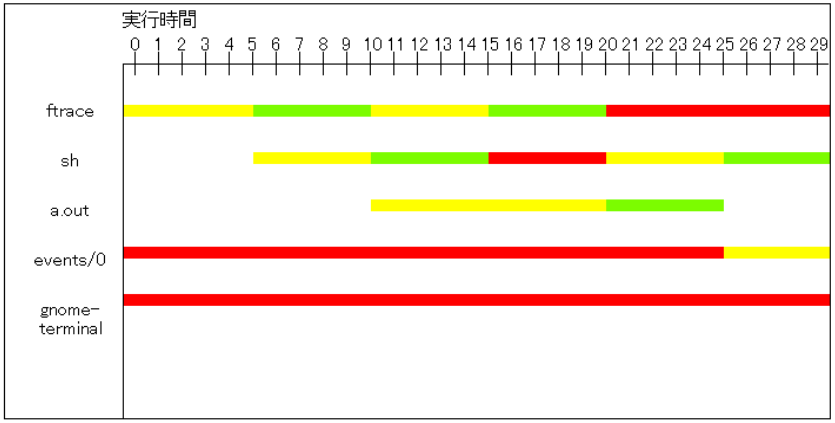

図 6　時間変化グラフの実行画面

左にある名前はプロセスの名前を表している．上にあるメモリと数値は，実行時間の表示をしている．時間の経過ごとに実行時間の目盛を変化させて表示していく．プロセスの名前の左にあるグラフはプロセスの状態の遷移するタイミングをグラフで表示させている．プロセスの状態については緑色が CPU に割り当てられている状態，黄色がスリープ状態，赤色が待ち状態と状態遷移図の実行画面と同じ色になっている．プロセスの名前の右にあるグラフがそのプロセスを表すグラフとなっている．グラフの時間変化の表示には時間の経過ごとにグラフを更新させていく.

時間変化グラフを表示することにより，プロセスの生成や消滅のタイミングが理解できるようになる．他にもプロセスの実行時間とそれに伴う状態の遷移を表示することにより，周期的に実行すべきプロセスのデッドラインが超えていないかを知ることができる.

次に図 7 にプロセスのツリー構造を可視化した実行画面を示す.

Vol.2011-SE-172 No.3
Vol.2011-EMB-21 No.3
2011/5/17

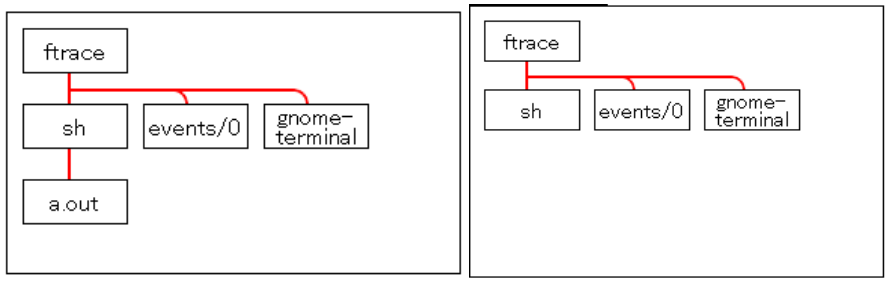

図 7　ツリー構造の実行画面

四角の中にはプロセスの名前を表示している．赤い線はプロセス同士の繋がりを表している．図 7 では ftrace というプロセスの下にある sh というプロセスがある．ftrace は sh の親プロセスである．また，sh は ftrace の子プロセスである．そして，プロセスの生成と消滅をアニメーションで表示をしている．

プロセスのツリー構造を表示することにより，プロセスの親子関係やプロセス同士の関係を知ることができる．

4. おわりに

本研究では可視化対象を Android とし，プロセス管理の可視化を行った．これにより，Android のプロセスの状態遷移の様子の情報を取得する機構を実現した．また，プロセスの親子関係の可視化も実現した．Android の可視化結果をブラウザ上で表示することにより，複数のマシン，ブラウザで動作が可能になった．

今後の課題としては可視化の実行画面のユーザビリティが良くなるように改良をすることや改善点があるのでそれを直していく．Android の Linux カーネルと DalvikVM の連携は取れているが, DalvikVM の Java 言語の可視化を行っていないので, DalvikVM の可視化をすることである．

参考文献

1)　Yosuke Nishino，Eiichi Hayakawa：Development of an OS Visualization System for Learning Systems Programming，HCI2003 International Conference on Human Computer Interaction（2003）
2)　安藤　友樹，柴田　誠也，本田　晋也，冨山　宏之，高田　広章：組込みマルチプロセッサシステムの設計改善支援，SWEST12（2010）
3)　後藤　隼弐，本田　晋也，長尾　卓哉，高田　広章：トレースログ可視化ツール TraceLogVisualizer，コンピュータソフトフェア（2010）8-23
4)　Vzet　LINEO
http://www.lineo.co.jp/modules/vaet/index.html
5) 本橋　大樹，西野　洋介，早川　栄一：組込みシステム学習支援環境の開発（コンピューターシステム），電子情報通信学会（2010）pp.279-285