IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-7, July 2018 E-ISSN: 2347-2693

Analysis of Radio Resource Energy Consumption Pattern in Cellular Network

S. Pandikumar [1*], G. Sujatha [2], M. Sumathi [2]

[1]Research Scholar, Madurai Kamaraj University, India
[2]Department of Computer Science, Sri Meenakshi Govt Arts College for Women, Madurai, India

**Corresponding Author: spandikumar@gmail.com*

Available online at: www.ijcseonline.org

Accepted: 18/Jul/2018, Published: 31/July/2018

Abstract —In the wireless communication especially in cellular network, energy efficiency is a major concern among other issues, specifically in high-speed data network i.e. 3G/4G. A Smartphone is a widely used handheld device to surf data in 3G/4G. Unfortunately, one of the constraints of Smartphone is having limited battery backup, which always makes the user inconvenience. Increasing of battery backup is not merely a solution to increase the user experience, because the growth of mobile technology is 25% per year but at the same time battery capacity increases 10% per year; it's not a balanced. So that the researchers essentially focus on energy efficient development to extend battery life. This paper analyzes various factors influences of power consumption and their characteristics (like RRC state transitions, inactivity timer setup and screen On/Off) and the paper also reviews the proposals of energy-aware developments through real-time measurements.

Keywords—RRC State Transition, Inactivity Timers, Keep-Alive Messages, Energy Efficient Strategies, Energy aware 3G/4G.

I. INTRODUCTION

In the 21st century, energy efficiency is almost the slogan of every field. Smartphone has become a very popular gadget today; not only a gadget but also the people consider this as their PA (Personal Assistant). Most of the consumer services like banking, internet, multimedia etc. are availed in Smartphone's. In high speed data network like 3G/4G requires high energy cost than the previous one [1] and so the user's equipment lost their battery very often .The entire network efficiency and user's experiences are fully depending on battery backup. So, energy efficiency is the very important factor for Smartphone development. The increase of battery backup is not a suitable option for improving Smartphone's operational time because high end software and high speed network drain power enormously.

3G networks consume more significant energy than 2G network and give the user better experience than the other one. In WCDMA technology, Total transmission energy is entirely related to its resource allocation, because radio resources are controlled by RRC (both 3G/4G networks). The RRC (Radio Resource Controller) is the main entity for radio resource management and it controls other entities of RNC (Radio Network Controller) like RLC, MAC, PDCP and BMC [1] (Figure 1). Based on the traffic, RRC creates a radio bearer with UE (user equipment) using logical channels and maintain a connection states and this state maintains both in UE and RNC. Each state consumes significant amount of energy. CELL_DCH consumes higher energy than CELL_FACH and PCH channels [3]. The state of the UE directly depends on the user traffic and also the characteristic of generated and received traffic of the UE is directly coupled with RRC state transition on the UTRAN (Figure 2 and Figure 3).

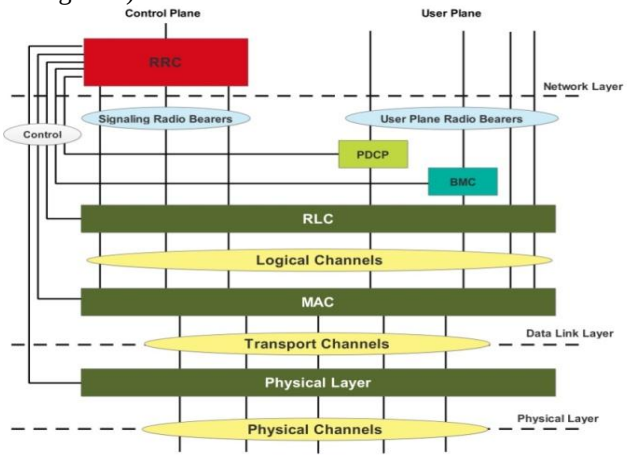

Figure 1. Relationship of RRC with radio interface entities

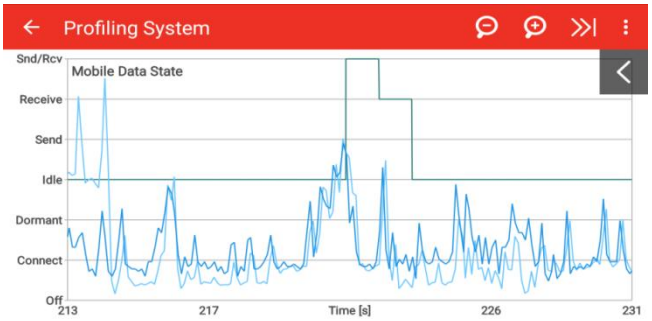

Figure 2. RRC states during data transmission

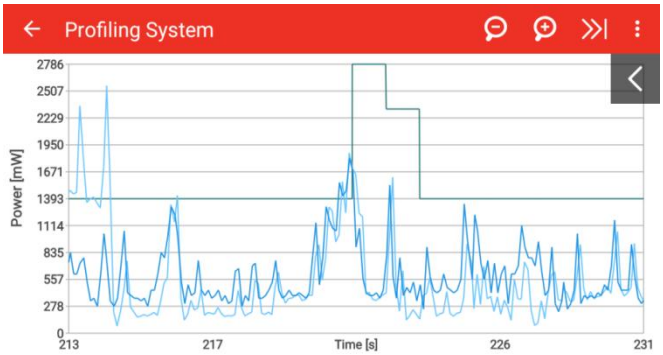

Figure 3. RRC states with related power consumption

This paper analyzes various parameters, which influences the network's power consumptions like inactivity timers, fast dormancy and screen On/Off status. This paper reviews four researches, based on the energy circumstances and highlights the results of their experiments. All these papers insist, should configure inactivity timers effectively and manage keep-alive messages. All methodologies and working procedures of the 3G and 4G network quoted in this paper are given or proven by real time measurements and images. Rest of the paper is organized as follows, Section I contains the technical foundation of high-speed mobile network like 3G and 4G , Section II contain the technical background of smartphone apps and states with real time measurements Section III contain the architecture of 3G network and correlation between the always-on apps and mobile network. This part elaborately discusses all the aspects of inactivity timers, state transition and how always-on apps control the states with results. Section IV contains the experimental setup and its details and Section VIII concludes research work with future directions.

II. INFLUENCE OF ALWAYS-ON APPLICATIONS

The major energy consuming unit of the mobile phone is categories into

- Display Unit
- Core Processing Unit
- Network Unit

Those three are the logical categories of energy consuming units. Display and processing units are apart from our research area. The network unit of the mobile phone is the most energy consuming unit [2, 3]. The core network operations are Call Management, Cell Updates, and Paging etc. All of the above operations are performed by mobile operating system and service provider by default without the user influences. Apart from this the network unit is utilized or controlled by Mobile Apps for their transactions.

Apps are most important one in smartphone. The Apps is nothing but the special program for smartphone designed for particular operations. All the Apps should respond to the user instruction and perform but at the same time it does something by its own. For example Apps update themselves, send data to servers, and messenger apps always connect with their server etc [4].

The study [3] exposes 61% of the apps usage energy is spending in their IDLE states. This data confirms the mobile apps are consuming energy un-necessarily without the knowledge of the users. The data collected from 403 applications from 23 different categories (entertainment, sports, utility, life style etc). These apps size have ranged from 1.6 KB to 18 MB. The idle time energy consumption is measured by the mechanism of splitting the App state into three categories (Figure 3) and they are

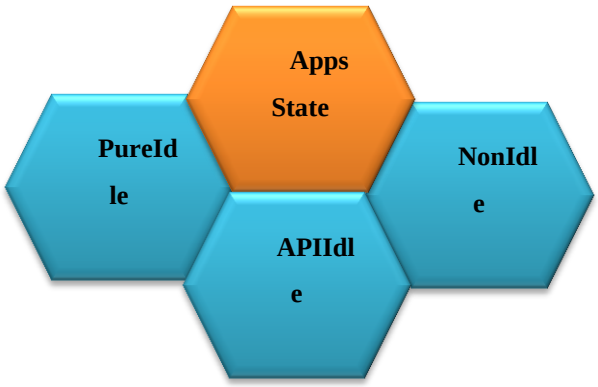

Figure 4. Apps State

PureIdle state means the app is waiting for user input or asynchronous sensor data. During this time no app code is running, but the app is still consuming energy.

APIIdle means app waiting due to sleep or wait for API calling.

The result of PureIdle, APIIdle and NonIdle consumes energy of 36.6%, 25.0%, and 38.4% respectively [3] (Figure 4). The results insist mere code optimization of the applications is not sufficient to reduce the overall app energy consumption.

Calculation on each of these three categories as follows: First, we calculate APIIdle as the sum of the energy for all *java.lang.Thread.sleep* and *java.lang.Object.wait* series of APIs. The energy of an API is the sum of energy samples

between its starting and ending timestamps. Second, we calculate the NonIdle energy as the sum of the energy of all execution paths minus the APIIdle energy. The energy of a path is the sum of all energy samples between the entry and exit timestamps of the path. This includes the energy of sleeping APIs called along the path so we subtract their energy from the summed value to get NonIdle. Lastly, we calculate PureIdle as the total energy minus APIIdle and NonIdle. PureIdle represents all the energy that has been consumed while the application is running but not caused by any code of the application.

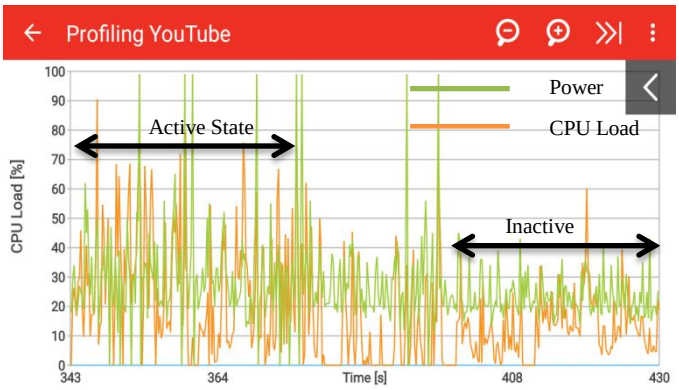

Figure 5. Profiling Youtube while its running and IDLE state

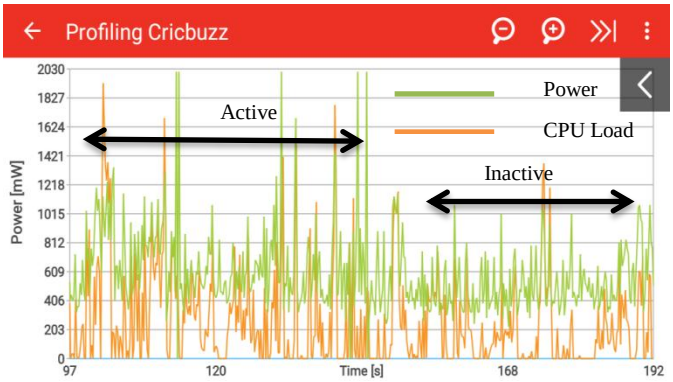

Figure 6. Profiling Cricbuzz while its running and IDLE state

The IDLE state energy consumption is not entirely related to network communication but in some way the App consumes energy unnecessarily (Figure 5 and Figure 6). The experimental data collected from two frequently used Apps YouTube and CricBuzz and it clearly says the average energy consumption of Youtube and Cricbuzz is 810 mW and 905 mW respectively. But both the Apps Youtube and Cricbuzz consume 420 mW and 460 mW respectively at the IDLE state that means one third of active state energy.

III. RADIO STATES AND ALWAYS-ON APPLICATIONS

Energy consumption of always–on application in WCMA networks is more significant. The keep-alive messages of always–on applications highly interact with RRC and this process leads to unacceptably short battery life of mobile phones [4, 5].

The cloud-based mobile applications rely on the Internet connectivity in order to keep their contents up-to-date and present them to the users through notifications. Therefore, the cloud-based applications require toggling the cellular radio module to the active state occasionally to transmit/receive data during the update intervals (Figure 12 and Figure 13). In order to reduce the number of oscillations between the states, the radio module preserves its state for a certain amount of duration after being toggled ON, even if the flow of packets in a user session is completed. Therefore, upon activation, the radio module consumes power at least for a constant period of time. As it is common to run a high number of cloud-based applications in the background simultaneously, high number of requests for toggling the radio module to the full active (i.e., the most power-consuming) state asynchronously would toggle the radio interface ON and OFF at different times, which in sum might increase the total energy consumption.

However, a mechanism that can control the cellular network interface of the smartphone in a way to toggle the mobile data state at particular periodicity with various durations would let all the applications' network activities to be accomplished at fixed and predefined time intervals. In other words, this would generate small bursts that are spaced out with some intervals into rather fewer number of large bursts (a burst is defined as a complete data transfer of any size). If small bursts are spaced out at regular intervals, the device must constantly ramp up and then down, not only draining the battery but introducing a two-second delay for each new connection (Figure 9.). Large bursts are more efficient since they entail fewer promotions and fewer demotions (Figure 14).

The state transition process are occur based on traffic volume and inactivity timers (T1, T2 and T3), each timer set in between state (Figure 7.).

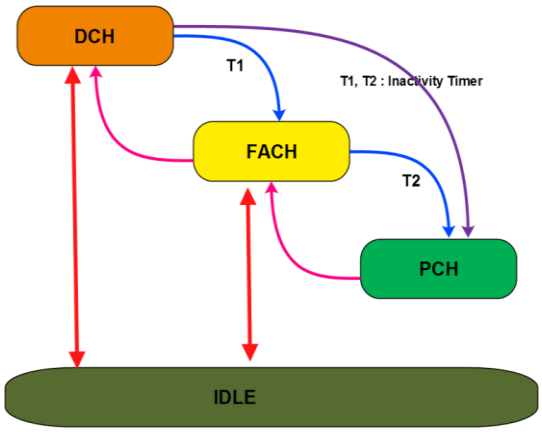

Figure 7. RRC State Transition and Inactivity Timer

The time period of the inactivity timer is set based on network operators and RRC configurations.

T1 is an inactivity timer that is used in the CELL DCH state, and is reset whenever there is traffic. The T1 value may rely on the DCH data rate. In the RNC the default values were 5 seconds for 8–32 kbit/s, 3seconds for 64 kbit/s, and 2 seconds for 128 kbit/s and faster.

T2 is an inactivity timer in the CELL FACH state; the state machine will enter either the CELL PCH state (if used) or idle state after being inactive for T2 seconds. In the same RNC implementation, the default value for T2 was 2 seconds (Figure 10.).

T3 is a timer used in CELL PCH. After staying in the CELL PCH for T3 seconds, the RRC connection will be released. This is typically a very long timer (several minutes or even tens of minutes).

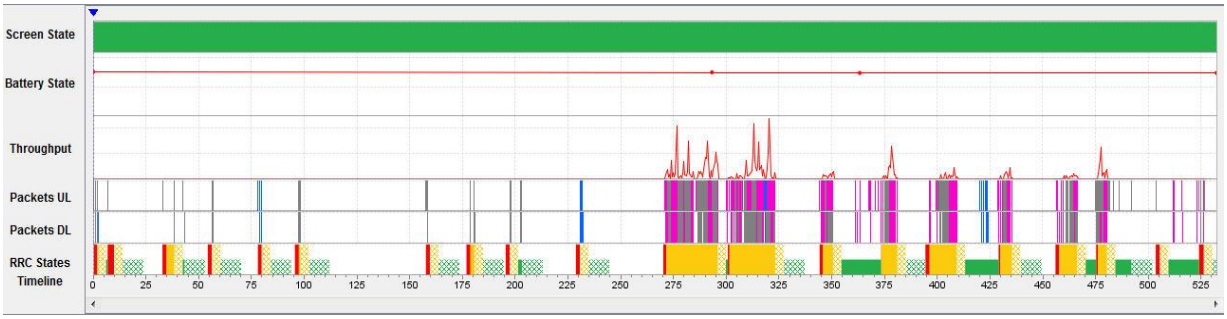

Figure 8. Overall Session Log

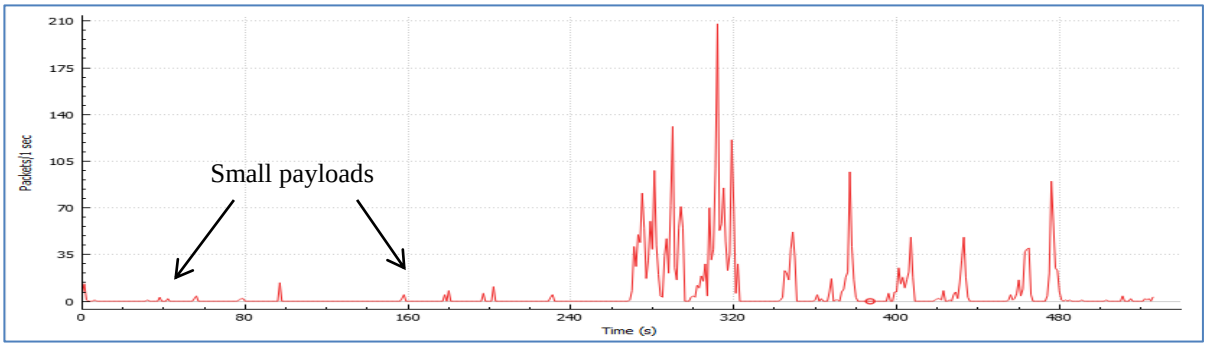

Figure 9. Transmission of packets in the Session

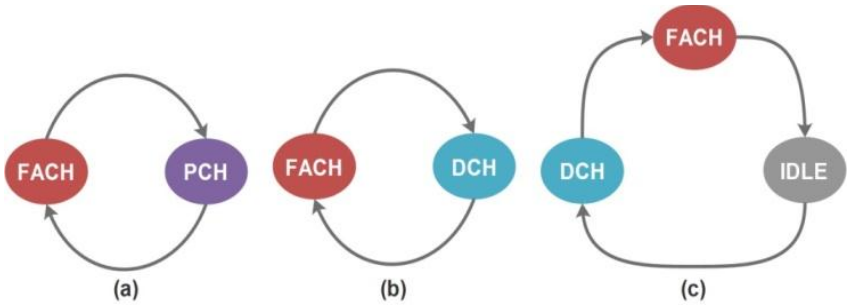

Figure 11. RRC State Transitions

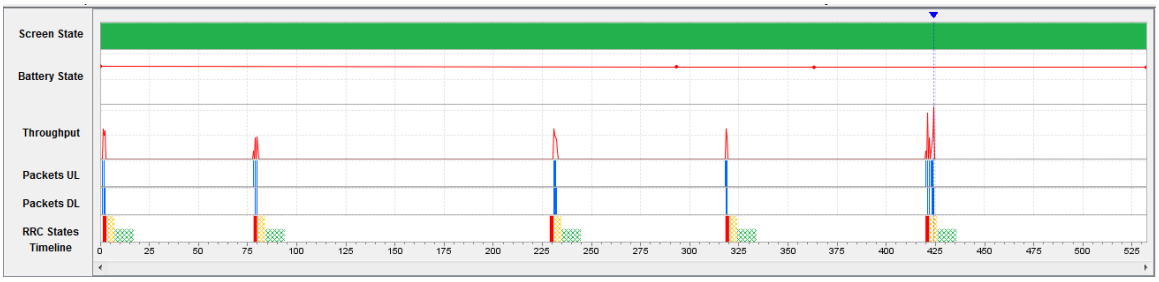

Figure 12. Periodic data transmission and state change of WhatsApp Application

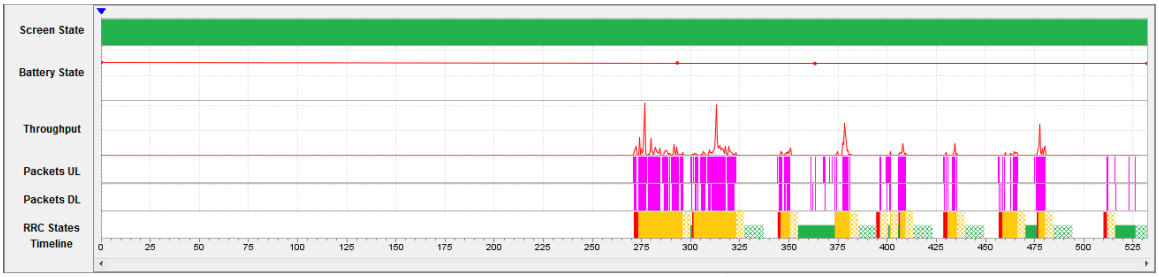

Figure 13. Periodic data transmission and state change of CricBuzz Application

Start Time	Time Elapsed	Bytes	Packet Count	Burst
1.267	1.232	101	8	SvrNetDelay
77.965	1.940	77	5	App
230.647	1.384	101	8	SvrNetDelay
318.050	0.706	77	5	App
419.887	4.107	296	14	App

Figure 14. Packet counts and payloads

File

Network Attribute	Profile Value
Carrier	AT&T
DCH (Active)->FACH (Standby) timer (sec)	5.0
FACH (Standby)->IDLE timer (sec)	2.0
Min IDLE->DCH (Active) promotion delay (sec)	1.5
Avg IDLE->DCH (Active) promotion delay (sec)	2.0
Max IDLE->DCH (Active) promotion delay (sec)	4.0
Min FACH (Standby)->DCH (Active) promotion delay (sec)	0.8
Avg FACH (Standby)->DCH (Active) promotion delay (sec)	3.0
Max FACH (Standby)->DCH (Active) promotion delay (sec)	0.0
RLC threshold for uplink (bytes)	543
RLC threshold for downlink (bytes)	475
Threshold for resetting DCH (Active) timer (bytes)	320
Timing window for resetting DCH (Active) timer (sec)	0.3
RLC consumption rate (^2) for uplink	0.0014
RLC consumption rate (^1) for uplink	1.6
RLC consumption rate (^0) for uplink	20.0
RLC consumption rate (^2) for downlink	0.0

Figure 10. Typical RRC Configuration

During the periodic Always-On messages, the RRC state transitions are

1) If CELL_PCH state enabled in RNC, then the state sequence is IDLE→ CELL_FACH→ CELL_PCH→ CELL_FACH→ CELL_PCH etc (Figure 11. a)

If CELL_PCH is not enabled the RRC connection will created directly to the CELL_FACH then the sequence is CELL_FACH→IDLE→FACH etc (Figure 11.b)

2) If the RRC does not support 1 and 2 the sequence is CELL_DCH→ CELL_FACH→Idle→ CELL_DCH→ CELL_FACH etc (Figure 11.c)

Request/Response View | Packet View | Content View

ID	Time	Direction	Type	Payload Length	TCP Flags
1	1.267249	DOWNLINK	TCP_DATA	37	AP
2	1.267563	UPLINK	TCP_ACK	0	A
3	1.333384	UPLINK	TCP_DATA	3	AP
4	1.999673	DOWNLINK	TCP_DATA_DUP	37	AP
5	1.999979	UPLINK	TCP_ACK_DUP	0	A
6	2.139637	DOWNLINK	TCP_ACK	0	A
7	2.139961	UPLINK	TCP_DATA	24	AP
8	2.499636	DOWNLINK	TCP_ACK	0	A
9	77.965467	UPLINK	TCP_DATA	3	AP
10	78.947041	DOWNLINK	TCP_ACK	0	A
11	78.947470	UPLINK	TCP_DATA	35	AP
12	79.899330	DOWNLINK	TCP_DATA	39	AP
13	79.905094	UPLINK	TCP_ACK	0	A
14	230.646625	DOWNLINK	TCP_DATA	37	AP
15	230.647001	UPLINK	TCP_ACK	0	A
16	230.669528	UPLINK	TCP_DATA	3	AP
17	231.370751	DOWNLINK	TCP_DATA_DUP	37	AP
18	231.477517	UPLINK	TCP_ACK_DUP	0	A
19	231.678687	DOWNLINK	TCP_ACK	0	A
20	231.678923	UPLINK	TCP_DATA	24	AP
21	232.030711	DOWNLINK	TCP_ACK	0	A
22	318.049930	UPLINK	TCP_DATA	3	AP
23	318.413139	DOWNLINK	TCP_ACK	0	A
24	318.417425	UPLINK	TCP_DATA	35	AP
25	318.755497	DOWNLINK	TCP_DATA	39	AP
26	318.756117	UPLINK	TCP_ACK	0	A
27	419.887151	UPLINK	TCP_DATA	3	AP
28	420.906081	DOWNLINK	TCP_ACK	0	A
29	420.907058	UPLINK	TCP_DATA	191	AP
30	421.850632	DOWNLINK	TCP_DATA	31	AP
31	421.850986	UPLINK	TCP_ACK	0	A
32	422.975204	UPLINK	TCP_DATA	3	AP
33	423.322606	DOWNLINK	TCP_ACK	0	A
34	423.323162	UPLINK	TCP_DATA	45	AP
35	423.652577	DOWNLINK	TCP_DATA	23	AP

Figure 15. Transmitted packet details

While testing the WhatsApp itself was sent periodic data—each constituting only 35 bytes or nearby—were being transmitted at 80-second intervals (Avg). Every single KB of

data Consumes 17.2 joule energy. During test period, the application sends 40 packets with 2,756 bytes data (Figure 14 and Figure 15).

Energy consumption of single Always-On message transitions varied between 0.15 mAh and 0.6 mAh in 3G and between 0.11 mAh and 0.13 mAh in 2G [6]. In the above measurements environment is,

- PCH state was enabled
- Inactivity timer values are RRC default and
- Intervals of keep-alive messages between 20 sec to 300 sec (Table 1).

Table 1. Power consumption of Keep-Alive messages

Interval (Sec)	Avg Power in 3G (mA)	Avg Power in 2G (mA)
20	34	29
40	24	16
150	16	9.1
300	14	7.3
infinite	6.1	5.2

The second result produced by changing the values of T2 then the result are highlights that (Table 2) shorter T1 and T2 timers produced better the battery performance of always-on applications. It proves that state transition to PCH state through shorted inactivity time consumes less power than DCH and FACH states [9].

Table 2. Power consumption of different T2 values

T2 (Sec)	CELL_PCH	Avg Current In 3G (mA)	Cost of a single keep alive (mA)
2	Enabled	20	0.15
5	Enabled	30	0.27
10	Enabled	45	0.43
2	Disabled	61	0.61
5	Disabled	74	0.75
10	Disabled	98	1.0

The parameters of RRC state machine not only impacts UE energy consumptions, also network managements and user experiences. State machines are static nature and it treats all traffic with same inactivity timer, making it very difficult to trade-off among radio energy consumption, network management overhead, user efficiency and performance. We use two algorithms for inferring state promotion and demotions [6]. The state promotions are happened based on traffic volume which is measured in RLC buffer and state demotions are happened based on inactivity timers.

Algorithm 1: State Promotion

Step 1: Keep UE on IDLE.
Step 2: UE sends min bytes. Server echoes min bytes.
Step 3: UE sends max bytes. Server echoes min bytes.
Step 4: UE records the RTT Δt for Step 3.
Step 5: Report P1 if $\Delta t \gg$ normal RTT. Otherwise report P2

Algorithm 2: State Demotion

Step 1: for n = 0 to 30 do
Step 2: UE sends max bytes. Server echoes min bytes.
Step 3: UE sleeps for n sec.
Step 4: UE sends min bytes. Server echoes min bytes.
Step 5: UE records the RTT $\Delta t1(i)$ for Step 4.
Step 6: end for
Step 7: for n = 0 to 30 do
Step 8: UE sends max bytes. Server echoes min bytes
Step 9: UE sleeps for n sec.
Step 10: UE sends max bytes. Server echoes min bytes.
Step 11: UE records the RTT $\Delta t2(i)$ for Step 10.
Step 12: end for
Step 13: Report D1 if $\Delta t1()$ and $\Delta t2()$ are similar, else report D2.

Through these algorithms one can trace power consumptions and performances of UE. Power consumptions of RRC states vary from network to network. Usually DCH consumes high power and transfer high bandwidth. FACH consumes low power low bandwidth (Table 3). IDLE state consumes almost neutral.

Table 3. RRC States and its power consumption

States	Power Consumption
DCH	800 mW
FACH	460 mW
IDLE	0

The experimental result shows the state transition of DCH→FACH→IDLE consumes more power and network overhead than DCH→FACH→PCH→IDLE. The RRC connection establishment and release exchange ten of signals between UE and UTRAN, every signaling message consumes certain energy and time delay even every RRC connection establishment take 2 sec delay [7]. This affects user experience and increase network overheads. The paper recommends the various modes for different application data to ensure high UE battery life (Table 4).

Table 4. Data level wise state change

Application	Data Level	State
Email	> 2 KB	DCH
Keep Alive	< 1 KB	FACH
Instant Messaging (IM)	< 1 KB	FACH
Web browsing	> 10 KB	DCH

Even the energy consumption not only depends RRS states and Apps states, its rely on screen states also. Normally Smartphone's screen is having two statuses called Screen On and Screen Off, in 3Gand 4G network and the IP data are sent both Screen Off and Screen On states. In this research experimental data were collected from 20 smartphone users over 5 moth duration, through this data (118 GB) the

UMICH data set [8] was built and specify measurement entities like BT (burst threshold) and IBT (inter-burst time). Foremost the mobile apps are categorized into gathered and scattered. If the application sends small packets and frequent short time keep-alive messages, it group as scattered ex. Facebook, skype, otherwise gathered ex. Gmail, google music. Totally 131.49 million packets are transmitted (both uplink/downlink) in that 55.13% packets are send in Screen On stats and 35.84% send in Screen Off State. In Screen Off transmission the packet payload is very small and it's transmitted more often (Table 5).

Screen On state commits less burst than Screen Off state because in Screen Off state, small keep-alive messages are sent more often by the applications. It outlets burst are smaller in size and duration. This behavior commit longer channel occupation time in high energy RRC state and therefore incur high energy consumption. Especially the scattered group of applications commits small burst in more often (Table 6). The result (Table 6) shows that some of the applications transfers more payload during their Screen Off session for ex Genie Widget, likewise 85.45% of all yahoo sportacular payloads are transferred in their Screen Off state. Facebook transfer totally 2,00,000 keep alive message send in Screen Off state and these packets occupy avg of 0.86 sec burst time and avg uplink and downlink payload is 318.83 B and 1.98 KB respectively.

Table 5. *Application wise data transmission*

Traffic Type	**Payload (GB) /%[a]**	**% of downlink payload**	**% of packets (× 10^6)/%[b]**	**% of downlink packets**	**Avg downlink packet payload size (B)**	**Avg uplink packet payload size (B)**
Screen-On	51.47/64.31%	96.31%	72.50/55.13%	60.71%	1126	67
Screen-Off	21.82/27.26%	93.52%	47.14/35.84%	52.60 %	823	63
Process Name	Off payload (GB) /%[c]	% of downlink off payload[d]	% of off packets (×10^6) /%[e]	% of downlink off packets[f]	Avg downlink off packet payload size (B)	Avg uplink off packet payload size (B)
Genie Widget	1.76/72.21 %	97.01%	3.80/73.16%	49.97%	901	28
Google Music	3.13/57.14 %	99.91%	3.30/57.02%	68.60%	1384	3
Epicurious Recipe	1.65/70.05 %	99.22%	2.69/69.29%	50.46%	1212	10
mediaserver	2.39/10.09 %	99.77%	2.66/11.05%	66.95%	1342	6
android.process.media	2.35/28.42 %	99.98%	2.37/29.06%	71.55%	1388	1
skypekit[g]	0.04/25.54 %	48.44%	2.07/46.73%	48.32%	22	22
Facebook	0.46/32.96 %	86.13%	1.95/40.67%	42.55%	487	58
yahoo Sportacular	0.23/80.45 %	83.53%	1.94/81.05%	41.98%	238	34
Gmail	0.39/46.00 %	63.65%	1.33/54.46%	47.70%	400	208

[a] Payload refers to the total screen-on/off payload, and % is relative to the total payload of all traffic.
[b] % relative to the total number of packets of all traffic.
[c] Off payload refers to the screen-off payload of the specific application, and % is relative to the total payload of this application.
[d] % of downlink screen-off payload of the specific application relative to the total screen-off payload of that application.
[f] % of downlink screen-off packet count of the specific application relative to the total screen-off packet count of that application.

Table 6. Application wise burst time

Traffic type	# of burst	Avg[a] # of uplink packets	Avg[a] # of downlink packets	Avg[a] uplink payload(B)	Avg[a] downlink payload(KB)	Avg[a] burst length(sec)	Avg[a] IBT following (sec)
Screen-On	650941	43.75	67.62	2910.44	76.17	2.92	335.13
Screen-Off	1910939	11.69	12.98	739.78	10.68	1.37	113.60
Process Name	# of burst	Avg[a] # of uplink packets	Avg[a] # of downlink packets	Avg[a] uplink payload (B)	Avg[a] downlink payload(KB)	Avg[a] burst length(sec)	Avg[a] IBT following (sec)
Genie Widget	5952	319.73	319.36	8852.48	287.88	17.87	3892.87
Google Music	5297	195.69	427.56	505.54	591.92	4.53	5111.50
Epicurious Recipe	63236	21.07	21.46	202.22	26.01	0.67	159.34
mediaserver	8163	106.44	215.53	669.82	289.35	5.01	14451.70
android.process.media	1442	461.88	1156.93	246.99	1605.84	19.83	123565.00
skypekit	42744	25.08	23.46	555.38	0.52	1.93	832.79
Facebook	203535	5.49	4.07	318.83	1.98	0.86	547.23
yahoo Sportacular	133785	8.39	6.07	285.44	1.45	1.52	261.78
Gmail	105478	6.60	6.02	1375.30	2.41	1.17	2002.60

Table 7. Comparison of fast dormancy and batching performance

Process Name	Optimization	Setting	ΔE[a]	ΔS[a] (%)	ΔD[a]
All Applications	Fast Dormancy	$T_{i,On}$ [b] = 8s, $T_{i,Off}$ [b]=8s	-16.39%	16.95	13.14%
		$T_{i,On}$ = 4s, $T_{i,Off}$ = 8s	-20.60%	28.29	21.26%
		$T_{i,On}$ = 8s, $T_{i,Off}$ = 4s	-34.44%	47.04	35.21%
		$T_{i,On}$ = 4s, $T_{i,Off}$ = 4s	-38.66%	58.38	43.31%
	Batching	only for screen-off, α=50s,β=10s	-22.33%	-6.24	-11.27%
		only for screen-off, α=50s,β=5s	-27.15%	-6.24	-10.67%
		only for screen-off, α=100s,β=10s	-36.72%	-30.00	-33.43%
		only for screen-off, α=100s,β=5s	-40.79%	-30.00	-34.25%
	Fast Dormancy + Batching	$T_{i,On}$ =8s, $T_{i,Off}$ = 4s batching only for screen-off traffic, α=100s,β=5s	-60.92%	-25.33	-30.59%
Facebook[c]	Fast Dormancy + Batching	$T_{i,On}$ =8s, $T_{i,Off}$ = 4s batching only for screen-off traffic, α=100s,β=5s	-60.19%	-36.27	-34.93%
Google Music[c]	Fast Dormancy + Batching	$T_{i,On}$ =8s, $T_{i,Off}$ = 4s batching only for screen-off traffic, α=100s,β=5s	-57.30%	7.12	-21.11%

[a] Each 'avg' in this table stands for the average value per burst

[a] *ΔE, S,D* are relative to the *E,S,D* of all traffic for the specific application

[b] $T_{i,On}$ is the inactivity threshold of fast dormancy for screen-on traffic and $T_{i,Off}$ is for screen-off traffic.

[c] For these two application rows, we consider the traffic of only one specific application, excluding that from other applications

The screen off study can be optimized by two traffic optimization technique called Fast Dormancy and Batching. The objective of these two methods is to give better solution to tradeoff of UE network Energy (E), Signaling Overhead (S) and Channel Scheduling Delay (D). In fast Dormancy, set different tail time (T_i) [10] for Screen On ($T_{i,On}$) and Screen Off ($T_{i,Off}$) sessions. Set conservative setting for Screen On session, that means long $T_{i,On}$ and set aggressive setting for Screen Off state that means short $T_{i,Off.,}$ then the combinational values are denote by $<T_{i,On},T_{i,Off}>$. The

research suggest <8, 4> for better tradeoff among ΔE, ΔS and ΔD.

Batching is another optimization technique which is applies only Screen Off state. These techniques send packets for particular time interval. It uses source window size (α) in sec and destination window size (β) in sec. The combinations of α = 100 sec and β =5sec is a better tradeoff, which saves upto 40.79% Energy, 30% of Signaling overhead, 34.25% in D (Table 7).

IV. EXPERIMENTAL SETUP

During this experiments, we used Redmi 3S with Android version 6 and MicroMax A089 with a rooted Android version 4.2.2, kernel version 3.4.5. The test used 2 different mobile applications to collect data. The first application is Application Resource Optimizer which used to collects real-time network activity of each application including detailed number of bytes sent by applications and the appropriate timestamp [11]. The second application is Trepn profiler which records Mobile data states, CPU state and power utilization in mW.

V. CONCLUSION

This paper investigates and reviews various factors that directly influence the energy consumption of Smartphone and analyze all the perspective of energy consumed parameters by real time measurements. In the previous researches all authors suggested that the network operators must focus fine tune inactivity timer's i.e. short interval consumes less energy than long intervals. Fast dormancy intervals also impacts energy drain, minimum intervals of dormancy save 80% of network energy.

This research confirms the keep-alive messages consume more energy anonymously and the long interval of keep-alive message consumes less energy than short interval messages. In future the author will propose a novel architecture to reduce energy consumption of 3G/4G data transmission.

REFERENCE

[1] Perala, P., Barbuzzi, A., Boggia, G., & Pentikousis, K.. "Theory and Practice of RRC State Transitions in UMTS Networks." 2009 IEEE Globecom Workshops, pp.1-6, 2009.

[2] Perrucci, G.P., Fitzek, F.H., & Widmer, J. *"Survey on Energy Consumption Entities on the Smartphone Platform"*. 2011 IEEE 73rd Vehicular Technology Conference (VTC Spring), pp.1-6. 2011.

[3] Li, D., Hao, S., Gui, J., & Halfond, W.G. *"An Empirical Study of the Energy Consumption of Android Applications"*. 2014 IEEE International Conference on Software Maintenance and Evolution, 121-130, 2014.

[4] Qian, F., Wang, Z., Gao, Y., Huang, J., Gerber, A., Mao, Z.M., Sen, S., & Spatscheck, O. "Periodic transfers in mobile applications: network-wide origin, impact, and optimization". *WWW*, 2012.

[5] Kononen, V., & Paakkonen, P. Optimizing power consumption of always-on applications based on timer alignment. 2011 Third International Conference on Communication Systems and Networks (COMSNETS 2011), pp. 1-8, 2011.

[6] Haverinen, H., Siren, J., & Eronen, P. Energy Consumption of Always-On Applications in WCDMA Networks. 2007 IEEE 65th Vehicular Technology Conference - VTC2007-Spring, pp. 964-968, 2007.

[7] GSMA, *"Fast Dormancy Best Practices"*, GSMA Official Document TS.18, 2011.

[8] Huang J, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. *"A Close Examination of Performance and Power Characteristics of 4G LTE Networks"*. In MobiSys'12. pp. 225-238, 2012.

[9] Qualcomm Engineering Services Group, *"System Parameter Recommendations to Optimize PS Data User Experience and UE Battery Life"*, Technical Document, 2007.

[10] Wang, Z, Qian, F., Gerber, A., Mao, Z.M., Sen, S., & *Spatscheck*, O. *"TOP: Tail Optimization Protocol for Cellular Radio Resource Allocation"*. The 18th IEEE International Conference on Network Protocols, pp. 285-294, 2010.

[11] Pandikumar, S, and Sumathi, M. *"Analysis of Energy Profilers in Smartphone Environment"*, International Journal of Advanced Research in Science and Engineering, Vol.06 Issue 02, pp. 20-29, 2017.

[12] Qian, F., Wang, Z., Gerber, A., Mao, Z.M., Sen, S., & Spatscheck, O. *"Characterizing radio resource allocation for 3G networks"*. Internet Measurement Conference. pp. 137-150, 2010.

