Android 端末省電力化のための ブロードキャストインテント情報の調査

早川 愛[1]　磯村 美友[1]　竹森 敬祐[2]　山口 実靖[3]　小口 正人[1]

概要：近年スマートフォン端末が爆発的に普及し，その消費電力の低減は非常に重要な課題の一つとなっている．スマートフォン端末におけるバッテリ消費の原因として，主に「ディスプレイ」,「通信機能」,「CPU 使用率」の三点があげられるが，これら三点の主要因すべてに深くリンクしているのがアプリケーションであり，端末のバッテリ消費に大きな影響を与えていると考えられる．本研究では，Android アプリケーション特有のブロードキャストインテントに着目し，それによって誘発されるアプリケーションの振舞いや連鎖状況を詳しく解析することで，端末のバッテリ消費を削減することを目的とする．アプリケーション，ブロードキャストインテント，電池消費の因果関係を明らかにしていくことで，最終的に Android 端末の電池消費削減を目指した検討を行う．

An Investigation of The Broadcast-intent Information for The Electric Power Saving of The Android Terminal

Ai HAYAKAWA[1]　Mitimo ISOMURA[1]　Keisuke TAKEMORI[2]　Saneyasu YAMAGUCHI[3]
Masato OGUCHI[1]

1. はじめに

近年スマートフォン端末が爆発的に普及し，その消費電力の低減は非常に重要な課題の一つとなっている．BCN の調査によると，スマートフォンユーザの 70.7%がバッテリの持続時間に不満を持っている [1]．スマートフォン端末におけるバッテリ消費の原因は主に三点考えられる [2]．一つ目は，最も影響があるとされる「ディスプレイ」による消費である．画面の明るさやスリープに入るまでの時間が電池消費に大きく関わってくる．二つ目は，Wi-Fi や 3G(LTE)，GPS，Bluetooth などの「通信機能」である．特に移動しながら接続先の電波を探索したり，頻繁にハンドオーバすることで電力が消費される．三つ目は，「CPU 使用率」である．現在では，デュアルコアやクアッドコアなどスマートフォンながら複数の CPU を搭載する端末が多く存在し，そのそれぞれが各処理を行っている，
そしてこれら三点の主要因すべてに深くリンクしていると考えられるのがアプリケーションである．アプリケーションは，スマートフォンを利用する上では欠かせないものであり，ほぼすべてのスマートフォンユーザが何かしらのアプリケーションを各個人の判断でダウンロードし，端末をカスタマイズして用いていると言える．さらに近年では，ハードウェアの制限なくアプリケーションを開発できてしまうため，他のプロセスに画面が切り替わっても終了せずに動き続けるといった，バックグラウンドで動作可能なアプリケーションが多く存在する．よってアプリケーションは，その種類に依存するものの，電池消費には大きな相関関係があると考えられる．

さらに本研究では，Android 特有の機能であるブロードキャストインテントの振舞いに着目した．ブロードキャストインテントは，例えば，ACTION_TIME_TICK(現在時刻が変わった) や，ACTIONBATTERY_CHANGED(バッテリ状態が変化した) などのイベントが発生した時に，主にシステムが発行するものであり，登録されている任意の

[1] お茶の水女子大学
〒 112-8610 東京都文京区大塚 2-1-1
[2] KDDI 研究所
〒 102-8460 東京都千代田区飯田橋 3-10-10 ガーデンタワー
[3] 工学院大学
〒 163-8677　新宿区西新宿 1-24-2

複数のレシーバが受信し，各々の処理を実行させるものである．
つまりこのブロードキャストインテントは，端末がスリープに入っている時にも発行されることから，ユーザだけでなくアプリケーションでさえも意図せず発行している可能性があると考えられる．以上より，ブロードキャストインテントと端末のバッテリ消費には何かしらの因果関係があると予想できる．

本研究では，Android 端末のバッテリ消費削減を目指して，その要因だと考えられるブロードキャストインテントとアプリケーションの連鎖状況やそれぞれの挙動を詳しく解析していくことで，ブロードキャストインテントと電力消費の相関関係を明らかにすることを研究目的とする．それによって, アプリマーケットの審査基準において，電力消費に影響を与える可能性のあるアプリケーションにフィルターをかけることができたり，アプリ開発者にコードの書き方や挙動に何かしらのルールを提示することができる．さらにそれらの解析情報を用いて，実際にアプリケーションを利用するユーザに対してインストール済みのアプリケーションを審査するようなツールを提供することも将来的には可能であると考えられる．

これらの目的を実現するために，Android 端末が発行するブロードキャストインテントを取得できるよう端末を改変し，実環境においてどのようなブロードキャストインテントが発行されているのかを調べる．またアプリケーションやウィジェットの apk ファイルから Manifest を解析することで，各ブロードキャストインテントのレシーバ登録数を調査する．次に，各ブロードキャストインテントを意図的に発行させた時の端末の通信量，CPU 使用率を測定することで，通信や電池消費と関連性の高いブロードキャストインテントを調べる．さらに，その影響の高いブロードキャストインテントとバッテリ消費量の関係を調査することで，アプリケーション，ブロードキャストインテント，電池消費の因果関係や連鎖状況を明らかにしていき，それらの結果から Android 端末のバッテリ消費削減を目指した検討を行う．

2. Android OS

Android は，OS，ミドルウェア，アプリケーション，ユーザインタフェースをセットにしたモバイル端末向けプラットフォームであり，Google 社を中心として開発が行われている．また，2013 年第 3 四半期では全世界のスマートフォン OS の中でも，81.9%とトップシェアを占めている [3]．
図 1 に示すように，Android は Linx カーネルをベースとし，スマートフォンやタブレット端末をターゲットに，それらに適したコンポーネントが追加されている [4]．Linux OS と大きく異なる部分は，独自に開発された Android の Runtime である Dalvik 仮想マシンを搭載している点である．その上にアプリケーション・フレームワーク，アプリケーションが乗る形態であるため，アプリケーションは Dalvik 仮想マシンに合わせて開発すれば，直感的な操作性に優れた UI を利用することができ，移植性も高い．

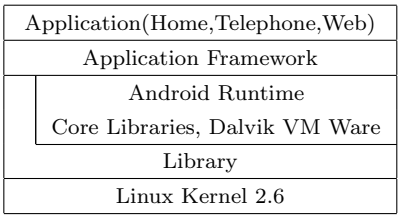

図 1 Android のアーキテクチャ

さらに Android は，無償で提供される開発環境において構築することができ，オープンソースである点からも対応アプリケーションが開発しやすく数も増えるというメリットがある．また Android はキャリア間の制約がないため，アプリケーション開発においても自由度及び汎用性が高いだけでなく，一度マーケットに登録すると，世界中の Android ユーザからインストールが可能となる．現在 Android マーケットでは，このような大きなビジネスチャンスを提供されているため，毎年多くのアプリケーションが登録されており，アプリケーション市場は賑わっている．

Android マーケットの存在により，ユーザから見てもアプリケーションの入手は容易である．Dalvik 実行形式のバイトコードの状態で配布されているため，必要なアプリケーションをインストールして，スマートフォンを自由にカスタマイズできる．広告から収益を得ることによりアプリケーション自体は無償で提供されているものも多く，気軽にインストールして利用できる．

よって本研究では，数あるスマートフォン OS の中でも，これらのサービスを提供するシステムプラットフォームとしての Android に焦点を当て，Android アプリケーションと Android 端末のバッテリ消費の関係について検討していく．

3. ブロードキャストインテント

本研究では，この端末無操作時における高頻度通信の原因として，アプリケーションや OS が発行するインテントに着目した．
インテントとは，Android 特有の仕組みで，アプリケーションの中の一つ一つの機能，例えばアプリケーション同士やアプリケーションとウィジェット，アプリケーションとシステムなどを橋渡しするものである．インテントには，明示的インテント，暗黙的インテント，そしてブロードキャストインテントの三種類が存在する．
明示的インテントと暗黙的インテントは，主にユーザがアプリケーション内のボタンをタップしたときなどに発行さ

れ，次の特定のアクティビティが受信することで画面が遷移するものである．
それに対して，ブロードキャストインテントは，例えば，ACTION_TIME_TICK(現在時刻が変わった) や，ACTION_BATTERY_CHANGED(バッテリ状態が変化した) などのイベントが発生した時に，主にシステムが発行するものであり，登録されている任意の複数のレシーバが受信し，各々の処理を実行させるものである．
つまりこのブロードキャストインテントは，端末がスリープに入っている時にも発行されることから，ユーザだけでなくアプリケーションでさえも意図せず発行している可能性があると考えられる．
よって，本研究ではブロードキャストインテントと電池消費に因果関係があると予想した．

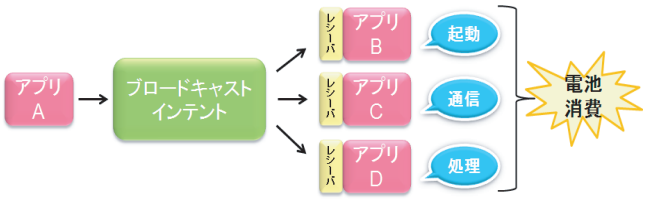

図 2 アプリケーション，ブロードキャストインテント，電池消費の因果関係

図 2 において，本研究で検討を行うアプリケーションとブロードキャストインテントと電池消費の因果関係を説明する．まず，アプリ A がトリガーとなって，あるブロードキャストインテントが発行されるとする．次にそのブロードキャストインテントをレシーバ登録しているアプリ B，アプリ C，アプリ D が存在すると仮定すると，それらのアプリがそのブロードキャストインテントを受信したことで，起動したり通信を始めたり，ディスク書き込みや各々の処理を実行することになる．そしてこれが結果的に端末の電池消費につながるという構図である．

またこのように，あるアプリケーションやブロードキャストインテントがきっかけとなって，ある時間に通信が集中してしまうなどという問題に対処している研究 [5] も行われている．
しかしながら，図 2 の中でも左側の，アプリケーションとブロードキャストインテント，レシーバの関係についてはまだ明らかになっていないことが多いため，これらの関係を深く解析していくことは大変興味深く有意義であると考えられる．

以上のことから本研究では，OS 側から発行されるブロードキャストインテントを取得し解析を行うことで，これらの因果関係についてさらに詳しく検討していく．

4. 解析方法

本節では，ブロードキャストインテントの取得方法について説明する．Android4.0.3 を対象に OS のソースコード内にデバッグメッセージを入れ，解析を行う．具体的には，”framework/base/core/java/android/content/Intent.java”において関数の中にデバッグメッセージを挿入し，発行されたブロードキャストインテントが入っている変数 mAction を取得する．このようにブロードキャストインテントを取得できるように改変した後，ソースコードを再コンパイルし，Android 実機に導入する．
そして実機において，このデバックメッセージを Android の Shell 上で Logcat により表示させることで，測定期間内にどのインテントが何回発行されたかを解析する．

5. 各環境におけるブロードキャストインテントの発行数の調査

ハードウェアが同じ場合でも，その端末にインストールされているアプリケーションや通信状況によってバッテリ消費量も変化する．さらに，Wi-Fi 通信時におけるアプリケーションごとの単位時間あたりのバッテリ消費量や発行されるブロードキャストインテントの種類も異なるということをすでに確認した [6]．
本稿では，さまざまなアプリの中でも SNS やメーラ・ブラウザなどのいわゆる「アプリケーション」と呼ばれるものと，ホーム画面に常駐しニュースや天気の情報を受信して画面に描画する「ウィジェット」によって振舞いが異なることを想定し，これらの違いに着目し測定を行った．
また，よりユーザの実利用環境に近づけるために，Android 端末に SIM カードを挿入し 3G(LTE) 通信をさせ，さらに条件に端末の操作と移動を加えた各環境において，どのようなブロードキャストインテントが多く発行されているかを調査した．

表 1 評価条件

		操作	移動
アプリケーション	ア	無	無
	イ	有	無
	ウ	無	有
	エ	有	有
ウィジェット	オ	無	無
	カ	有	無
	キ	無	有
	ク	有	有

5.1 測定環境

端末にアプリケーション 10 個をインストールした場合 (ア〜エ) と，ウィジェット 10 個をインストールした場合 (オ〜ク) の 2 通りを用いた．
表 1 に端末の操作と移動の有無に関する評価条件を示す．

5.2 測定結果

表 2，表 3 に取得したブロードインテントの一覧を示す．BATTERY_CHANGED や TIME_TICK，SIG_STR，ANY_DATA_STATE，CONNECTIVITY_CHANGE などが評価条件に関わらず比較的多く発行されていることが分かる．
さらに，SCREEN_ON，SCREEN_OFF や各アプリケーションのアップデートに関連するインテントは無操作時に比べて操作時の方が発行数が多い傾向にあることがわかった．移動性においては操作性ほどはっきりとした傾向は見られなかったが，アプリケーションにおいては，「エ」の操作時＋移動時における総発行数が最も多い．
また，全体的にアプリケーションよりもウィジェットの方が発行されるブロードキャストインテントが少ないことが分かった

6. レシーバ登録数の調査

次に，これらのブロードキャストインテントのレシーバ登録数について調査を行った．
レシーバ登録数を調べる方法は二種類あり，一つ目は実際のアプリのコード内に書かれている内容を見ること，二つ目は，Android Manifest に書かれている内容を見ることである．前者は，オープンソースのアプリケーションでない限り調査することは困難であるので，本研究では後者の手法においてのみ解析を行った．
まず，Google Play Store[7] から，ランキング上位のアプリケーションとウィジェットを任意で各 100 個ずつインストールし，apk ファイルをアンジップする．さらに，ファイル内の Android Manifest をバイナリファイル形式からテキストファイル形式に変換し解析を行い，レシーバに登録されているブロードキャストインテントをトラッキングする．
表 4 は，アプリケーションとウィジェットの各 100 個のうち，レシーバ登録されているブロードキャストインテントの上位 20 位までを示したものである．

アプリケーションとウィジェットに共通して多く登録されていたのは，BOOT_COMPLETED，APPWIDGET_UPDATE と INSTALL_REFERRER である．第 3 節で述べたように，これらのレシーバ登録されたインテントが発行されると，それを受け取ったアプリケーションなどが何らかの動作を行い電池消費に繋がると考えられる．

表 2 ブロードキャストインテントの発行数 (アプリケーション)

ブロードキャストインテント	ア	イ	ウ	エ
APPWIDGET_UPDATE	0	30	0	36
ACTION_POWER_CONNECTED	11	8	0	0
ACTION_POWER_DISCONNECTED	9	11	6	6
ANY_DATA_STATE	9552	6008	6712	7920
BATTERY_CHANGED	5335	5578	8604	9915
CLOSE_SYSTEM_DIALOG	0	98	28	72
CONNECTIVITY_CHANGE	464	782	756	483
CONNECTIVITY_CHANGE_IMMEDIATE	51	107	107	67
DATE_CHANGED	0	5	6	6
DROPBOX_ENTRY_ADDED	26	23	9	15
GPS_ENABLED_CHANGE	0	24	4	80
HDMI_PLUGGED	0	5	1	1
MAIN	0	76	14	18
POLL_ACTION	108	210	238	292
RINGER_MODE_CHANGED	0	0	0	22
SYNC_ALARM	50	116	112	168
SCREEN_OFF	83	169	36	140
SCREEN_ON	44	92	13	94
SERVICE_STATED	3426	1950	2214	2793
SIG_STR	2883	3681	5319	6921
TIME_TICK	5992	6106	9763	11734
USB_STATE	12	18	5	5
UMS_CONNECTED	1	2	0	0
UMS_DISCONNECTED	1	2	1	1
USER_PRESENT	0	45	0	57
RINGER_MODE_CHANGED	0	0	0	22
settings.LOCATION_SOURCE_SETTINGS	0	6	1	20
settings.SYNC_SETTINGS	11	27	25	45
calendar.APPWIDGET_SCHEDULED_UPDATE	2	10	10	12
calendar.APPWIDGET_UPDATE	2	4	4	6
calendar.intent.CalendarProvider2	6	0	0	6
NETWORK_STATS_POLL	72	160	168	208
NETWORK_STATS_UPDATED	2341	1490	1610	1996
NetworkTimeUpdateService.action.POLL	4	0	0	12
ThrottleManager.action.POLL	216	420	476	584
facebook.abtest.action.UPDATE_CACHE	2	2	0	2
facebook.AnalyticsEventUploader.ACTION_ALARM	40	93	86	122
facebook.common.appstate.peers	164	86	14	24
facebook.WakingExecutorService.ACTION_ALARM.com.f	97	184	445	238
facebook.FbSharedPrefsProvider.CHANGED_ACTION	572	498	538	386
facebook.orca.database.ACTION_ALARM	0	15	0	5
facebook.orca.notify.ACTION_NEW_FOLDER_COUNTS	13	7	2	3
facebook.presence.ACTION_PRESENCE_RECEIVED	0	8	1	4
facebook.push.mqtt.ACTION_CHANNEL_STATE_CHANGED	91	164	265	151
facebook.push.mqtt.ACTION_MQTT_PUBLISH_ARRIVED	0	8	1	4
google.android.apps.chrome.omaha.ACTION_REGISTER_REQUEST	0	7	3	3
skype.action.BACKGROUND	47	105	110	137
skype.action.FOREGROUND	163	362	385	478
twitter.android.poll.alarm	38	84	86	153
twitter.android.poll.data	20	34	30	54
twitter.androidAVATARS_CHANGED	1	0	1	4
yahoo.mobile.android.swupdate.refresh	20	6	0	5
yahoo.mobile.client.weather.action.ACTION_LOCATION_UPDATE	0	14	1	7
yahoo.mobile.client.weather.action.ACTION_PHOTO_DOWNLO	0	37	0	28
yahoo.mobile.client.weather.action.ACTION_WEATHER_UPDATE	0	42	0	40
yahoo.mobile.client.weather.action.REFRESH	0	38	4	27
jp.naver.line.android.legy.SpdyHeartbeatChecker.sendPing	0	9	0	0

表 3 ブロードキャストインテントの発行 (ウィジェット)

ブロードキャストインテント	オ	カ	キ	ク
APPWIDGET_UPDATE	240	332	343	265
ACTION_POWER_CONNECTED	4	6	5	0
ACTION_POWER_DISCONNECTED	3	3	4	4
ALARM_CHANGED	0	8	6	4
ANY_DATA_STATE	1992	1920	1120	1854
BATTERY_CHANGED	4629	4165	3172	3742
CLEAR_DNS_CACHE	0	1	1	1
CLOSE_SYSTEM_DIALOG	12	48	3	16
CONNECTIVITY_CHANGE	259	237	193	294
CONNECTIVITY_CHANGE_IMMEDIATE	56	52	44	63
DATE_CHANGED	5	5	6	5
DROPBOX_ENTRY_ADDED	68	96	88	92
MAIN	0	6	0	0
SCREEN_OFF	38	98	17	171
SCREEN_ON	52	85	17	153
SERVICE_STATED	621	450	211	518
SIG_STR	1878	1242	771	1317
SYNC_ALARM	64	64	65	68
TIME_SET	0	26	13	0
TIME_TICK	7520	6751	5901	6238
USB_STATE	14	14	19	5
UMS_CONNECTED	2	3	3	1
UMS_DISCONNECTED	1	1	1	1
USER_PRESENT	8	64	0	32
GPS_ENABLED_CHANGE	38	52	48	106
RINGER_MODE_CHANGED	0	0	7	1
POLL_ACTION	284	276	270	292
settings.LOCATION_SOURCE_SETTINGS	10	15	22	44
settings.SYNC_SETTINGS	9	8	9	8
calendar.APPWIDGET_SCHEDULED_UPDATE	9	15	11	9
calendar.APPWIDGET_UPDATE	4	6	6	4
deskclock.ALARM_ALERT	0	4	3	2
calendar.intent.CalendarProvider2	6	10	12	12
NETWORK_STATS_POLL	196	192	193	204
NETWORK_STATS_UPDATED	542	416	254	476
NetworkTimeUpdateService.action.POLL	12	4	5	8
ThrottleManager.action.POLL	568	548	533	580
facebook.orca.notify.ACTION_NEW_FOLDER_COUNTS	13	7	2	3
google.android.gms.icing.INDEX_RECURRING_MAINTENANCE	3	3	3	3
google.android.gms.recovery.WAKEUP	5	1	1	1
google.android.intent.action.SEND_IDLE	5	52	12	31

7. ブロードキャストインテント発行時における CPU 使用率測定

そこで，これらのブロードキャストインテントが連続して多く発行された時の端末の CPU 使用率の測定実験を行った．

7.1 測定方法

今回は以下に示すように，第 5 節において比較的多く発行されていたものと，さらに第 6 節においてレシーバ登録数の多かったものの計 12 個のブロードキャストインテントを測定対象とした．

- APPWIDGET_UPDATE
- BOOT_COMPLETED
- INSTALL_REFERRER

表 4 レシーバ登録数 (アプリケーション 100 / ウィジェット 100)

Top	アプリケーション		ウィジェット	
	インテント	登録数	インテント	登録数
1	BOOT_COMPLETED	43	APPWIDGET_UPDATE	96
2	c2dm.RECEIVE	32	BOOT_COMPLETED	26
3	INSTALL_REFFERRER	30	INSTALL_REFERRER	11
4	APPWIDGET_UPDATE	30	USER_PRESENT	8
5	c2dm.REGISTRATION	28	PACKAGE_ADDED	8
6	CONNECTIVITY_CHANGE	26	PACKAGE_REMOVED	7
7	PACKAGE_ADDED	18	APPWIDGET_DISABLED	6
8	PACKAGE_REMOVED	16	DATE_CHANGED	5
9	PACKAGE_REPLACED	15	BATTERY_CHANGED	5
10	POWER_CONNECTED	12	c2dm.REGISTRATION	4
11	LOGIN_ACCOUNTS_CHANGED	12	c2dm.RECEIVE	4
12	LOCALE_CHANGED	11	TIME_SET	4
13	PURCHASE_STATE_CHANGED	10	TIMEZONE_CHANGED	4
14	RESPONSE_CODE	10	PACKAGE_REPLACED	4
15	SMS_RECEIVED	9	VOLUME_CHANGED	3
16	USER_PRESENT	9	POWER_DISCONNECTED	3
17	MY_PACKAGE_REPLACED	9	POWER_CONNECTED	3
18	MEDIA_MOUNTED	8	strCalendar.SET	2
19	POWER_DISCONNECTED	8	GET_VERSION	2
20	PHONE_STATE	6	CONNECTIVITY_CHANGE	2

- c2dm.RECIEVE
- TIME_TICK
- BATTERY_CHANGED
- SIG_STR
- CONNECTIVITY_CHANGE
- SCREEN_ON
- SCREEN_OFF
- ANY_DATA_STATE
- c2dm.REGISTRATION

Android 端末に任意のアプリケーション 10 個がインストールされた状態 (A) と，同様に任意のウィジェット 10 個がインストールされた状態 (W)，さらにこれらのアプリケーションとウィジェットが両方インストールされた状態 (A+W) の 3 パターンにおいて Android の Shell 上で am コマンドを用いて，各ブロードキャストインテントを意図的に約 10 分間連続的に発行する．また，それと同時に”/proc/stat/”から CPU の使用時間を取得し，その値から CPU 使用率を算出する．

7.2 測定結果

今回の測定結果としては，時間的な変動はみられなかったので測定中の安定時における値を以下の図 3 に示す．「user」は優先モードでユーザレベルのアプリケーションによる CPU の使用率，「nice」は低優先モードで優先度 (ナイス値) によるユーザレベルの CPU 使用率，「system」はシステムモードでシステムレベル (kernel) の CPU 使用率，「idle」はアイドルモードでディスク I/O 待機時間を除く

CPU のアイドル時間割合を示している．

グラフより，インテントを発行していないデフォルト時には idle の割合が高くなっていることがわかる．それに対し，各ブロードキャストインテントを発行した時には user の割合が増加している．これはブロードキャストインテントが発行されたことによって，さまざまなアプリケーションやウィジェットが誘発されたからだと考えられる．

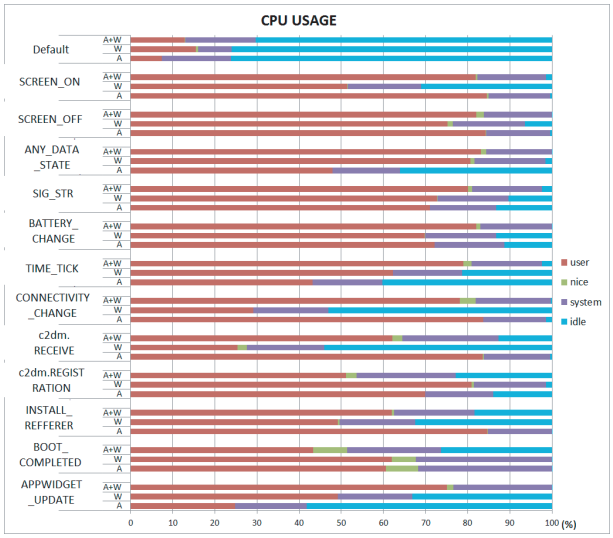

図 3　CPU 使用率

8. ブロードキャストインテント発行時における通信量測定

第 7 節と同様にブロードキャストインテントが連続して多く発行されたときの通信量を測定した．

8.1 測定方法

発行するブロードキャストインテントや使用する端末の状況とパターンは前節と同様である．この環境において同様に am コマンドで各ブロードキャストインテントを発行した時の通信量 (受信量と送信量の和) を”/proc/net/dev”から取得する．

ただし，これらのブロードキャストインテントはシステムの動作の都合上，単位時間当たりに発行できる回数がそれぞれ異なるため，各ブロードキャストインテントの 1 発行ごとの通信量を評価した．

8.2 測定結果

図 4 にその結果を示す．グラフより，1 発行あたり通信量に最も大きな影響を与えるブロードキャストは BOOT_COMPLETED であり，次いで APPWIDGET_UPDATE，c2dm.RECIEVE であることが分かった．BOOT_COMPLETED はシステムの起動が完了した時に発行され，例えばシステムの起動と同時にウィジェットなどが天気予報やニュースを取得するために通信を行うなどの処理がされる．

また，これらのブロードキャストインテントは第 6 節においてレシーバ登録数の割合が多かったものであることから，頻繁に発行される可能性のあるブロードキャストインテントが多くのアプリケーションにおいてレシーバ登録されている場合，通信量の増加に大きな影響を与えてしまい，結果的に端末のバッテリ消費につながると考えられる．

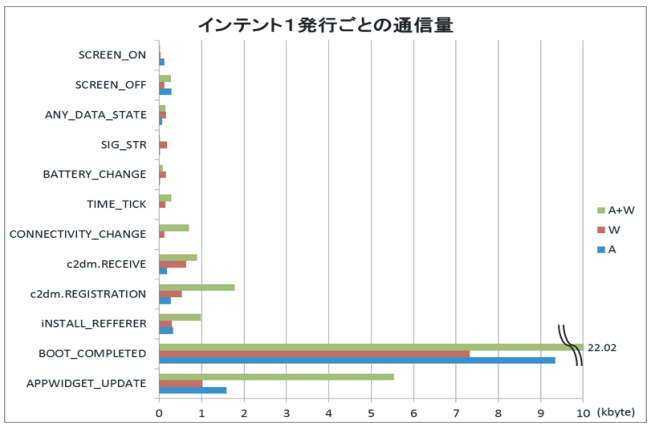

図 4　1 発行あたりの通信量

9. BOOT_COMPLETED 発行時における評価

第 7 節の結果より，BOOT_COMPLETED を発行することによって通信量が大幅に増加することが確認された．本節では，この BOOT_COMPLETED 発行時のバッテリ消費量と電圧変動を測定し，さらに発行されたインテントを取得することで，BOOT_COMPLETED と端末の電力消費の相関関係を分析する．

BOOT_COMPLETED は前述の通り，端末の起動時に発行されるものであるため，本来は連続的に多く発行されることは現実的ではないが，今回の測定ではあくまでもブロードキャストインテントとバッテリ消費の因果関係を明らかにする上で，最もその傾向が現れやすいという点でまずは BOOT_COMPLETED の発行時に着目した．

9.1 測定方法

前節までの測定ではブロードキャストインテントを意図的に発行するために，端末と PC を USB 接続し，接続先の PC のターミナルから am コマンドを用いていたが，この手法だと常に端末が充電状態になってしまい，バッテリの使用状況を正しく測定することが不可能であった．

よって今回の測定では，端末を PC と USB 接続させることなく am コマンドによりブロードキャストインテントを発行するために，Android の Shell 上で動作するシェルスクリプトを作成し，そのプロセスを常駐させるようにした．また，端末のバッテリ容量を”/sys/class/power_supply/battery/capacity”から取得するシェルスクリプトと，端末の電圧変動

を”/sys/class/power_supply/battery/voltage_now”から取得するシェルスクリプトを作成し，同時に動作させることで端末の情報を取得している．さらに，第 4 節で紹介した方法を用いて，logcat により発行されたブロードキャストインテントのログ情報を取得した．

9.2 測定結果

図 5 と図 6 に示すのは，それぞれ BOOT_COMPLETED 発行時における端末のバッテリ消費量と電圧変動の時間的遷移である．アプリケーションがインストールされた状態 (A)，ウィジェットがインストールされた状態 (W)，その両者がインストールされた状態 (A+W) のそれぞれについて，デフォルトの場合とインテントを発行した場合との比較を行った．両者ともインテントを意図的に発行していないデフォルト時に比べ，インテントを発行した時には時間が経過するとともに大きく低下している．
図 5 のバッテリ残量に関しては，BOOT_COMPLETED を約 2 時間続けて発行すると端末の電池容量を使い果たしてしまうほどの影響力があることが分かった．

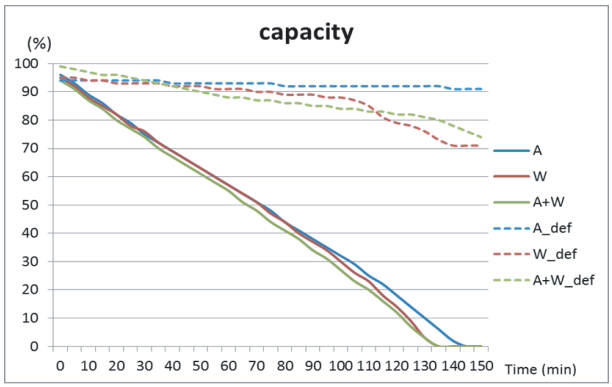

図 5 バッテリ消費量

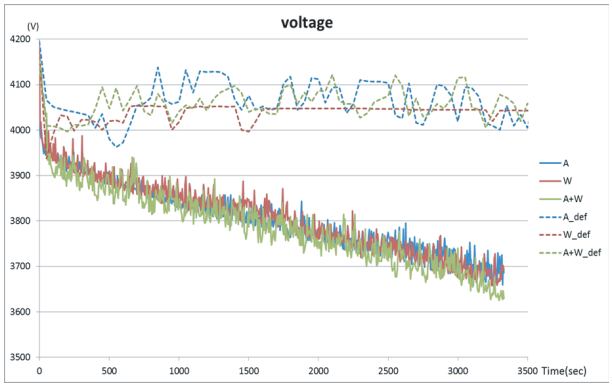

図 6 電圧変動

ここで，logcat を用いて取得したブロードキャストインテントの発行数の一部を図 7 に示す．BOOT_COMPLETED を意図的に発行しているときの方が，発行していないデフォルト時に比べて，発行数が増加しているブロードキャストインテントが多く観察された．
特に，ウィジェットがインストールされている端末においては APPWIDGET_UPDATE がデフォルト時の約 16 倍多く発行されていることがわかる．これは，BOOT_COMPLETED を意図的に発行したことにより，他のインテントが誘発され，さらにそれによりまた別のインテントが誘発されるなどの連鎖反応が生じているのだと考えられる．
これらのブロードキャストインテントの連鎖反応が図 5 や図 6 に示した電池の消耗に起因しているのだと考察できる．

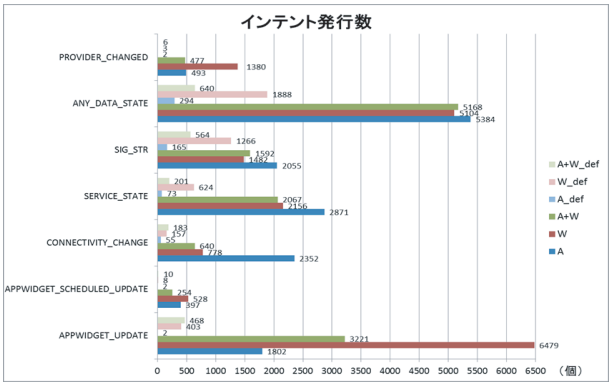

図 7 logcat で取得したインテントの一部

10. まとめと今後の課題

本研究では，Android 端末のバッテリ消費削減を目指してアプリケーションを連鎖させるブロードキャストインテントと電力消費の関係の解析を行った．測定結果より，ブロードキャストインテントを意図的に連続して発行させた時の CPU 使用率が，何も発行していないデフォルト時に比べて，いずれのインテント発行時に関しても idle モードより user モードの割合が高くなった．このことから，ブロードキャストインテントの発行がアプリケーションやウィジェットを誘発させる要因になると考えられる．
また特に，BOOT_COMPLETED インテントを意図的に発行した時には通信量とバッテリ消費量の大幅な増加が観測され，それと同時に誘発される他のブロードキャストインテントが多く存在することを確認した．
これは BOOT_COMPLETED の発行によって，さまざまなアプリケーションやウィジェットが端末起動時の初期設定やデータ通信を行うなどの処理を行い，それが結果的に端末のバッテリ消費に繋がったのだと考えられる．このことから，ブロードキャストインテントの中でも少なくとも BOOT_COMPLETED はあらゆるアプリケーションや機能を連鎖させる可能性があり，結果的に端末のバッテリ消費に何らかの影響を与えると考察できる．よって，BOOT_COMPLETED インテントを必要以上にレシーバ登録したりするアプリケーションをインストールすることでバッテリを消費してしまう可能性が高くなるという判断をすることができるようになる．

さらに，ユーザが端末を移動・操作しているときには比較的多くのインテントが発行されることが分かったが，その移動や操作によって様々なブロードキャストインテントの発行が促進され，アプリケーションなどや反応し，結果的に電池消費に繋がるのではないかと推測される．

今後の課題として，第 9 節でも述べたようにBOOT_COMPLETED が多く発行されることは一般には現実的でないため，実環境においてより発行数の多いブロードキャストインテントとバッテリ消費の相関関係を調査する必要があると考えられる．

また、それらのブロードキャストインテントがアプリケーションやウィジェットを相互連鎖させている可能性があるので，その連鎖の様子や影響を細かく解析するために，端末にインストールするアプリケーションやウィジェットの数や種類を変化させた環境において通信量やバッテリ消費の測定を行いたい．

さらに，最終的にはそれらの相関係数などを算出することで，その情報を元にして各ユーザのアプリケーションインストール時に，バッテリや通信にどの程度影響を与えるものかの推定値を提示するようなツールの開発につなげていきたい．

謝辞

本研究を進めるにあたり，ご強力賜りました株式会社 KDDI 研究所の磯原隆将さんと UCLA の高井峰生先生に深く感謝致します．

参考文献

[1] BCN:http://www.bcnranking.jp/news/gallaery/1210/
[2] Carroll, Aaron, and Gernot Heiser. ”An analysis of power consumption in a smartphone.”, Proceedings of the 2010 USENIX conference on USENIX annual technical conference. 2010.
[3] Gsartner:http://www.gartner.com/it/page.jsp?id=2237315
[4] Android developers:http://developer.android.com
[5] 川崎 仁嗣，神山 剛，稲村 浩:「Android OS の状態変化通知機構における通信集中回避制御手法の検討」，GN・CDS 合同研究発表会，Vol.2013-GN-86, No.20, pp.1-8, 2013 年 1 月．
[6] 早川 愛，磯村 美友，竹森 敬祐，山口 実靖，小口 正人：「Android 端末におけるブロードキャストインテント情報を用いた省電力化に関する一検討」，DEIM2014，E8-1, 2014 年 3 月．
[7] Google Play:https://play.google.com/store?hl=ja
[8] Takeshi Kamiyama, Hiroshi Inamura, Ken Ohta:”A Model-based Energy Profiler using Online Logging for Android Applications”, Mobile Computing and Ubiquitous Networking (ICMU),2014 Seventh International Conference, pp.7-13, January 2014.