Thesis Project

Software Performance Analysis for ARM Architectures

Author: Shahriar Derhami
Supervisor: Sabri Pllana
Examiner: Mathias Hedenborg
Semester: VT 2015
Subject: Computer Science

Abstract

This bachelor thesis discusses existing performance analysis techniques for ARM based architecture processors. This includes a comparison between couple of performance analysis applications installed on two Android test devices. Each application monitored CPU performance of the device in three test scenarios. Each test was done in five iterations. The results were compared for each test and for each application. The results of these iterations were compared to find the most stable application among the rest.

Keywords: ARM, Android, CPU usage, ARM Cortex-A9, Exynos 7420, Smartphone,

List of Acronyms and Abbreviations

ABC - Acorn Business Computer	
BBC - British Broadcasting Corporation	
CMA - CPU Monitor Advanced Lite	
CMM - CPU Memory Monitor	
CMOS - Complementary Metal Oxide Silicon	
COC – Clash of clans	
CPM - CPU Monitor	
CPS - CPU Stats	
FPGA - Field Programmable Gate Array	
IPV - Intel® Performance Viewer	
MKS - MK system monitor	
MOS - Metal Oxide Semiconductor	
OS – Operating System	
OSM - OS Monitor	
PMP - PassMark PerformanceTest	
RISC - Reduced instruction set computing	
RTOS - Real-Time Operating System	
SD – Standard Deviation	
SM - System Monitor	
SSM - Simple system monitor	
UTF - Usage timelines free	

List of figures

List of tables

Contents

1 Introduction

Nowadays, mobile devices such as smart phones and tablets are becoming more popular, hence the number of their users is increasing every day [1]. On the other hand, embedded systems are everywhere, from cars and TVs to DVD players and refrigerators. So, the significant role of embedded systems in human life cannot be ignored and denied. [2]

1.1 Growth of mobile devices

It seems to be no compelling reason to argue that mobile devices have become more popular than laptops. The heart of mobile devices and embedded systems is their CPU. It is their CPU that affects their performance and battery usage. This will bring us to ARM processors which are a part of RISC (reduced instruction set computer) architecture.

Some of the key features of ARM processors are: being single cycle execution, having a power saving design, high performance due to their 64-bit and 32-bit execution. At the moment, ARM based processors are being used in most of mobile devices and embedded systems. Therefore, software performance is highly relevant to ARM based processors. [3]

There have been numerous studies in the field of software architecture while there have not been much researches on RISC family software architectures. [4]

It necessitates them to make progress in CPU power and disk storage repeatedly in their next models and generations. These changes during last decades are necessary due to giving users a lot of new services and abilities with different kind of applications. Examples of such applications are different kind of social media for sharing images, videos, text, location and etc. many other sharing applications besides most graphical advanced games. They are the cause of growth in CPU usages, and make the manufactures to use more powerful CPU on their new devices, but the technology used in cell phones batteries has less progress compared to CPUs. One of the existing limitations of mobile devices is their battery usage which is mainly caused by CPU performance. Also, most of embedded systems have limited performance due to their CPU architecture. On the contrary, ARM architecture processors seem to have the key feature to consume less power. [5][6]

Hence, the aim of this project is to run some applications on ARM family processors and monitor the CPU performance of them.

1.2 Related Work

While there have been many studies of performance analysis methods and tools for x86 architectures, not much work has been done so far with respect to ARM architectures.

Examples of performance analysis methods and tools for x86 architectures include Pllana et al. [7] and McCraw et al. [8] which have developed two articles, relatively similar to what is intended to be conducted by the author of this report.

In the study of Pllana et al. [7] they described a wide range of performance modeling approaches from high level mathematical modeling to detailed instruction-level simulation. In each approach, an explanation about how program and machine are modelled, evaluation effort, efficiency and accuracy are provided. Then an overall evaluation of the described approaches is presented. [7]

In another study, McCraw et al. [8] investigated a progressive execution performance deterioration of the well-known ASCI Sweep3D compact application. They used effective data collation, management and graphical presentation, which made Scalasca for the first time, able to demonstrate performance measurements and analyses with 294,912 processes. [8]

1.3 Problem definition

The ever growing number of computer devices, such as embedded systems and smartphones is obvious. Fast and clear response to most of human needs everywhere seems necessary. From the car you drive to temperature thermostat of bedroom, all can make life easier. With fast and more powerful computer systems, these kinds of services come in a lot of varieties and options. Thus, they need to be provided with a good hosting system in order to achieve the expected results in a shorter time with fewer errors.

ARM based processors are the most common ones used in mobile devices and embedded systems. Therefore, software performance is highly relevant to ARM based processors.

1.4 Purpose and research question

The objective of this report is to investigate tools for performance analysis of software that runs on ARM-based processors. The aim is to find the answer to questions as below:

RQ1: Which features are these applications going to support and how accurately are these existing tools supporting performance analysis in comparison to one another?

RQ2: For the future generations of these tools what improvements will lead to achieving more accurate results?

1.5 Limitation

This thesis defines the analysis of some application on devices which contains ARM processors. There are only a few analysis performance tools such as www.dyninst.org available to be used during this research. Neither exist a lot of equipment for testing applications on them. Therefore, just a few performance analysis applications and benchmarks are used during this report and only one mobile device is the host of the applications for testing.

1.6 Target group

As mentioned previously the performance analysis would be tested on two mobile devices with an Android OS (Operating System).

This would be useful for Android developers to develop application for such mobile devices. They should be able to determine the performance requirements of their applications and find the best result for their requirements by comparing different options.

1.7 Outline

This report divides in eight chapters. The following chapter presents the background and gives more information of the problem. In chapter three, the approach through this project is explained. Chapter four describes the applications which are used in this project. Then in chapter five test scenarios and their results are presented. In chapter six the results of the test scenarios are compared and analyzed. Chapter seven and eight contain respectively, the discussion and the conclusion.

2 Background

In this section, ARM architecture, embedded systems and the aim of software performance analysis will be described in order to help the reader to better understand the topic and the area of ARM architecture.

Then we will have a short description about Androids, the most common operating systems between smartphones, as our test devices are within this OS and Play Store as the official Google application store for Android devices. [13]

Then a description about the CPU of our two test devices will be presented in the following sections 2.6 and 2.7.

2.1 ARM architecture

The history of the ARM architecture began in 1980s. After success of the Acorn Computers ltd on development of BBC (British Broadcasting Corporation) micro platform, they decided to develop something more powerful than simple Metal Oxide Silicon CMOS processors. [5][9]

Their first ARM-based products was the BBC micro series of computers and due to success of this generation of their products, they decided to target business market. [5][9]

That was Acorn Business Computer's (ABC) plan. ABC planed BBC micro platform to work with a second processor, processors like Motorola 68000 and National Semiconductor 32016. The problem was that neither of these processors was suitable for this case. The Metal Oxide Semiconductor (MOS) Technology 6502, which they used, was not matched with their plan. When they did not succeed, Acorn decided to have a new architecture, inspired by white papers on Berkeley RISC project. Acron RISC machine project started in October 1983 with the key goal of reaching a low-latency input/output handling. On April 26th 1985, the first samples of ARM silicon were tested and the results showed that they performed as expected. In 1987 by release of Acron Archimendes the original aim of ARM-based computer was achieved. [5][9]

2.2 Embedded Systems

Embedded systems are the computers which you cannot see or at least are not observable like normal computers. They are inside the machines, as a small processor which is only programmed to do the task that they need to.

Like a processor in a washing machine, or the one in a toaster or a microwave. Their performance, size and other specifications are enough for the task they assigned to. Hence, both in economic and also power consumption they are effective. The embedded systems have dedication with a large electrical or mechanical system.

Embedded systems are commonly come in two shapes; one is in based on a microcontroller and another one a FPGA (Field Programmable Gate Array). Microcontrollers have better calculation power and whereas FPGAs offer more I/O-pins and they are faster than microprocessors. But with RTOS (Real-Time Operating System) and multicore technologies for microcontrollers and also soft CPUs for FPGAs, it seems that the two shapes of embedded system do not have much difference in parameter performances. [10][11]

2.3 Software Performance Analysis

As *Raj Jain* said in his book [12]: "Contrary to common belief, performance evaluation is an art. Like a work of art, successful evaluation cannot be produced mechanically".

It needs an affectionate knowledge about the system which is going to be modelled, to help to find the most efficient solution. [12]

When it is about to compare two devices with each other, there would be three ways for this comparison. First, assuming that the two devices are similarly good the average of their performance could be taken. Second, base is on that the device A is better than B, therefore the ratio of device B performance, considered as the base. The other way is opposite of the previous way, the performance ratio of device A is the base and the conclusion is that device B is better.

This technique is known as "Ratio Game". "The technique of using ratios with incomparable bases and combining them to one's advantage is called a ratio game" said Raj Jain. [12]

2.4 Android

Android is an Operating System based on Linux kernel, and it is developed by Google. It is used for mobile devices such as smartphones, tablets, TVs, cars and also watches. Asides from that it has been used for game consoles and regular PCs.

Android designed to have a user interface based on direct manipulation which is suitable for touch screen mobile devices. Therefore the OS uses touch with all kind of actions (Swap, Tap, Pinch and etc.) as an input and also getting help by providing a visual keyboard. Now, in 2015, Android is the most installed operating system. [13]

2.5 Play Store

Is the official application store for Android devices, owned by Google. Play store contain application match with mobile phones, tablets and Android TVs.

October 22, 2008, it has been released as “Android Market” until March 6, 2012, Google play store was created with combination of Google music, Android market and Google eBookstore.

Google play store makes it possible for developers from 61 countries all over the world to deliver their paid application on it as on November 2014. [14][15]

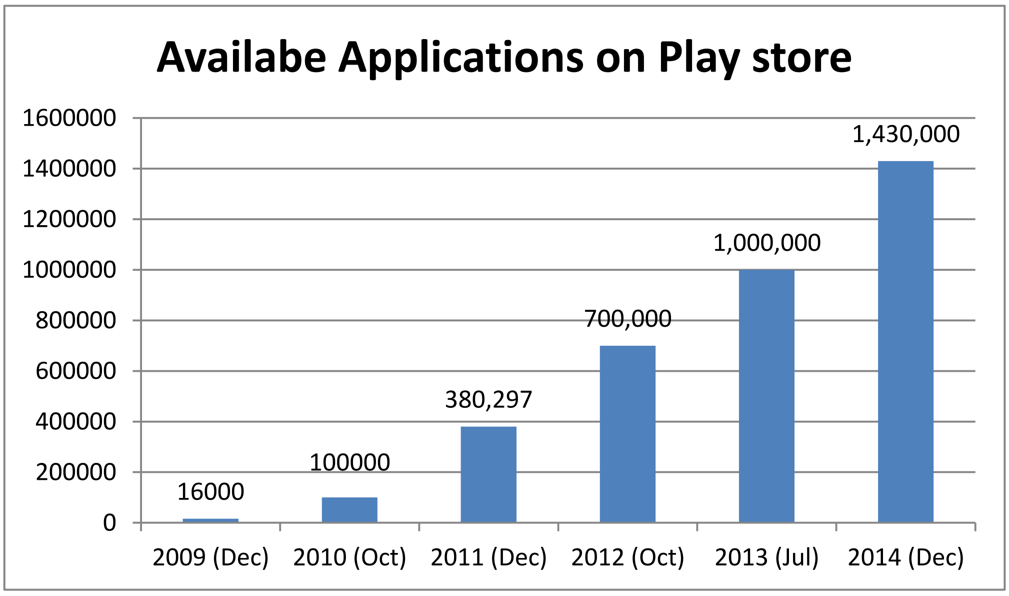

Figure 2.1 – Available applications on Play store for download

As it shown on figure 2.1, number of available applications on Play Store increased rapidly. [14]

2.6 Test Devices

Two (personal) devices, based on their availability in the time of this research, are used. Twelve applications are chosen and installed on both of them. These applications are introduced in section 4. In sections 2.6.1 and 2.6.2 a short description of these two devices’ specifications/information are presented.

2.6.1 ARM Cortex-A9

A Samsung galaxy S2, with ARM Cortex-A9 CPU and Android OS version 4.2.2, used for this report. The device CPU licensed by ARM Holding under ARMv7-A architecture, a 32-bit multicore processor providing up to 4 cache-coherent.

Cortex-A9 compare to Cortex-A8 has an overall increase performance of above 50 percent due to single processor solution.

In this generation, increase GPU to Mali-T624, system IP to CoreLink NIC-400/301 and memory controller to CoreLink DMC-342, makes it a fast system design which makes it a popular among smart phones, digital TVs and etc. (During its release date – 2008). The specifications of the ARM Cortex A-9 are shown on table 2.1. [16][17][39][40]

2.6.2 Exynos 7420

A Samsung galaxy S6 with an "Exynos 7420" CPU has been used as another test device for this report. The device Android OS is in version 5.1.1.

This CPU is an ARM base system on chips series which constructed by Samsung Electronics. Exynos 7420, with 8 cores and a clock speed of maximum 2.1 GHz, has a great and extremely efficient performance.

It has been released on 2015 with Samsung S6 series. It has a Cortex-A57 as its primary CPU and a Cortex-A53 as its companion CPU licensed by ARM. The specifications of the Exynos 7420 are shown on table 2.1. [18][19]

Table 2.1 – "Exynos 7420" and "ARM Cortex A-9" specifications

	Exynos Octa 7420	**ARM Cortex-A9**
CPU cores	8	2
Primary CPU	Cortex-A57	2x Cortex-A9 Harvard Superscalar processor
CPU Architecture	ARMv8-A	ARMv7-A
Word Length	64 bit	32 bit
Data Bus	32 bit	64/32-bit Multi-layer
CPU Clock Speed	2.1 GHz	1 GHz

3 Method

The chapter contains a description of the scientific approach and the list of chosen applications used in this thesis project.

3.1 Scientific approach

This thesis report is based on most practical works such as monitoring CPU performance during running applications, ten applications had been chosen for monitoring CPU performance from Play Store while they are running on two Android devices in three main scenarios. Six additional test scenarios have been performed on all of the devices. Therefore, an experimental approach had been chosen for this purpose.

In section 4 we will take a look at their description and functionalities. Each of them will monitor our two test devices CPU usage during running three applications. In section 5 we will have a short description plus the average test result for these three applications. Average result is based on five times results provided by monitoring applications. The results captured during running applications at least two times longer than application interval update chosen during monitoring.

3.2 Human centered approach

In this section, first we started by doing a search in app store and play store in order to find appropriate applications for monitoring CPU performance. Below is the list of applications which were suitable for our purpose:
Play store:

- Intel® Performance Viewer
- CPU Memory Monitor
- OS Monitor
- CPU Monitor Advanced Lite
- CPU Monitor
- Simple system monitor
- System Monitor
- Usage timelines free
- MK system monitor
- CPU Stats
- PassMark PerformanceTest
- Pi

These applications have been chosen based on the good reviews from the users who have installed them on their devices and the fact that are free of charge. There were other applications available on Play Store in the same category, but they either did not have good reviews from users or they were not free of charge.

3.3 Ethical considerations

This report deals with three primary test scenarios monitored by ten applications on two test devices.

Applications: All applications in both sides, test scenarios and monitoring applications, are free of charge for the user and there is no restriction regarding using them in this report.

Test Devices: Both two test devices are personal devices.

Accounts: The accounts used on these devices are also personal accounts.

This thesis is presenting how these applications monitor performance analysis on ARM based CPUs compared to each other. The experiments are focused on performance and do not involve any personal data. This comparison will help developers in their way of developing the applications based on these kinds of CPUs.

4 Application Description

In this part we are going to present a short description of the applications that we found suitable for our purpose base on their functionalities and use them during this report. With suitable, it means that they can present a live view of our needs and give this opportunity to take screen shots and record the results. Description of their key features and functionalities will follow in the next sub sections. More test scenarios and their results would be presented in the sections 5 and 6.

4.1 Intel® Performance Viewer

This performance viewer is an application for Android devices and is developed by Intel Corporation. This tool is used for monitoring performances of the device, by presenting the information collected from foreground applications into a real time graph.

The application would monitor CPU, 3G/Wi-Fi, data storage, battery status and memory usage regarding to analyze them as explained bellow:

- Monitor CPU performance of foreground application by ability to have the collection rate update of showing graph from 100 to 2000ms. It is also possible to monitor system's core frequency separately in MHZ.
- Observe the amount of data transferred on KB per second via 3G/Wi-Fi.
- It has six different sections for watching battery status. Battery power (mW), energy used (mWh), voltage (V), current (mA), Charge (percent) and actual capacity (mAh).
- Presenting device and SD card free space (MB).
- IPV (Intel® Performance Viewer) presents the CPU usage of applications in average amount during monitoring them.

[20]

4.2 CPU Memory Monitor

This is another application for monitoring applications process in real time, developed by "Pavel Ugo".

CPU memory monitor (CMM) provides its monitoring results in a list which could be sorted by CPU load, total CPU consumptions or used memory.

CMM provides the information such as a real time auto sort list of running and dead processes, a total CPU usage and a system CPU usage. The applications which are shown on its list have information such as PID, running time and a real time CPU usage by percent.

Interval refresh time for presenting the results in CMM is in threshold of 1 to 10 seconds. [21]

4.3 OS Monitor

OS Monitor (OSM) provides services in four sections, *Processes* (monitor all processes), *Connections* (Present all network connections), *Misc* (monitor battery, processors, network interfaces and file system) and *Messages* (searching dmesg or logcat in real-time).

The application shows a real time CPU usage for the first three applications in the usage list on the notification bar and it also shows an overall real time CPU usage. [22]

4.4 CPU Monitor Advanced Lite

Another application for Android devices, develop by ECNApps. “CPU Monitor” is analyzing CPU performance by record historical information about applications process. Beside this, it can monitor memory and battery usage on single or all running applications on the device. It also allows you to graph the results of each process.

Since it is a free version of the app, analyze duration time would be 4 minutes, the paid version could handle analyzing up to 720 minutes. It also can keep recording up to 10 processes in its free version.

CPU Monitor Advanced Lite (CMA) has the functions for start automatically for the processes defined on its option, based on CPU sustained, CPU sustained over time and also battery temperature.

The idle CPU sampling interval is from 1 up to 50 seconds and also the same range for recording CPU sampling interval. [23][24]

4.5 CPU Monitor

CPU Monitor (CPM) is an android application produced by Cygnus Software. The latest version of this application is 1.17.1 (18) published on 21 of July, 2015. The free version of application let users to have CPU usage status on notification bar, the first two applications in the list of active applications, plus the total CPU usage. In the Pro version it is possible to choose an application to monitor specifically.

When the list of applications is presented, users can kill the chosen ones manually or through the application manager. The list is auto sorted due to CPU usage of applications. [25]

4.6 Simple system monitor

Simple system monitoring (SSM) is a monitoring application for Android devices, developed by Darshan Parajuli. It graphically shows different system components. CPU usage, CPU frequency, Network Status and usage, RAM usage, Battery status, checking storage speed, they are all the services which provided by SSM to users.

SSM also shows a list of available processes to user with ability to kill the unnecessary ones by user. [26]

4.7 System monitor

System monitor (SM) provided by “Pavel Pedrov”, is an application for monitoring system performance. SM presents report about CPU usage, memory usage and battery status on the notification bar. A list of active applications and processes will be presented to the user by opening SM. The user will be able to stop the chosen application/process via SM. [27]

4.8 Usage timelines free

Usage timelines free (UTF) version is an application provided by “Dr. Achim Leubner” available on play store for Android devices. It is a CPU and process monitor application. UTF presents the results in a graphical load history on the notification bar and with a list of running applications.

The application presents a total CPU usage in notification bar which describes the total usage of CPU at time for a live application monitoring process. But in the list view it is providing a list of system process and their CPU usage separately. [28]

4.9 MK system monitor

MK system monitor (MKS) is a monitoring application on Android device and represents the results of CPU, memory and network status on graphs. It also shows the real time CPU status on notification bar.

Other functions provided by MKS are a list of status of all the processor cores and total CPU usage. It also gives users the ability to kill/terminate the selected process. [29]

4.10 CPU Stats

Under the name of “takke”, CPU Stats (CPS) is an application which can shows SPU usage and frequency on notification bar, total usage and also usage per core. [30]

4.11 PassMark PerformanceTest

PassMark PerformanceTest (PMP) is a benchmark application. It allows testing the device in different types of test scenarios to perform a device performance and speed status. [37]

This application has been used for additional test scenarios which are described in section 5.5.

4.12 Pi

It is an application for testing the speed of the device while it calculates the Pi number. It uses “Optimized Parallel Chudnosky Algorithm” for calculation. [38]

This application has been used for additional test scenarios which are presented in section 5.5.

4.13 Application functionalities

Functionality of the first ten applications regarding their area of monitoring devices is presented in table 4.1.

Since “PMP” and “Pi” are used for additional tests and their functionalities differ from the rest of the applications, they are not presented in table 4.1.

Table 4.1 – Applications functionalities

	Update interval time range	CPU usage	CPU frequency	List of process	CPU graph
Intel Performance viewer	100 – 2000ms	✓	✓	✓	✓
CPU memory monitor	1-10s	✓	✗	✓	✗
OS monitor	Fast-Normal-Slow mode	✓	✓	✓	✗
CPU monitor advanced lite	1-60s	✗	✓	✓	✗
CPU monitor	1-10s	✓	✗	✓	✗
Simple system monitor	1-60s	✓	✓	✓	✓
System monitor	1-300s	✓	✗	✓	✗
Usage timelines free	0.5-10s	✓	✗	✓	✓
MK system monitor	1000-5000ms	✓	✗	✓	✓
CPU stats	0.5-10s	✓	✓	✗	✗

The (default) Update interval time of these applications is considered while running the test. More description about test conditions is presented in section 5.1.

A list of the active processes, supported with these applications is a useful feature to show what other processes are running at the same time, in order for the user to be able to close/terminate the unnecessary processes, while tests are being performed on device.

The application that has most of these functionalities and has a wide range for choosing “Update interval time” is a better tool for most of the scenarios.

5 CPU usage performance test

In this section, based on observations, following scenarios have been deployed in order to test ARM based CPUs using applications which mentioned in previous section. The results are presented in tables in each section.

5.1 Test Cases

The following samples are gathered from two test devices (sections 2.6.1. and 2.6.2). In each of the tables an average result of mentioned ARM CPU is presented.

These samples are generated by using "Samsung Galaxy S2, CPU ARM Cortex-A9, 1.2 GHz dual-core", "Samsung Galaxy S6, CPU Exynos 7420 Octa-core (2.1 GHz Quad + 1.5 GHz Quad)" [31][32]

The comparison is made using three different applications as YouTube, Clash of Clans and Skype. The results are based on monitoring the CPU usage of devices, while using mentioned applications. In each scenario, device was only monitored while it was running the mentioned application and other unnecessary processes in background were terminated.

In each process, the results of CPU performance monitored five iterations. The interval time of calculating the CPU performance of each application was decided to be twice of its default value. All the data gatherings started after at least twenty second after the application ran on the device. This decision was made to give the applications a certain amount of time to run properly.

The applications were chosen because they use the most of the main functionalities of the device as Wi-Fi/cellular data usage, video and audio input/output and etc. These functionalities are referred to in each section according to the related test scenarios.

5.2 YouTube

YouTube app for Android is developed by Google Inc. and exists in Media and Video category of Android applications. YouTube application need some permission from device in order to run and performs its services.

Due to measure the CPU usage during running an application, we chose YouTube app as it needs Wi-Fi/Cellular connection, Access to audio output and also use graphical process for displaying videos.

Table 5.1 is presenting the (average) results of each application while they were monitoring YouTube test case.

Table 5.1 – Applications average result for YouTube

Monitoring Application	Average CPU usage (percent) for "Exynos 7420"	Average CPU usage (percent) for "ARM Cortex-A9"
Intel mobile performance Viewer	7.91	11.6
CPU Memory Monitor	8.2	10
OS Monitor	5.44	11.26
CPU monitor advanced lite	10	9
CPU Monitor	8.4	10.8
Simple system monitor	7.628	7.048
System Monitor	22.3*	30.7*
Usage timelines free	15.8*	55.8*
MK system monitor	37.6*	40*
CPU Stats	21.6*	44.8*

Numbers with sign "*" referring to the whole CPU usage during the test while the others presenting the CPU usage just for the mentioned application.
.

The different results in this table could show the difference in performance between the CPUs characterization as mentioned in section 5.1 and also table 2.1. Regarding to "Exynos 7420" specifications (Table 2.1) the difference between the results in table 5.1 is obvious. But for CMA and SSM, it is "ARM Cortex-A9" which has a lower CPU usage in their outcomes (in average). Among all test scenarios, these are the two average outcomes which show a lower CPU usage for "ARM Cortex A-9". By looking through their stability results in the next sections, it can be verified that if their results are accurate or not.

These results as mentioned before are the average of five test iterations. In the next chapter, the results of the five test iterations are presented.

Five test iterations, Mean value and their Standard Deviations are presented in appendixes 1 and 2.

5.3 Skype

Skype is an application, available on Android and IOS, developed by Skype team. It is categorized in communication category.

The purpose of using Skype as another application in our observation was Skype needs noticeable permissions such as accessing camera(s), audio output and input and system keyboard.

The table 5.2 represents the results of CPU usage monitor during running Skype application on test devices.

Table 5.2 - Applications average result for Skype

Monitoring Application	Average CPU usage (percent) for "Exynos 7420"	Average CPU usage (percent) for "ARM Cortex-A9"
Intel mobile performance Viewer	9.19	21.75
CPU Memory Monitor	7.28	34.7
OS Monitor	13.64	32.28
CPU monitor advanced lite	5	20
CPU Monitor	15.8	25.8
Simple system monitor	7.056	32.736
System Monitor	33.56*	44.7
Usage timelines free	47.8*	50*
MK system monitor	45.6*	49*
CPU Stats	37.8*	61.4*

Numbers with sign "*" referring to the whole CPU usage during the test while the others presenting the CPU usage just for the mentioned application.

The different results in this table could show the difference in performance between the CPUs characterization as mentioned in section 5.1.

These results as mentioned before are the average of five test iterations. In the next chapter, the results of the five test iterations are presented.

Five test iterations, Mean value and their Standard Deviations are presented in appendixes 3 and 4.

5.4 Clash of Clan

A multiplayer game produced by Super Cell team. It is categorized under game and entertainment section in play and app store.

The game using connection (cellular - Wi-Fi), needs access to device storage, it uses system graphic and audio outputs, screen touch and visual keyboard.

These functions used by app were the main reasons for mention Clash of Clans (COC) as one of our test scenarios in order to monitor CPU usage performance.

The results of average CPU usage during running COC application on the test devices are shown in table 5.3.

Table 5.3 - Applications average result for COC

Monitoring Application	Average CPU usage (percent) for “Exynos 7420”	Average CPU usage (percent) for “ARM Cortex-A9”
Intel mobile performance Viewer	6.82	17.38
CPU Memory Monitor	6.38	13.7
OS Monitor	9.22	28.72
CPU monitor advanced lite	8	33
CPU Monitor	7.8	29
Simple system monitor	6.44	30.78
System Monitor	21.32	26
Usage timelines free	21.8*	46.5*
MK system monitor	18.2*	44.4*
CPU Stats	19.4*	44*

Numbers with sign “*” referring to the whole CPU usage during the test while the others presenting the CPU usage just for the mentioned application.

These results as mentioned before are the average of five test iterations. In the next chapter, the results of the five test iterations are presented.

Five test iterations, Mean value and their Standard Deviations are presented in appendixes 5 and 6.

5.5 CPU Mathematical operation speed

Monitoring CPU speed in mathematical operations would be another measurement method regarding for ranking systems due to their calculation speed. In sections 5.5.1, 5.5.2, 5.5.3 and 5.5.4, four additional test scenario and their results will explained. Likewise two other additional test scenarios would be performed regarding to monitor system CPU speed in data section.

In the following subsections a description about the test scenarios can be found. In each section, the formulation and algorithm are described to help readers to understand the calculation. Each test scenario ran for five iterations on the device. The outcome which is presented in upcoming tables is the average amount of these five iterations. The Aim of the additional test scenarios is to find out more about the CPU performance of the two test devices.

5.5.1 Pi (π) Calculation

A mathematical constant which is represent the ratio of a circle's circumference to its diameter. Pi does not settle into an infinitely repeating pattern of digits since it is irrational.

As long as it is an infinite number, calculation of this number would be a suitable choice for another scenario in order to observe the CPU performance of the device. [33]

The device is supposed to calculate the Pi number with a define number of digits which could be a thousand or five millions of digits. This threshold can be chosen in the "Pi" application. The consumed amount of time for calculation is considered as a majoring instrument to monitor the CPU performance.

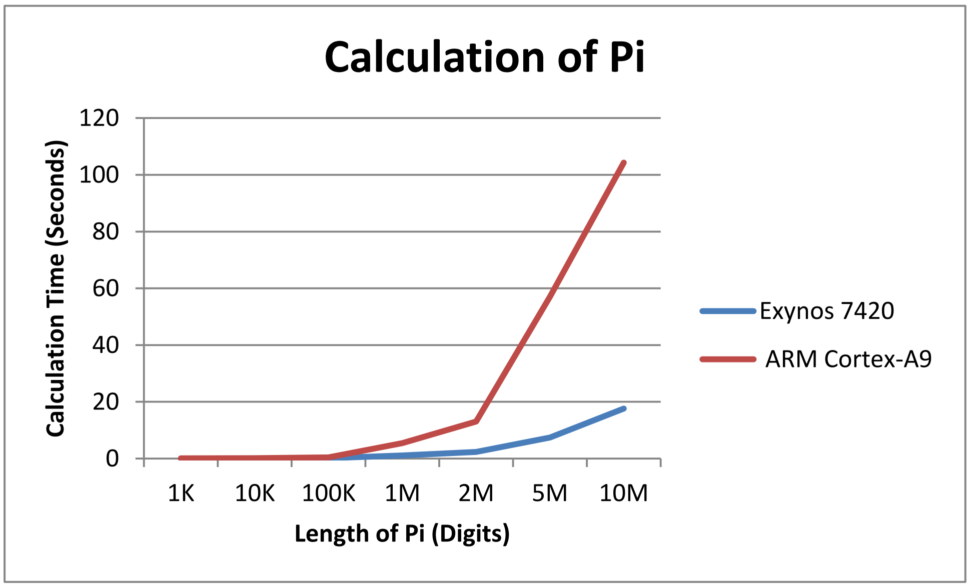

Figure 5.1 – Time spent for calculation Pi in different lengths.

The amount of spent time for both devices is near to each other for calculation of Pi until the length of it is under 2 million digits, but the difference would be revealed when the length reached to 5 and 10 million digits.

ARM Cortex-A9(Android) spent time increased approximately 4.361 times higher when the length of pi digits increased 2.5 times, from 2 to 5 million (13.075sec for 2M – 57.021sec for 5M).

But in the same way, *Exynos 7420* spent 7.395 Second which was 3.162 times higher comparing to the previous calculation. (2.338sec for 2M – 7.395sec for 5M).

5.5.2 Integer math, Floating point math and find prime numbers test

"Integer math test" targeted how fast the CPU is while performing mathematical integer operations where the integer is whole number without any fractional part.

These calculations are the basic operations in computers and shown a perfect point of "raw" CPU throughput. Integers in a large set of random 32 to 64 bit used in this test while add, divide, multiply and subtract are performed on them. [34]

"Floating point math test" is like the integer math test but with the integers with fractional part. This test would uses 240kb per core totally in memory buffers. [34][35]

"Find prime numbers test" measures how fast could be a CPU during search for prime numbers. The *Sieve of Atkin* formula uses in this test and targeted the limit of 32 million of it. 64-bit integers using for all operations and about 4MB per core would be uses during this test. [34][35]

Table 5.4 is showing the results of mentioned three tests on "Exynos 7420" and "ARM Cortex-A9". Results are based on average outcome of five times running test on mentioned device.

Table 5.4 - Integer math, Floating point math and find prime numbers test results on "Exynos 7420" and "ARM Cortex-A9"

Processor	Integer Math Result (Mops/Sec)	Floating point result (Mops/Sec)	Find prime numbers (Thousand Prime/ Sec)
Exynos 7420	6014.6	2711.4	5263.6
ARM Cortex-A9	91.8	199	39.34

As presented in table 5.4, "Exynos 7420" had a faster computational speed compared to "ARM Cortex-A9" in all three sections based on its CPU speed (CPU click speed) and its advantage in the number of cores it possesses (Table 2.1). The higher difference is observed while these devices run the "Find prime numbers" scenario.

5.5.3 Data Encryption and Data compression test

"Data encryption test" using different encryption methods to encrypt random blocks of data with an encryption key in order to someone can access the data by using it. It further checks the system ability to create a hash data, as a common cryptographic method.

PMP uses "TwoFish", "AES", "SHA256" and "Salsa20" as the techniques for this test scenario. About 1MB per core would be used as memory buffer. [34]

Table 5.5 – Data Encryption test results for "Exynos 7420" and "ARM Cortex-A9"

Processor	Data Encryption result (Mbytes/Sec)
"Exynos 7420"	25
ARM Cortex-A9	2.85

"Data compression test" monitor how fast a CPU can compress blocks of data into the smaller ones without losing any original data.

PMP uses an encoding algorithm suggested by a method from *Ian H. Witten, Radford M. Neal, and John G. Cleary* ("Arithmetic Coding for Data Compression", Stanford University, 1987). 16kb memory buffering would be used per core during this test. [34]

Table 5.6 – Data compression test results for "Exynos 7420" and "ARM Cortex-A9"

Processor	Data compression result (Mbytes/Sec)
"Exynos 7420"	9.066
ARM Cortex-A9	0.611

As presented in table 5.6 and also the word length and data bus deifference between theese two CPUs, Exynos 7420 is 14.83 times faster than ARM Cortex A-9 in data comparission test.

5.5.4 Random String Sorting

Sorting strings is a common task between applications. PMP uses "Quick sort" algorithm in this process. "Quick sort" is a systematic method presented by *Tony Hoare* in 1959, which is placing the elements of any type in an array in order. [36]

The "random string sorting" test is using 25MB per core in total as memory buffering.

Table 5.7 – Random string sorting test results for “Exynos 7420” and “ARM Cortex-A9”

Processor	Random String Sorting result (Thousand Strings/Sec)
“Exynos 7420”	8887.6
ARM Cortex-A9	705.8

Table 5.7 represents the speed of random string sorting of these two CPUs. While “Exynos 7420” has the advantage in CPU speed and also for its number of cores, in this scenario, it behaves faster than “ARM Cortex A-9”.

6 Analysis

We gathered test results in some test scenarios with mentioned monitoring applications. Now we are going through to analyze their results, their stability during monitoring test, compare this stability with other ones and find out which on these applications would be a better choice in order to use as a suitable monitor CPU performance.

These applications present the CPU usage in real time, average time and also real time graphs. For the ones which show the results in real time, we decided to record the results in five times and calculate the average of these five times for our final result. These results would be shown in next sub sections in the related figures regarding to observe their stability in giving the results.

Then we would see and compare the whole results of these ten applications in each scenario.

6.1 Stability during performance monitoring

As it is explained in previous chapters, each test was repeated five times. The following figures are presenting the result of these five iterations for each test.

The related line about the results of each application in each scenario presents how the application shows the results and how they are close to each other. The shape of the lines performs this similarity in results and can help to find out how stable does the application behaves while it shows the performance of the CPU. In other hand, Standard deviation (SD) formula is used for these five iterations.

Standard deviation, in the area of statistics, is a way to measure the amount of variation of dispersion of a set of values. Since the standard deviation is close to zero, it represents that the selected data leans to the "mean" value of the data set. [41]

Further information in each upcoming section is presented in the tables and figures in appendix section of this report.

6.1.1 CPU memory monitor

Figure 6.1 represents the stability of this application during running our three target applications on device with "Exynos 7420" processor.

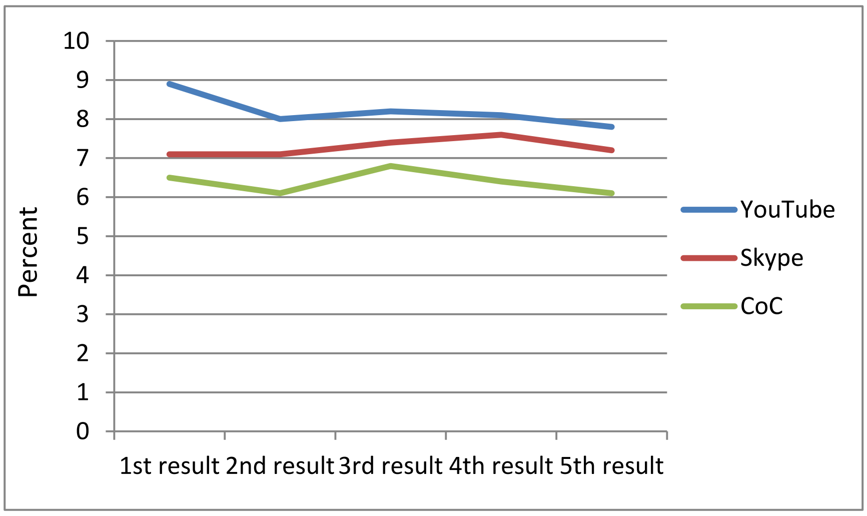

Figure 6.1 – CMM test results for "Exynos 7420"

If all the results of an application are in a same range, the application can be described as a stable application. This means that stable application will give the same result in the test iterations in a small tolerance. COC had less than 1 percent difference in its results threshold. Skype shows less than one percent changes on CPU usage results. YouTube had a little higher difference, which was not high enough to made CMM results as an unstable outcome. The SD of the results for CMM in these three cases is 0.4183 for YouTube, 0.2167 for Skype and 0.2949 for COC. Since the SD for Skype is closer to zero (the definition of SD) it is closer to mean (expected) value. (Appendix 7)

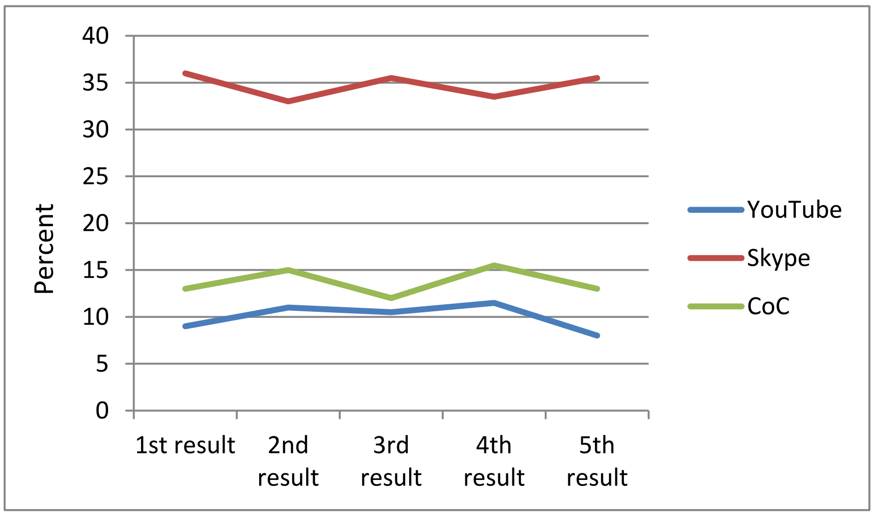

Figure 6.2 - CMM test results for "ARM Cortex-A9"

Figure 6.2 representing the stability of the application during running our three target applications on device with "ARM Cortex-A9" processor.

The SD values of these three scenarios outcomes are 1.4832 for COC, 1.3509 for Skype and 1.4577 for YouTube. Closeness to zero of SD for Skype can define it as a stable result. (Appendix 7)

6.1.2 OS Monitor

Due to previous descriptions, OS Monitor would present the CPU Usage on notification bar, the total usage and also the first three active processes in its list.

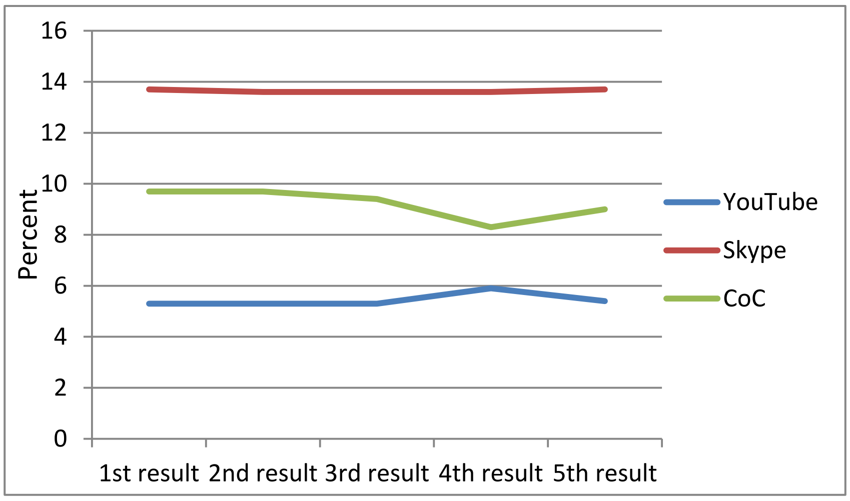

Figure 6.3 –OSM test results for "Exynos 7420"

As it is shown in figure 6.3, a complete smooth in results of OSM for Skype is observable. Then the next one is YouTube outcome. COC also has the stability but compare the other two, during their monitoring times, COC results will not be stable enough like the other two. The calculated SD for COC is equal to 0.58906 while it is 0.05477 for Skype and 0.26076 for YouTube. (Appendix 7)

Figure 6.4 representing the five iterations test results of the application during running our three target applications on device with "ARM Cortex-A9" processor.

While monitoring ARM Cortex-A9, "OS monitor" had more stable results during monitoring YouTube. Standard deviation of the five test iterations for YouTube is 0.57271. It is 2.77794 for Skype and 4.42797 for COC. (Appendix 7)

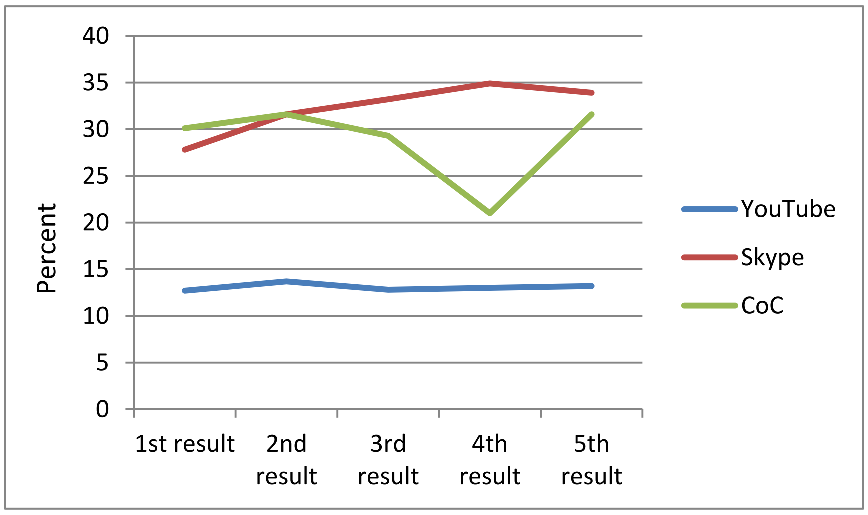

Figure 6.4 - OSM test results for "ARM Cortex-A9"

The reason behind this SD difference for COC is its graphical specifications; ARM Cortex-A9 had difficulties while rendering 2D/3D image process which caused a non-stable result during monitoring COC as the game graphic changes during time upgrades provided by its developers.

6.1.3 CPU Monitor

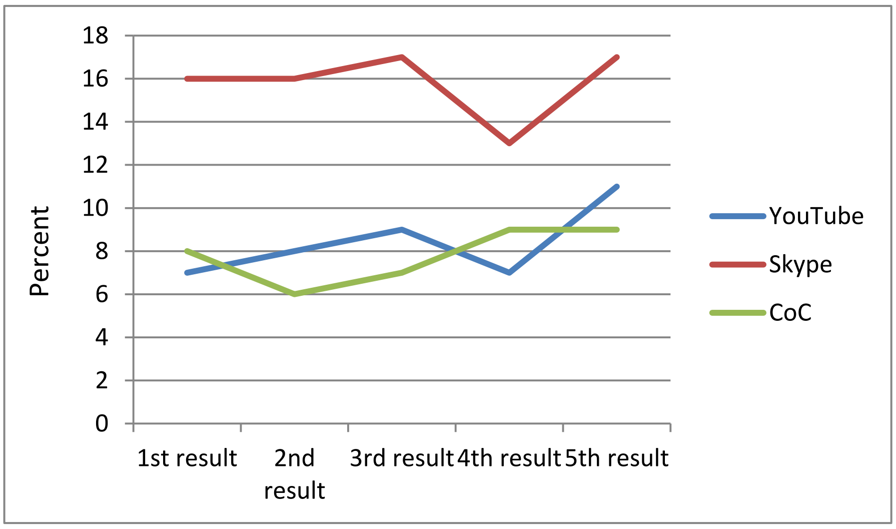

Figure 6.5 – CPM test results for "Exynos 7420"

As it shown in figure 6.5, three percent difference between higher and lower node of COC results, causes to mention that a stable result for this scenario. The SD for its outcomes is 1.3038. For YouTube it turns to four which makes it worth to mention another stable result in this scenario while the SD for these five iterations is 1.67332.

But compare to these two, results for Skype, especially on 3rd and 4th results, a drop of 4 percent and then an increase for 4 percent is an unusual. This makes the outcomes of Skype have a SD of 1.6431. These changes could be caused by any environmental changes even changes the CPU temperature. This could also show how the application is not stable regarding different environmental disturbances. So the application could still be considered as an unstable application. The related numbers about standard deviations of CPM in each scenario are available in appendix 7.

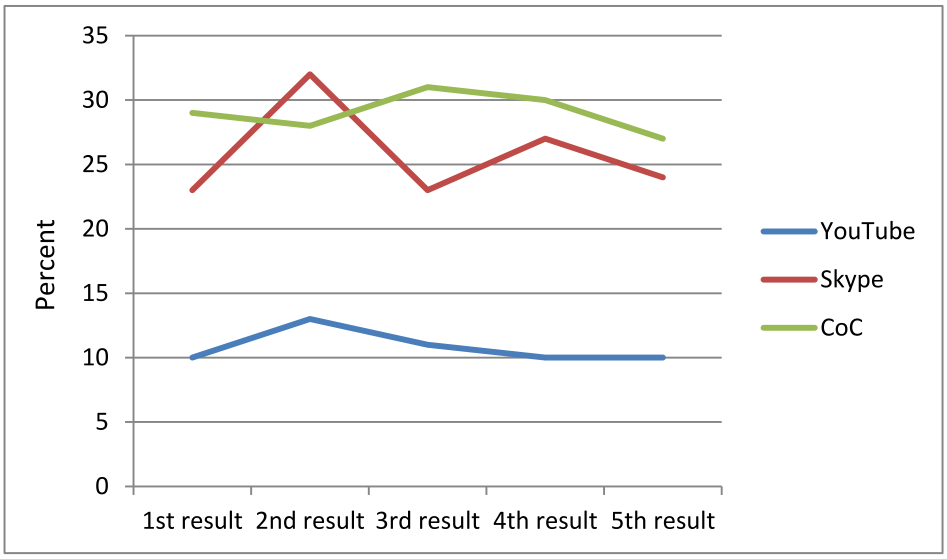

Figure 6.6 - CPM test results of "ARM Cortex-A9"

For the second test device, CPU monitor only has an acceptable results regarding to stability which we are looking for, on YouTube, while it has a SD of 1.30384.

The other two test scenarios, chiefly results for Skype, had a sudden drop down and drop up among its step and its SD is calculated as 3.83405. Somehow, COC's results had more balance and 1.58113 is the calculated result for its SD. The related numbers for standard deviations of CPM in each scenario are available in appendix 7.

6.1.4 Simple system monitor

As mentioned in section 4.6 Simple system monitor can show a list of active processes. Each process will be shown as its real time CPU usage. In the next figures, we are going to take a closer look at the gathered data from five times monitoring the real time of the application process, monitored by Simple system monitor.

The outcome of monitoring three applications on "Exynos 7420", especially on COC, provides an almost erratic result. Somehow, less than 3 percent difference dimension could be considered as a stable result for Skype. Figure 6.7 represents these results.

Calculated SD for YouTube iterations is 1.47012. It is 0.9139 for Skype and 2.25078 for COC. The SDs indicate that SSM has a stable result for Skype.

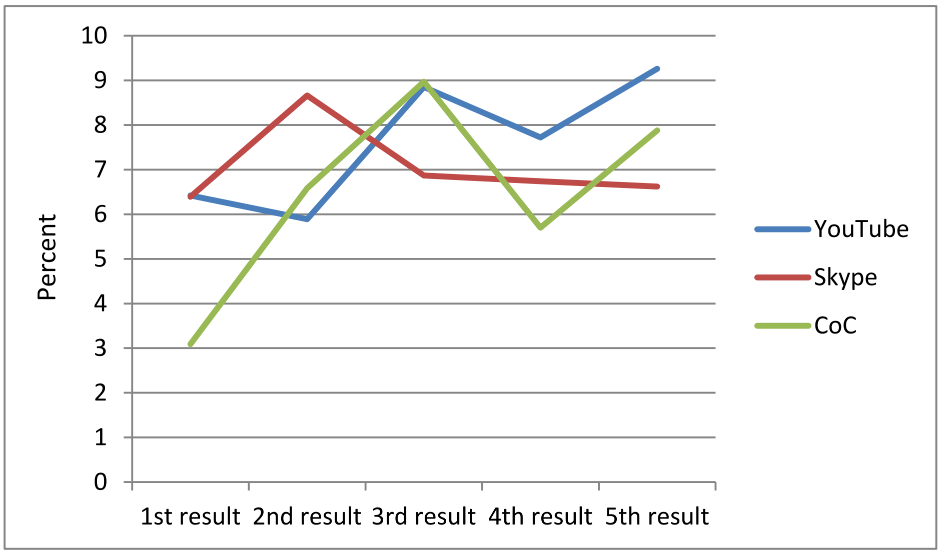

Figure 6.7 – SSM test results for "Exynos 7420"

Figure 6.8 represents the stability in the results of Simple system monitor during running our three target applications on device with "ARM Cortex-A9" processor.

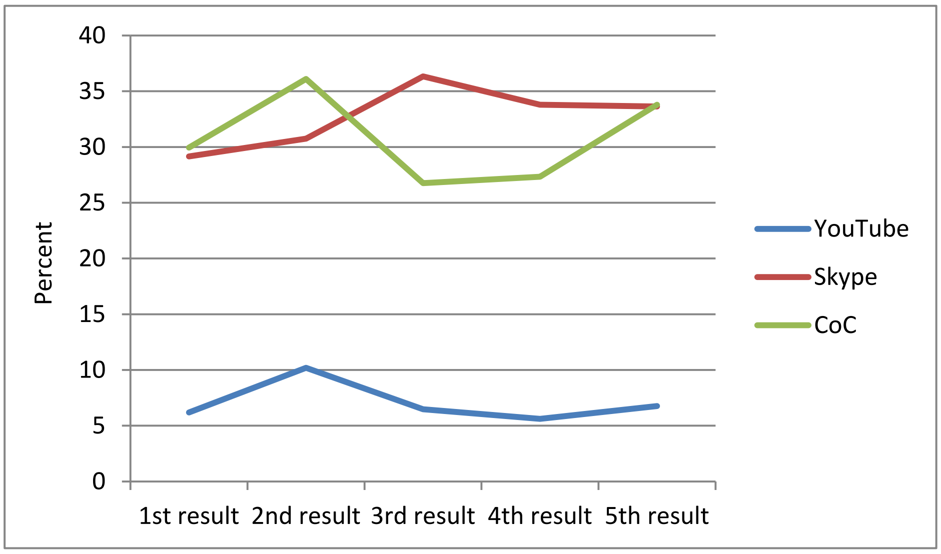

Figure 6.8 – SSM test results for "ARM Cortex-A9

Considering figure 6.8 and since the SD illustrates 1.81159 for YouTube, 2.8128 for Skype and 4.0687 for COC, stability in results is remarkable for YouTube. Appendix 7 contains the related information about these iterations.

6.1.5 System Monitor

Figure 6.9 represents the stability in the results of System monitor during running our three target applications on device with "Exynos 7420" processor.

Skype is only scenario which had a more stable results compare to the other two. Standard deviation of the outcomes of Skype is equal to 2.35435. YouTube and COC had a huge drop in final results at the first three steps, then going to behave and show more balance results in the CPU usage monitoring. Standard deviation for these two test cases shows 6.0116 for YouTube and 6.65184 for COC. The high deviation of SDs from zero in YouTube and COC, compared to Skype makes the iterations of Skype be called "stable" in this matter.

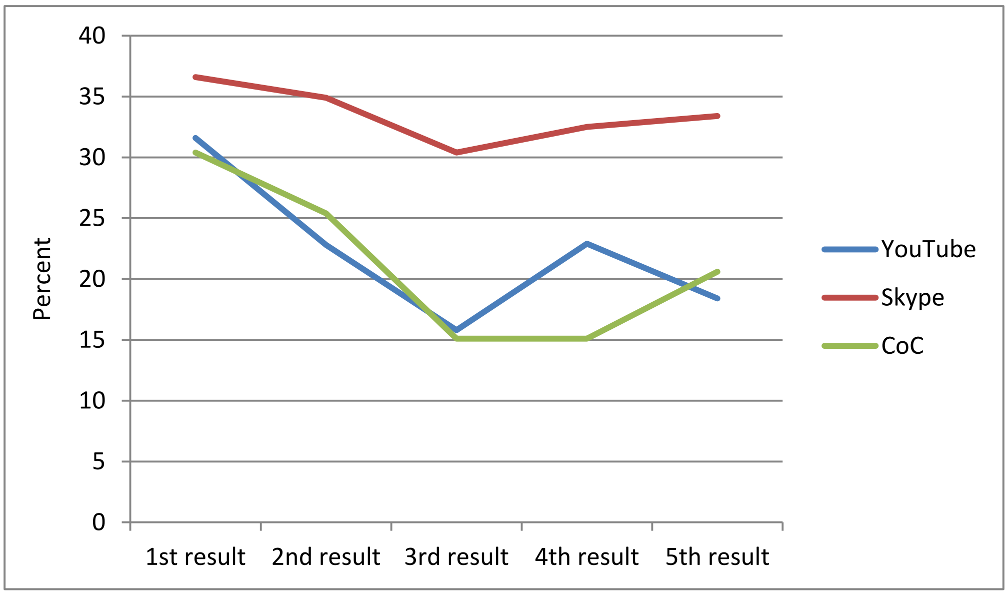

Figure 6.9 – SM test results for "Exynos 7420"

Figure 6.10 represents the stability in the results of System monitor during running our three target applications on device with "ARM Cortex-A9" processor.

Increasing, being balance, then a dropping down is showing on the results for Skype. In this matter the SD is 5.8266. A small and routine changing and a slight increasing are obvious for YouTube and COC in figure 6.10. Standard deviation of YouTube is 4.0558 while for COC is 2.89395.

Regarding the closeness to mean value, among these three scenarios, SM has a more stable monitoring process on COC scenario.

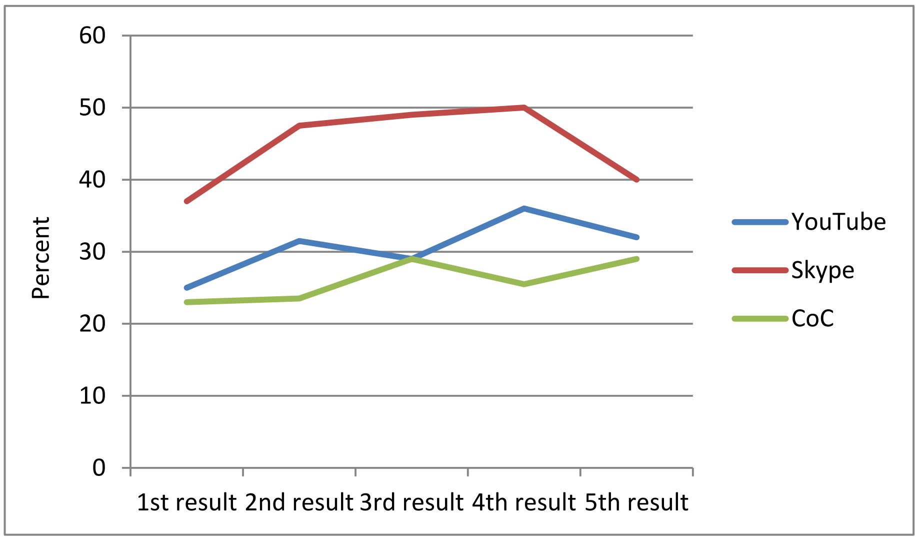

Figure 6.10 – SM test results for "ARM Cortex-A9"

6.1.6 MK system monitor

Figure 6.11 represents the stability in the results of CPU performance monitor during running our three target applications on device with "Exynos 7420" processor.

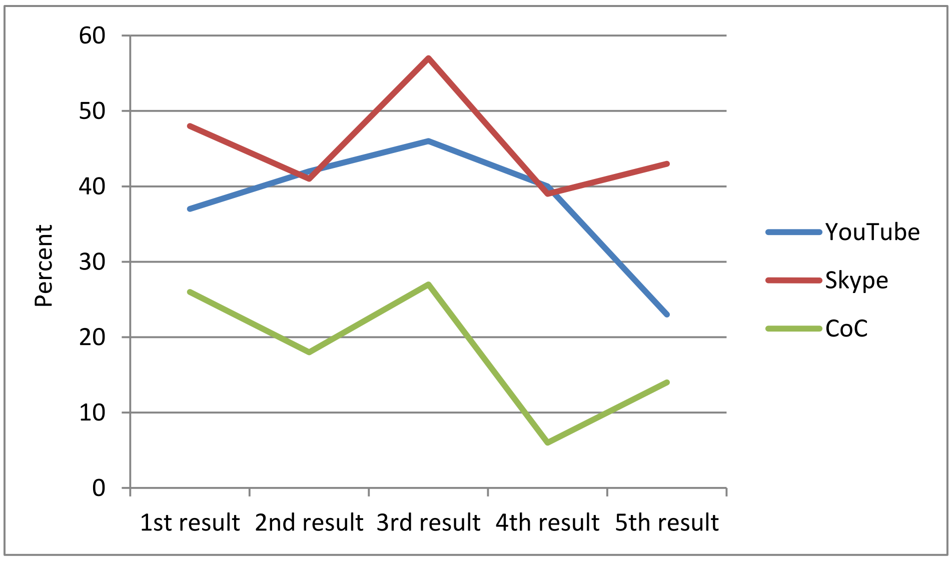

Figure 6.11 – MKS test results for "Exynos 7420"

MKS shows a non-stable result during all three test scenario running on "Exynos 7420". As it represents the total amount of CPU usage, a difference about 20 percent between maximum and minimum node of the diagram is a reason to show that MKS results will not be accurate enough. The SD regarding these iterations is calculated as 8.7292 for COC, 7.1972 for Skype and 8.792 for YouTube.

Figure 6.12 represents the stability in the results of MK System monitor during running our three target applications on device with "ARM Cortex-A9" processor.

Here, while the SD shows 2.6457 for YouTube, 3.8078 for Skype and 3.0495 for COC, it can be concluded that MKS has more stable outcomes for YouTube among these three scenarios.

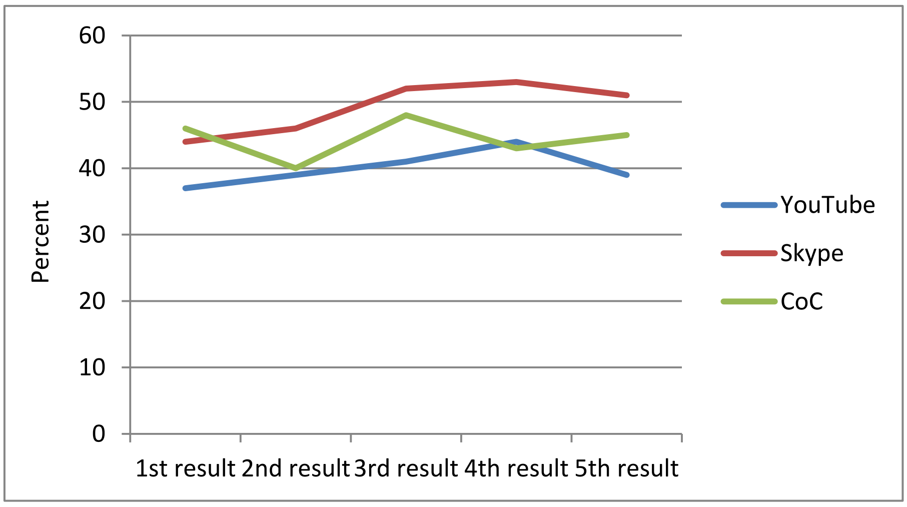

Figure 6.12 – MKS test results for "ARM Cortex A-9"

6.1.7 CPU Stats

Figure 6.13 represents the stability in the results of CPU performance monitor during running our three target applications on device with "Exynos 7420" processor.

Regarding the results generated by CPS, the SD is calculated as 1.3416 for COC, 0.8366 for Skype and 4.3931 for YouTube. According to figure 6.13 and also the SD values, CPS is more stable while it is monitoring Skype.

Figure 6.14 represents the stability in the results of CPU performance monitor during running our three target applications on device with "ARM Cortex-A9" processor.

In this case, the SDs of the CPS outcomes in each scenario, show that CPS has more stability for monitoring Skype while its SD is 0.8944. It is 4.1472 for YouTube and 2.4494 for COC. More details are available in Appendix 7.

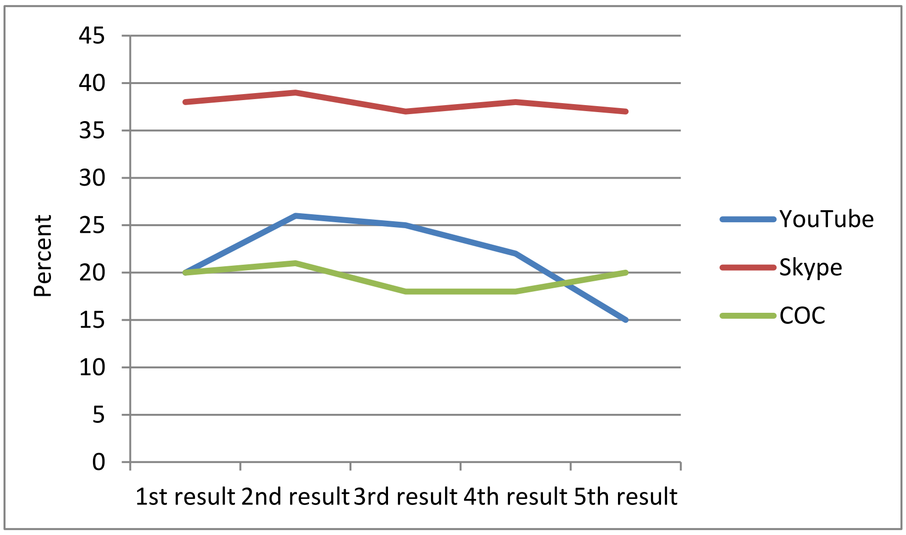

Figure 6.13 – CPS test results for “Exynos 7420”

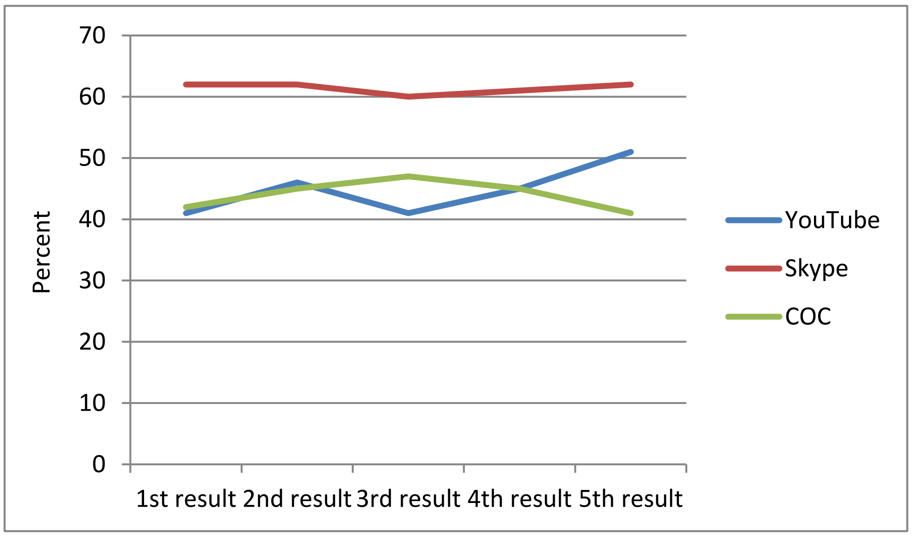

Figure 6.14 – CPS test results for “ARM Cortex-A9”

6.1.8 Usage timelines free

UTF results for both of the test devices are shown in Figure 6.15 and 6.16. The closest value to mean value, regarding the calculated SD in three scenarios performed on “Exynos 7420” is 1.3038 which belongs to YouTube outcomes. For “ARM Cortex A-9” it is 3.4205 which also belongs to YouTube scenario.

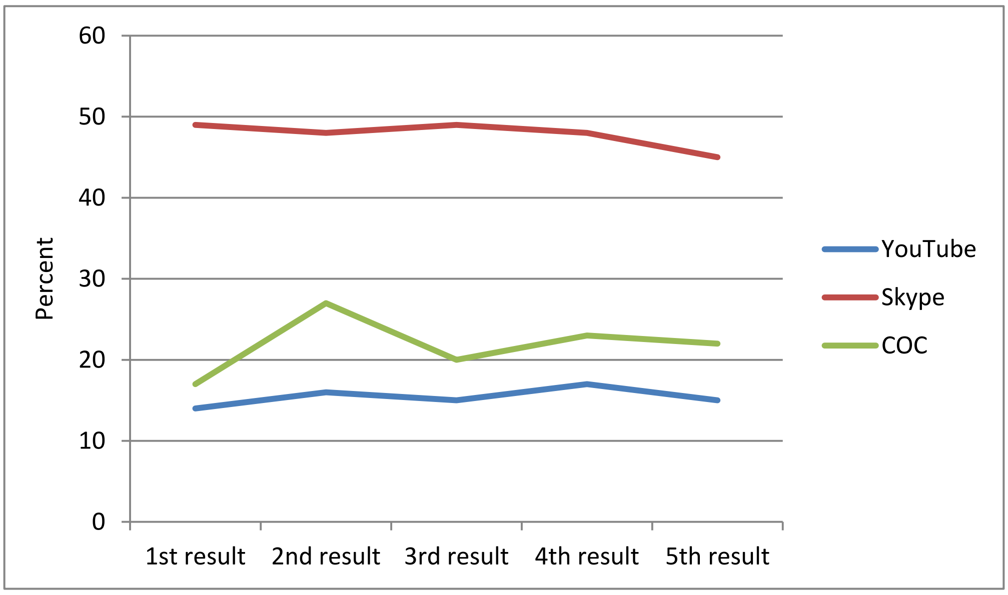

Figure 6.15 – UTF test results for "Exynos 7420"

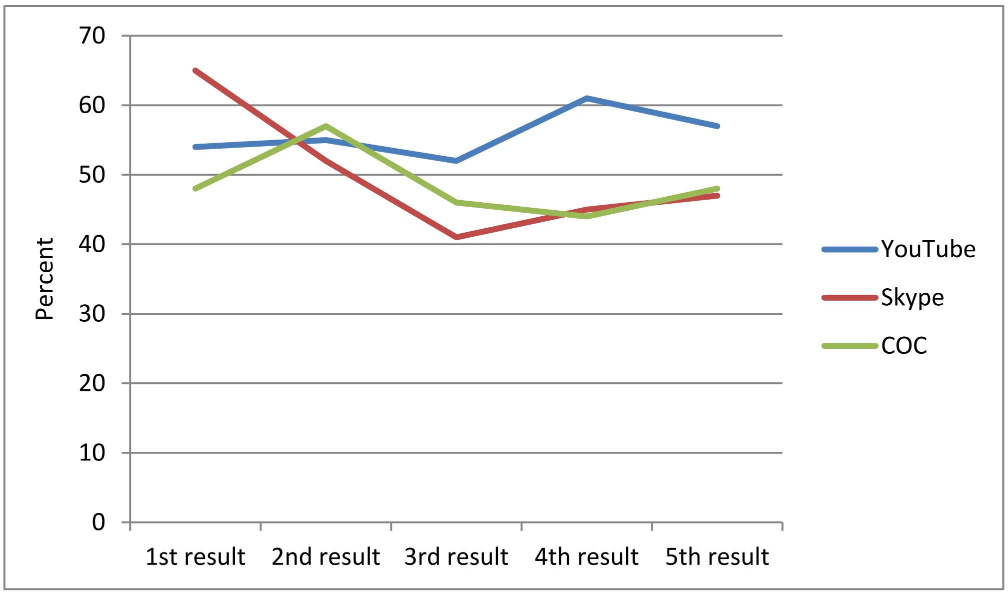

Figure 6.16 – UTF test results for "ARM Cortex-A9"

6.2 Analyze result's stability

In this section the outcomes of each test scenario is compared to each other. The purpose of this comparison is to find out which of these applications could provide more stable results. The term of stability refers to how an application's results are repeatable within an acceptable tolerance.

6.2.1 YouTube

The results and their stability are shown for each one of the monitoring apps. Afterwards, they should be comparing to each other. In the upcoming figures, the results of monitoring apps are going to show beside each other.

In Figure 6.17, the outcomes of four apps for test YouTube scenario are showing next to each other on the device with "Exynos 7420" Processor.

These four apps, as it described on section 4, presenting the real time CPU usage for the targeted applications.

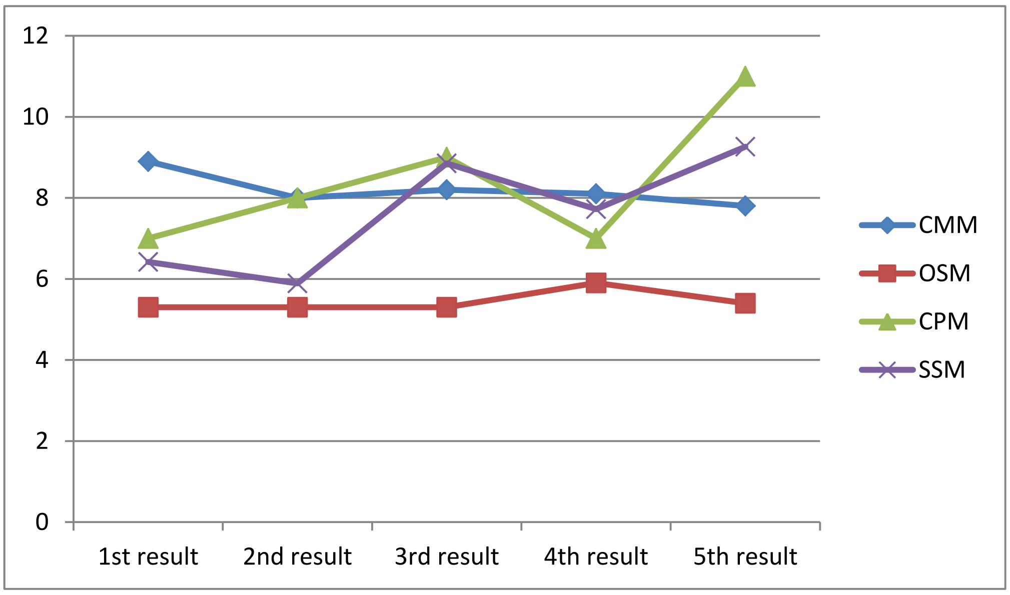

Figure 6.17 – Outcome comparison for YouTube scenario on "Exynos 7420"

IPV and CMA, present the average CPU usage during their time of monitoring the targeted application. Figure 6.18 shows their results.

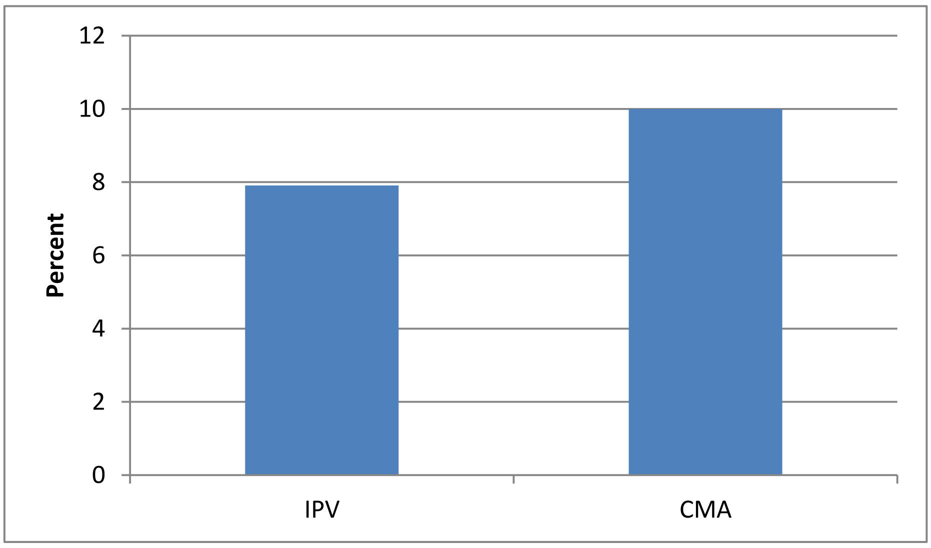

Figure 6.18 – IPV and CMA results for YouTube scenario on "Exynos 7420"

Comparing the threshold of these six outcomes, shows the estimate average for CPU usage would be from around 8 up to 10. This domain and also the stability of the results, makes it to choose IPV and CMM for a better monitoring result in this case.

CMA was in the chosen target to, but it monitored a 12 percent usage as max CPU usage during its monitoring time, which makes it a highest outcome between these six applications in this scenario.

For "ARM Cortex-A9" the first four application results can be observe and compare in figure 6.19.

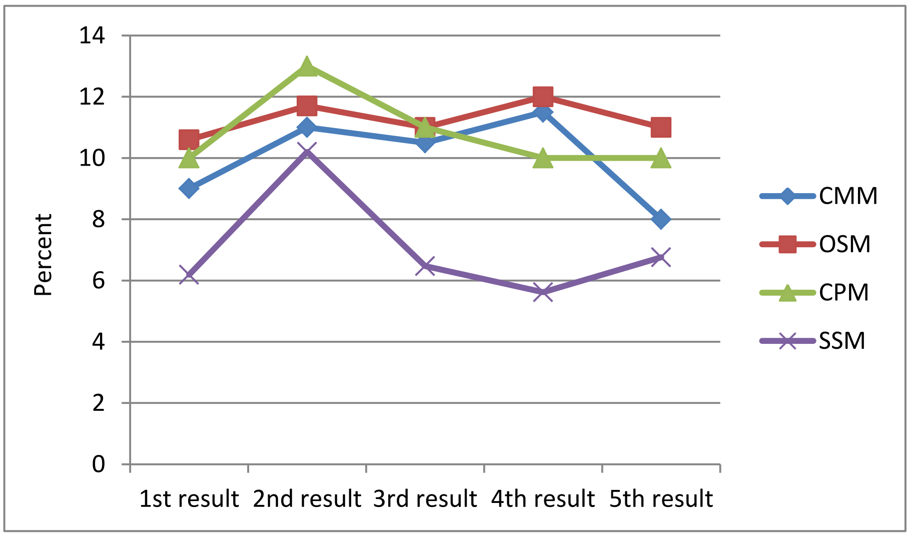

Figure 6.19 - Outcome comparison for YouTube scenario on "ARM Cortex A-9"

IPV and CMA, show their results of average CPU usage during monitoring the targeted application in figure 6.20.

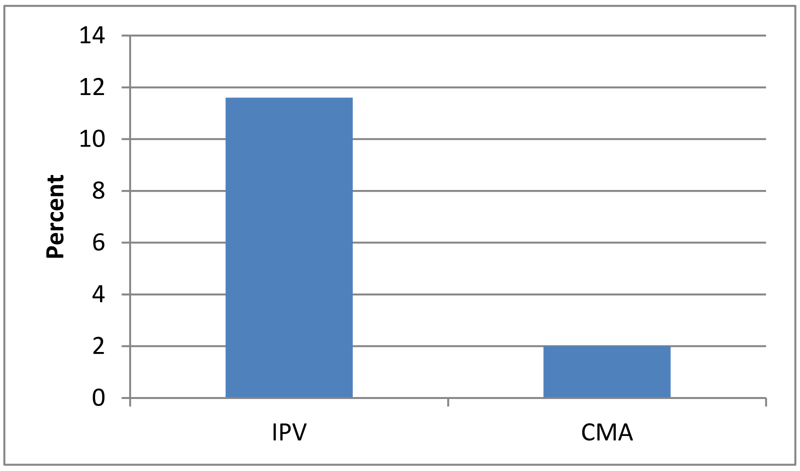

Figure 6.20 – IPV and CMA results for YouTube scenario on "ARM Cortex-A9"

A huge difference of average usage exists between IPV and CMA. CMA shows an average of only 2 percent while it was observe 8 percent as a maximum usage at time. These outcomes make CMA to move out from this scenario.

Base on IPV results beside look at figure 6.19, CMM, OSM and CPM have an acceptable outcome during the range 9 till 12 percent. But CMM and CPM once drop out of this range, while OSM result remains on this domain.

6.2.2 Skype

The outcomes are shown before. Now in the next figures, those results will put beside each other, for give a better view to see how their results stable compare to each other are. Figure 6.21 represents the results for real time CPU usage while monitoring Skype on device with "Exynos 7420" processor.

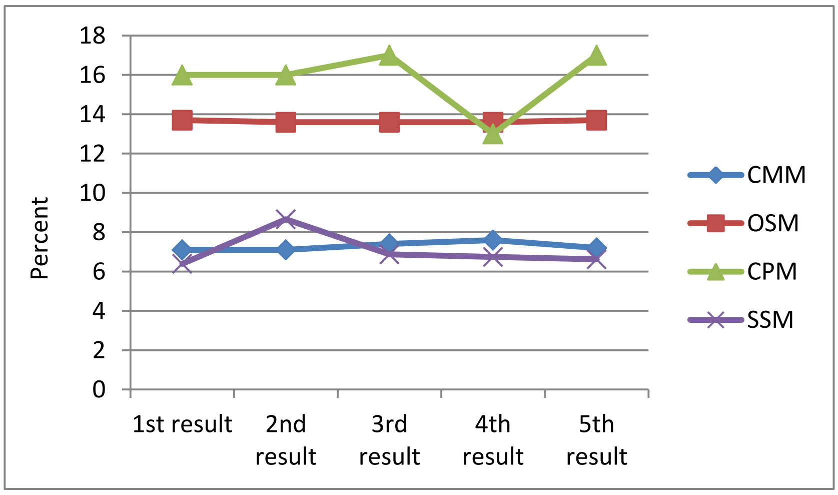

Figure 6.21 - Outcome comparison for Skype scenario on "Exynos 7420"

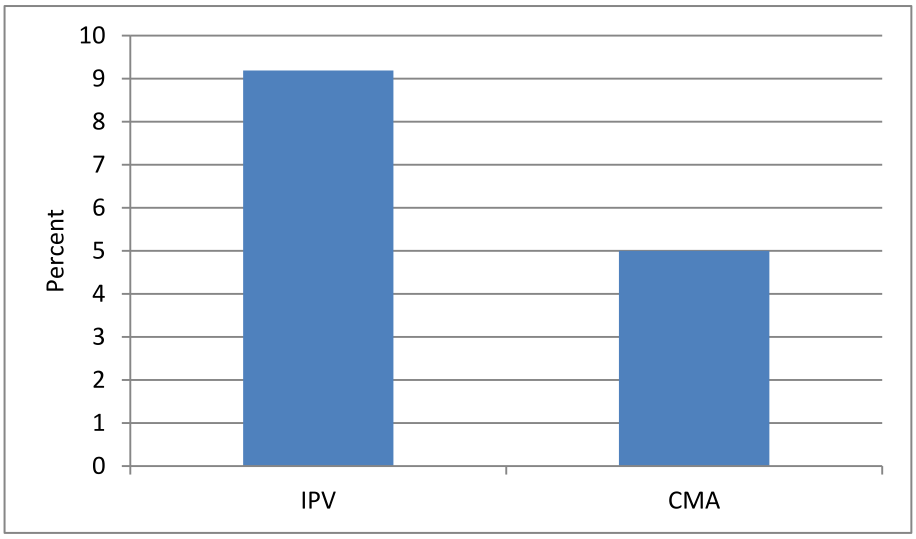

Figure 6.22 – IPV and CMA results for Skype scenario on “Exynos 7420”

With considering the average range shown by IPV and CMA, we can consider it is CMM with the nearest result to this range beside it shows stability on its results.

OSM is stable according to its SD (0.0547), but it is above the range provided by IPV and CMA.

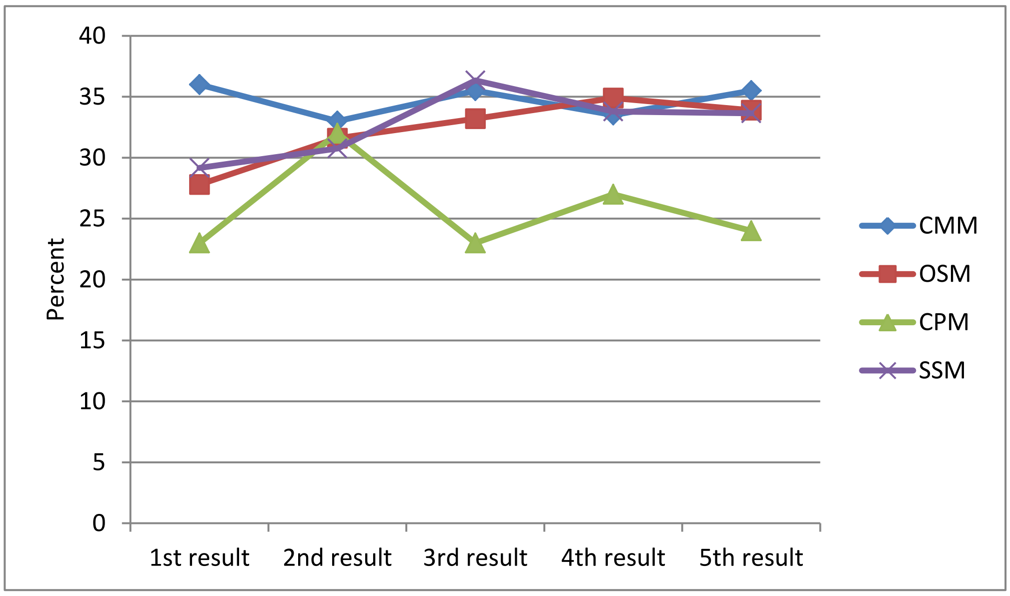

Figure 6.23 - Outcome comparison for Skype scenario on “ARM Cortex-A9”

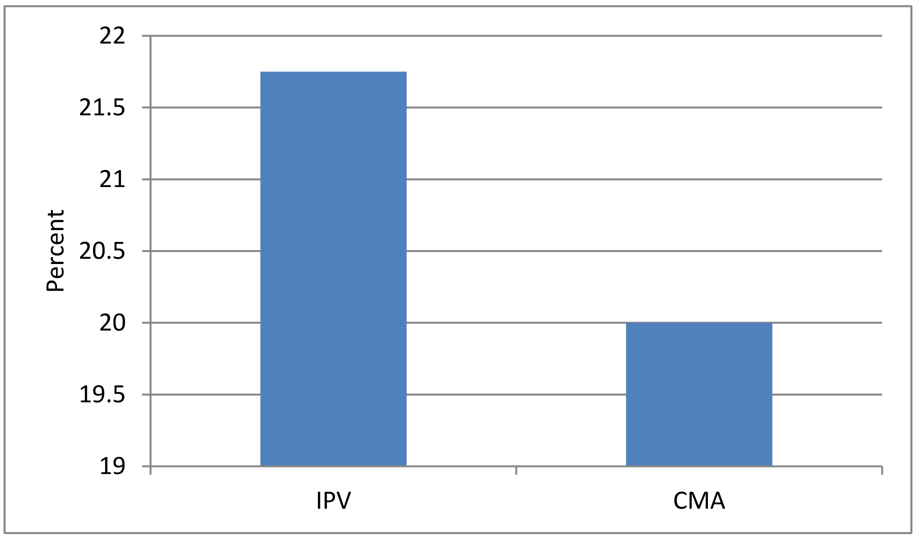

Figure 6.24 - IPV and CMA results for Skype scenario on "ARM Cortex-A9"

CMM and OSM show stability on their results, not a perfect stability for OSM for sure. The SD for these two outcomes is 1.3509 for CMM and 2.7779 for OSM. In the other side, IPV and CMA range of expectation is between 20 to 22 percent with a maximum usage of 39 percent monitored by CMA. It makes to say CMM has more stability plus a better range of showing CPU usage in this scenario.

6.2.3 COC

Figure 6.25 presents the comparison among the four applications, CMM, OSM, CPM and SSM while they are monitoring CPU performance on "Exynos 7420" in COC test Scenario.

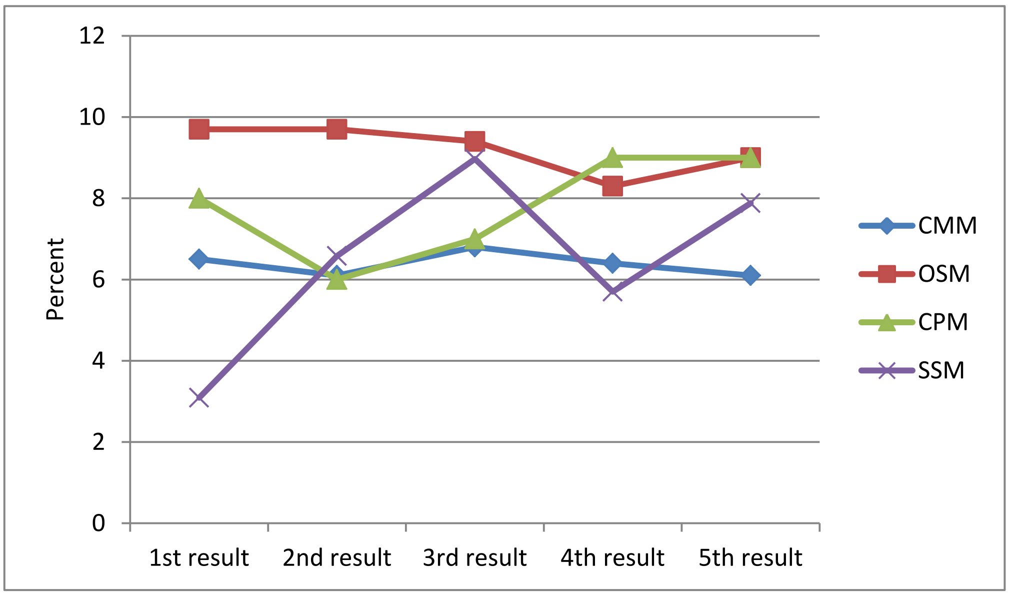

Figure 6.25 - Outcome comparison for COC scenario on "Exynos 7420"

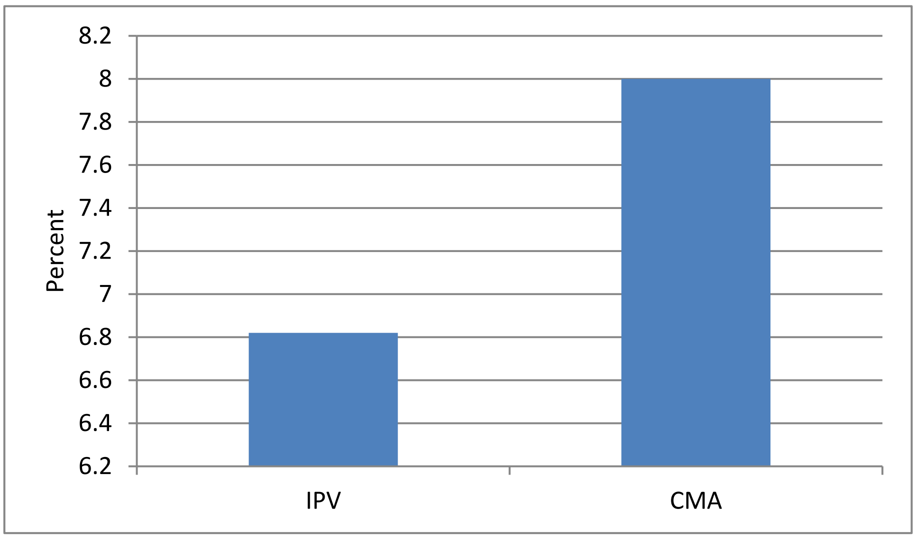

Figure 6.26 - IPV and CMA results for COC scenario on "Exynos 7420"

SSM and CPM as shown in the figure did not have a stable result and also regarding their calculated SDs, 2.2507 for SSM and 1.3038 for CPM. But CPM had a more stability on its results for the first three outcomes.

CMM and OSM show their results stable (Regarding their SDs available in previous sections and also in Appendix 7) and in the threshold of average usage which IPV and CMA presents for their results.

Both of CMM and OSM had the acceptable results in here.

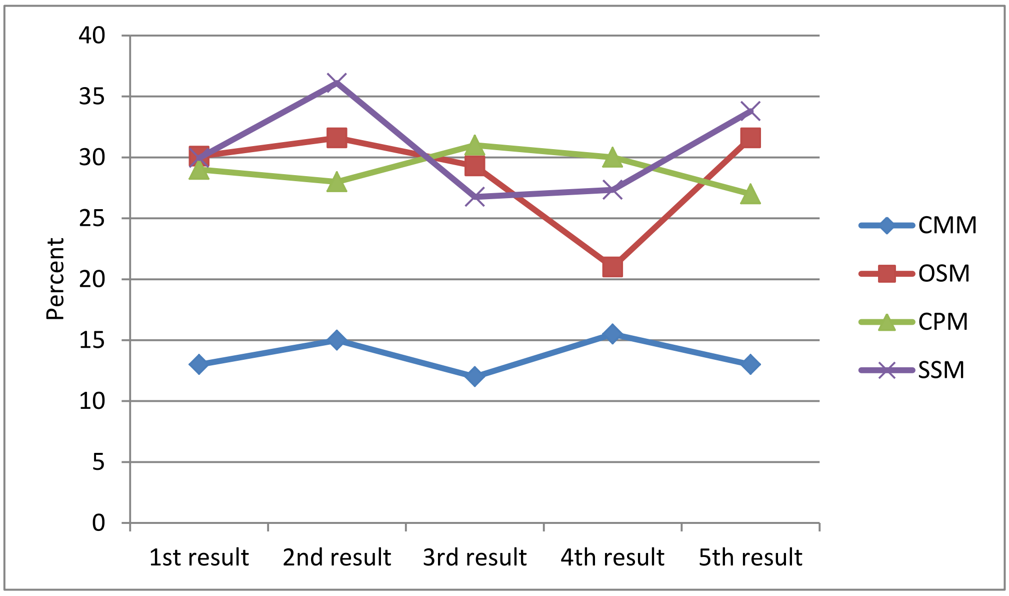

Figure 6.27 - Outcome comparison for COC scenario on "ARM Cortex-A9"

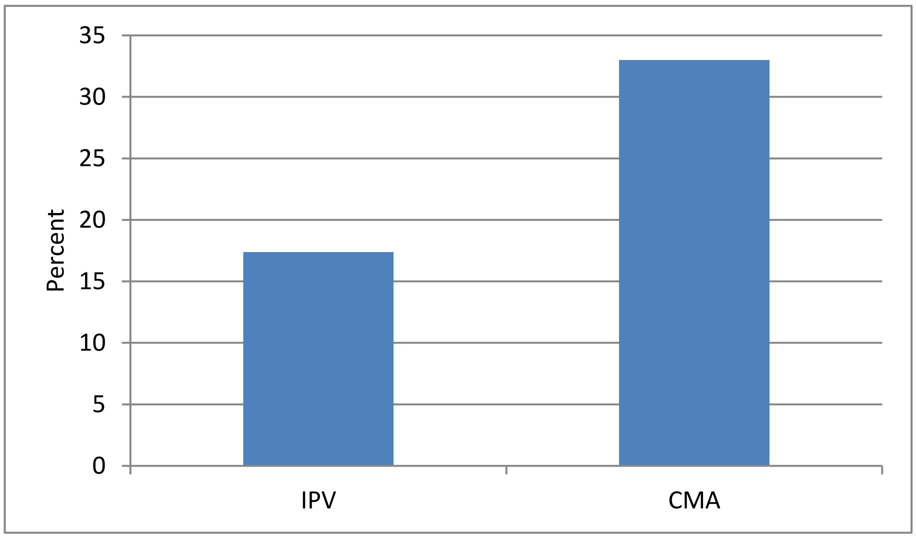

Figure 6.28 - IPV and CMA results for COC scenario on "ARM Cortex-A9"

If the average time of IPV be the base of conclusion in this scenario for "ARM Cortex-A9", CMM had a stable result in the domain near the outcome of IPV. But the other three are near the domain, provided by CMA and among those three, this is CPM which had stable results on its outcome.

6.2.4 Total CPU Usage results

System monitor, Usage timelines free, MK system monitor and CPU Stats are the four applications which represents the real time total CPU usage on the notification bar. Therefore, here we are going to analyze their results base on their outcome, CPU usage in total.

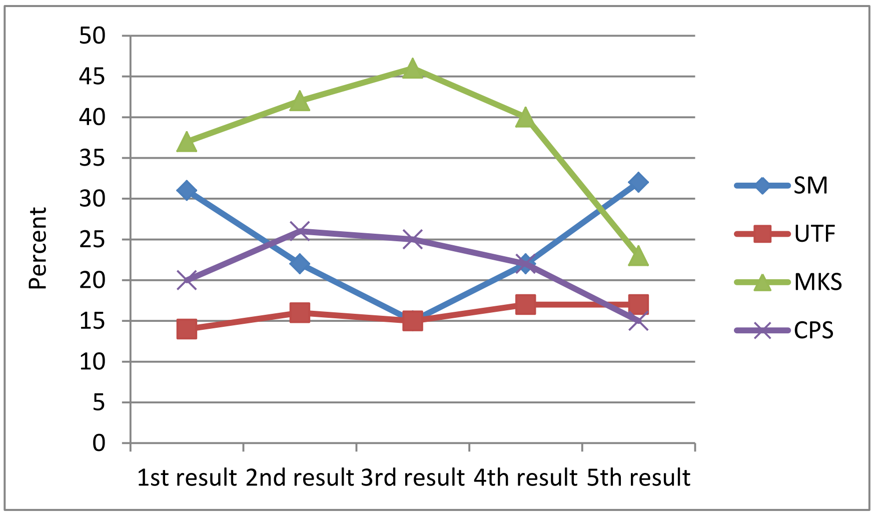

Figure 6.29 – Total CPU usage for YouTube on "Exynos 7420"

SM, MKS and CPS had unstable results in all their five times outcomes. As it is observable in figure 6.29, UTF shows more stability (UTF SD = 1.30384) beside its result in a domain near the results of the system when it has been monitored by single process, which makes it more sense.

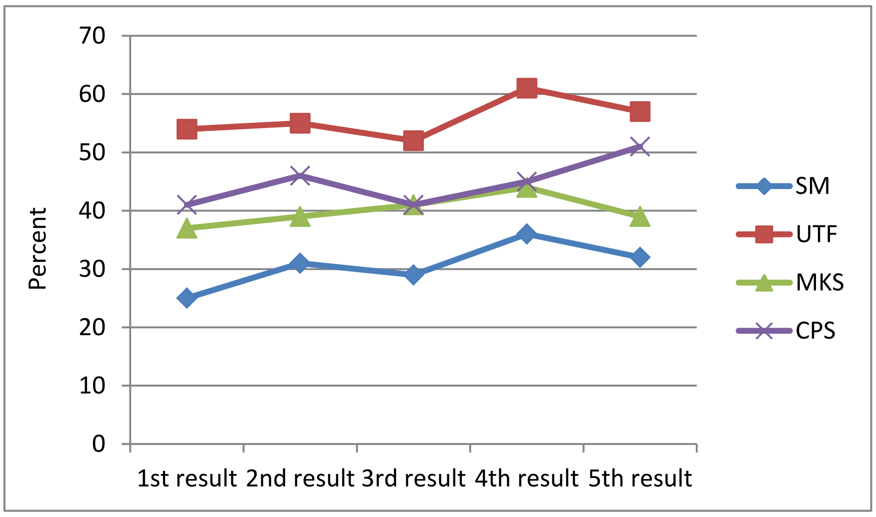

Figure 6.30 – Total CPU usage for YouTube on "ARM Cortex-A9"

Figure 6.30 shows that these four apps have four different outcomes, in four different domains and also all of them behave in an unstable manner during monitoring the target app. While calculating the SD of their result in this test case, MKS with an SD of 2.6457 has more stable results compared to the rest of the applications. All SD values are available in Appendix 7.

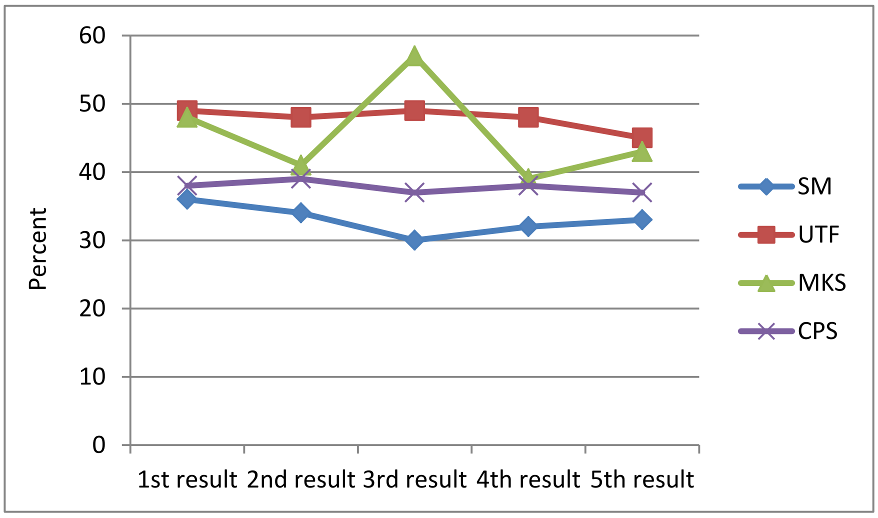

Figure 6.31 – Total CPU usage for Skype on "Exynos 7420"

Based on the stability of the results, UTF and CPS would be the two chosen monitoring apps in this scenario. Also it should be mentioned the CPS results are closer to the threshold of the single process monitoring of Skype by the other four apps. The SD value for CPS in this scenario is 0.83666.

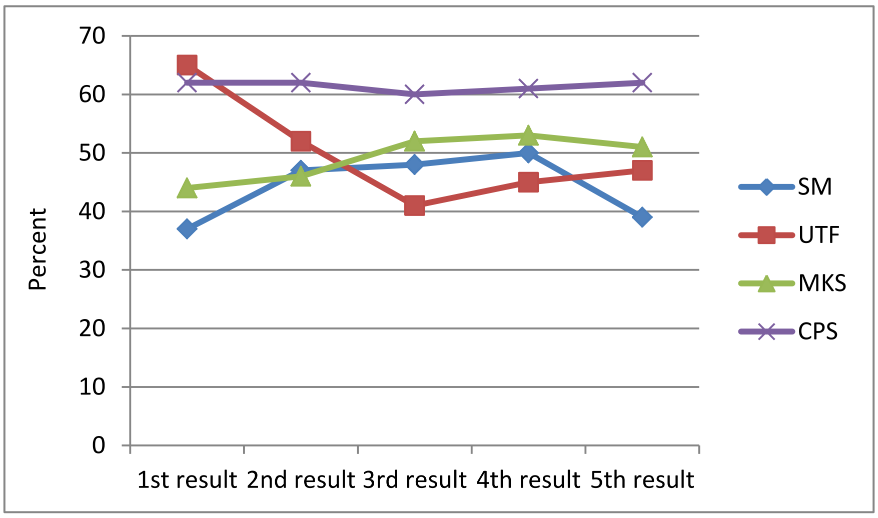

Figure 6.32 – Total CPU usage for Skype on "ARM Cortex-A9"

MKS with its stability and closing to the domain, similar the average result of single processing apps is the better monitoring result between these four apps. CPS has more smooth results with 0.8944 as its SD, but its results and the fact that they are above the expectation range, caused to avoid it in order to get an expected stable result.

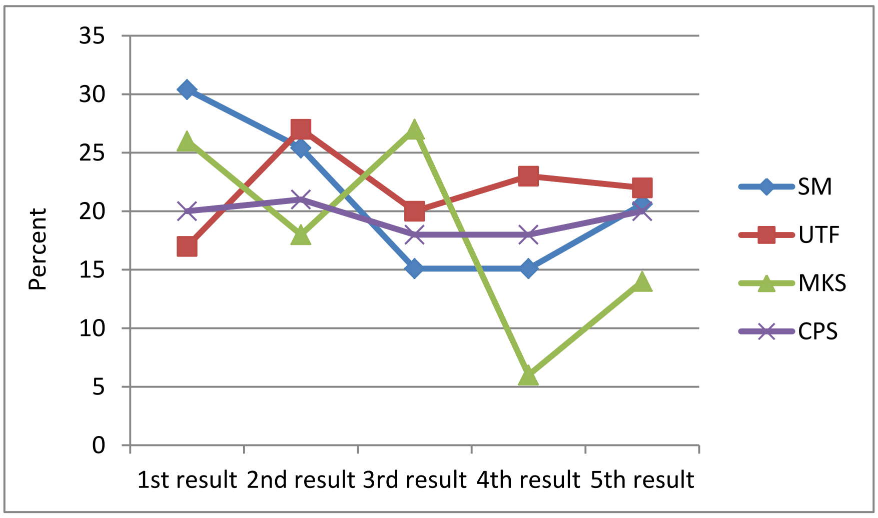

Figure 6.33 – Total CPU usage for COC on "Exynos 7420"

It is obvious that there is only CPS in here which has more stable results compare to the other three. The SD values of these four applications are

6.6518 for SM, 3.7013 for UTF, 8.7292 for MKS and 1.3416 for CPS. CPS results are much above the expected values based on the single processes on "Exynos 7420" for COC, but still CPS has the stable results in this scenario.

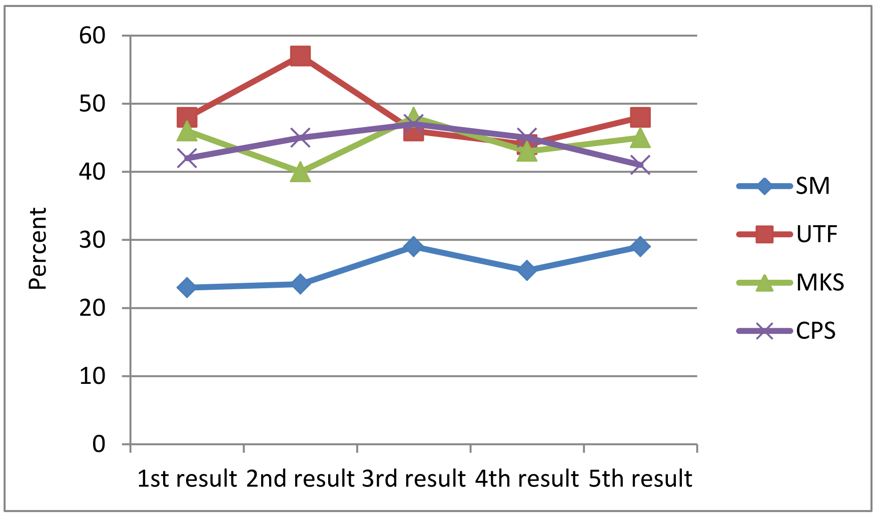

Figure 6.34 – Total CPU usage for COC on "ARM Cortex-A9"

As long as all the four lines present the instability during monitoring, but CPS has a results which is near the range of single process monitoring COC on "ARM Cortex-A9".

"Close to the range of single process monitoring" has been used in the section analysis. It is about existence of other processes on the system, like the monitoring application itself. Adding the CPU usage of other processes with the target app which we monitored in previous sections makes the total CPU usage of our expectation. All the SD values are available in Appendix 7.

Regarding to the data gathered, analyzed and shown in this and previous sections, and based on the comparison between them due to the diagrams, in table 6-1 and table 6-2 the applications are going to be marked based on their stability shown for their results.

Table 6-1 - Visible stability shown by monitoring applications on each scenario

	"Exynos 7420" processor			ARM Cortex-A9" processor		
	YouTube	Skype	COC	YouTube	Skype	COC
CMM	✓	✓	✓	✓	✓	✓
OSM	✓	✓	✓	✓	-✓	✗
CPM	✓	✓	✓	✓	-✓	✓
SSM	✓	✓	-✓	✓	-✓	✗
4<SD=Worse Stability(✗) 2<SD=Less Stability(-✓) SD<2=Most Stability(✓)						

Table 6-2 - Visible stability shown by monitoring applications on each scenario base on total CPU usage.

	"Exynos 7420" processor			ARM Cortex-A9" processor		
	YouTube	Skype	COC	YouTube	Skype	COC
SM	✗	✗	✗	✗	✗	✗
UTF	✓	-✓	✗	✗	✗	✗
MKS	✗	✗	✗	✗	✗	-✓
CPS	✗	✓	-✓	✗	✗	✓
4<SD=Worse Stability(✗) 2<SD=Less Stability(-✓) SD<2=Most Stability(✓)						

According to the applications' results in all of their five times outcome and the comparison which was made in previous section base on their SD these applications tried to get graded in three ways.

"Worse Stability" referred to those which provided their results in different outcome during their five outcomes, beside the range of the applications' answers which were far from the threshold of the others. Their calculated SD was above 4.

"Most Stability" referred to those applications which behave stable while they providing their results in each five times of monitoring. Their calculated SD was under 2.

"Less Stability" referred to those which neither had an unstable results nor have a stable outcome like the ones that marked with Most stability. Their calculated SD was between 2 and 4.

7 Discussion

The results shown in this report were from ten applications for three scenarios. Eight of these ten shows the real time usage, hence for more accurate result the results came from average of five times their outcome and each scenario was tested twice on each device. 240 results gathered for using and showing the best result. Each time, for capturing the result, a delay at least twice higher than interval setting of the application considered.

Two other applications show the average time during their monitoring time. For each one of them, five minutes of monitoring the application was considered and also a redo on test for having a better result.

For having a better understanding of how fast the CPUs are in mathematical tests and also their speed on storage reading and writing, six additional test scenarios were done on our test devices, with the same condition as average of five times results and the test was repeated for more accurate outcome.

Now their outcomes are available in the figures. By getting inspired by the technique “ratio Game” [12], IPV is the base of our measurement.

The first device, with an “Exynos 7420” processor, got monitored in three mentioned scenarios, analyzing its results showed this is CMM with the best outcome and can be used for monitoring real time CPU usage for single process. UTF could be the next choice when it is about monitoring total CPU usage.

The other device with an “ARM Cortex-A9”and its results during the test scenarios is CMM that can be useful for monitoring its CPU performance for single process usage. But not accurate enough like its results for “Exynos 7420”. Unfortunately there will not be any suggestion for monitoring total CPU usage while running an application on “ARM Cortex-A9” but under a stable condition and slow monitoring process, CPS can be used in this area.

8 Conclusion

During this thesis, finding the best tool for monitoring ARM architecture processors was the main goal. While the test devices available for this subject were two smartphones with Android OS, we decided to go through the available applications for monitoring CPU usage. Ten of the best among them were chosen and used during our tests. All the results gathered together and their stability checked, beside their outcome base on expectation.

RQ1. *Which features are these applications going to support and how accurately are these existing tools supporting performance analysis in comparison to one another?*

These tools used for two devices, while Samsung galaxy S6 had a better CPU and an updated OS, their results were more stable and accurate during the test scenarios on S6. Then for the other test device, S2, outcomes did not show enough accuracy and stability compare to S6. The CPU performance speed of these two test devices were shown in section 5.4.

In section 6, the final outcome with comparison made base on average results revealed which one of them can support well the software performance on ARM architectures processors. CMM provides the results in a more stable way, especially when it comes to monitoring the applications' performance on S6. OSM can be is the second choice due to its results.

CMM, also is a good choice for analyze the applications' performance on S2.

As it mentioned in section 4, total and single CPU usage, CPU usage per process, Total and single CPU frequency were the main features supported by them. While some of them monitor storage usage while running a scenario.

Supporting these features would help for reaching a more realistic result while they are monitoring a process.

RQ2. *For the future generations of these tools what improvements will lead to achieving more accurate results?*

While this report is written, more applications with more functionalities and features are going to be available for users, especially in the field of our discussion, Android OS. The accurate and most real results can be useful for developers, for their developing process for creating a better app, an application with minimum amount of CPU usage while it is not using a lot of battery energy.

Since these applications functionalities are mentioned in table 4.1, a report (log) file, containing the real time status of the device will be a useful feature for this matter. The log file includes the status of the CPU temperature, battery condition and other necessary details about the device while the test is running.

Therefore, in future works, merging these features and creating a more complete and accurate log report can be the best upgrade in the next generations of software performance tools.

References

[1] “Smartphone Users Worldwide Will Total 1.75 Billion in 2014“, January 2014, [Online]. Available: http://www.emarketer.com/Article/Smartphone-Users-Worldwide-Will-Total-175-Billion-2014/1010536
[Accessed: May 2015]

[2] A. L. SUSEELA, V. LALITH KUMAR ” EMBEDDED SYSTEMS IN REAL TIME APPLICATIONS, DESIGN & ARCHITECTURE“, [Online]. Available: http://ubiquity.acm.org/article.cfm?id=1086450”
[Accessed: May 2015]

[3] Matthew Haughn, Stephen J. Bigelow, “ARM processor“, [Online]. Available: http://whatis.techtarget.com/definition/ARM-processor
[Accessed: May 2015]

[4] "Introduction to the Special Issue on Software Architecture", IEEE Transactions on Software Engineering, vol.21, no. 4, pp. 269-274, April 1995, doi:10.1109/TSE.1995.10003

[5] Anshul Thakur, “ARM (Advanced RISC Machines) Processors”, [Online]. Available:http://www.engineersgarage.com/articles/arm-advanced-risc-machines-processors
[Accessed: May 2015]

[6]Kutty S Banerjee - Emmanuel Agu,” PowerSpy: Fine-Grained Software Energy Profiling for Mobile Devices”, Computer Science Dept, Worcester Polytechnic Institute, Proc IEEE WirelessCom Conference, Maui, Hawaii, 2005.

[7] S. Pllana, I. Brandic and S. Benkner. “A Survey of the State of the Art in Performance Modeling and Prediction of Parallel and Distributed Computing Systems” International Journal of Computational Intelligence Research (IJCIR), Vol. 4, No. 1, pp. 17-26, 2008.

[8] Wylie, B.J.N.; Bohme, D.; Mohr, B.; Szebenyi, Z.; Wolf, F., "Performance analysis of Sweep3D on Blue Gene/P with the Scalasca toolset," in Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on , vol., no., pp.1-8, 19-23 April 2010

[9]”ARM architecture”, *Wikipedia.* [Online]. Available: https://en.wikipedia.org/wiki/ARM_architecture
[Accessed : June 2015]

[10] “Embedded system”, *Wikipedia.* [Online]. Available:
https://en.wikipedia.org/wiki/Embedded_system
[Accessed: June 2015]

[11] Garrison Prinslow, “Overview of Performance Measurement and Analytical Modeling Techniques for Multi-core Processors”, [Online]. Available: http://www.cse.wustl.edu/~jain/cse567-11/ftp/multcore/#jain91
[Accessed: July 2015]

[12] Raj Jain, “Art of Computer Systems Performance Analysis Techniques For Experimental Design Measurements Simulation And Modeling”, Wiley Computer Publishing, John Wiley & Sons, Inc. 1991

[13] “Android (operating system)”, *Wikipedia.* [Online]. Available:
https://en.wikipedia.org/wiki/Android_(operating_system)#cite_note-manjoomurky-13
[Accessed: August 2015]

[14] “ Google Play”, *Wikipedia.* [Online]. Available:
https://en.wikipedia.org/wiki/Google_Play
[Accessed: September 2015]

[15] “Google Play Store”, [Online]. Available:
http://www.androidcentral.com/google-play-store
[Accessed: September 2015]

[16] “ARM, Cortex-A9 Processor”, *ARM.* [Online]. Available:
http://www.arm.com/cortex-a9.php
[Accessed: September 2015]

[17] “ARM Cortex-A9”, *Wikipedia.* [Online]. Available:
https://en.wikipedia.org/wiki/ARM_Cortex-A9
[Accessed: September 2015]

[18] ”Samsung Exynos Octa 7420”, [Online]. Availabe:
http://system-on-a-chip.specout.com/l/1145/Samsung-Exynos-Octa-7420
[Accessed: September 2015]

[19] “Exynos”, *Wikipedia.* [Online]. Available:
https://en.wikipedia.org/wiki/Exynos
[Accessed: September 2015]

[20] Google Play Store, Intel® Performance Viewer, [Online]. Available: https://play.google.com/store/apps/details?id=com.intel.mpv&hl=en
[Accessed: June 2015]

[21] Google Play Store, CPU Memory Monitor, [Online]. Available: https://play.google.com/store/apps/details?id=pavel.ugo.cpumonitor4&hl=en
[Accessed: August 2015]

[22] Google Play Store, OS Monitor, [Online]. Available: https://play.google.com/store/apps/details?id=com.eolwral.osmonitor&hl=en
[Accessed: August 2015]

[23] Google Play Store, CPU Monitor Advanced Lite, [Online]. Available: https://play.google.com/store/apps/details?id=com.bigbro.ProcessProfiler
[Accessed: July 2015]

[24] Alexander Bakker, "Comparing Energy Profilers for Android", 21st Twente Student Conference on IT, Vol. 21, Num. June 23, 2014, University of Twente.

[25] Google Play Store, CPU Monitor, [Online]. Available: https://play.google.com/store/apps/details?id=eu.uvdb.tools.cpumonitor&hl=en
[Accessed: July 2015]

[26] Google Play Store, Simple system monitor, [Online]. Available: https://play.google.com/store/apps/details?id=com.dp.sysmonitor.app
[Accessed: September 2015]

[27] Google Play Store, System Monitor, [Online]. Available: https://play.google.com/store/apps/details?id=com.p_soft.sysmon&hl=en
[Accessed: August 2015]

[28] Google Play Store, Usage Timelines Free, [Online]. Available: https://play.google.com/store/apps/details?id=com.als.usagetimelines&hl=en
[Accessed: July 2015]

[29] Google Play Store, MK system monitor, [Online]. Available: https://play.google.com/store/apps/details?id=cz.raven4.MKSysMon1&hl=en
[Accessed: August 2015]

[30] Google Play Store, CPU Stats, [Online]. Available: https://play.google.com/store/apps/details?id=jp.takke.cpustats&hl=en
[Accessed: August 2015]

[31] “Samsung Galaxy S II”, *Wikipedia.* [Online]. Available: https://en.wikipedia.org/wiki/Samsung_Galaxy_S_II
[Accessed: June 2015]

[32] “Samsung Galaxy S6”, *Wikipedia.* [Online]. Available: https://en.wikipedia.org/wiki/Samsung_Galaxy_S6
[Accessed: June 2015]

[33] ”Pi”, *Wikipedia.* [Online]. Available: https://en.wikipedia.org/wiki/Pi.
[Accessed: June 2015]

[34] ”CPU Benchmarks”, [Online]. Available: https://www.cpubenchmark.net/cpu_test_info.html. [Accessed: July 2015]

[35] “Sieve of Atkin”, *Wikipedia.* [Online]. Available: https://en.wikipedia.org/wiki/Sieve_of_Atkin. [Accessed: July 2015].

[36] “Quick sort”, *Wikipedia.* [Online]. Available: https://en.wikipedia.org/wiki/Quicksort
[Accessed: September 2015]

[37] Google Play Store, PassMark PerformanceTest, [Online]. Available: https://play.google.com/store/apps/details?id=com.passmark.pt_mobile&hl=en.
[Accessed: July 2015]

[38] Google Play Store, Pi, [Online]. Available: https://play.google.com/store/apps/details?id=com.gg.pi&hl=en. [Accessed: July 2015]

[39] “ARM Cortex-A9”, [Online]. Available: http://system-on-a-chip.specout.com/l/38/ARM-Cortex-A9-MPCore
[Accessed: November 2015]

[40] “First look at Samsung Orion ARM Cortex-A9, [Online]. Available: http://www.phonearena.com/news/First-look-at-Samsung-Orion-ARM-Cortex-A9_id14608
[Accessed: November 2015]

[41] “Standard Deviation”, Wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/Standard_deviation [Accessed: December 2015]

Appendices:

In this section, all of the necessary data and calculated results are presented in their tables based on their usage I this report.

Appendix1:

The table indicates the five iterations of the applications while they were monitoring the CPU performance, in YouTube scenario on “Exynos 7420”.

Application name	1st	2nd	3rd	4th	5th	Average	Mean	Standard Deviation
CPU Memory Monitor	8.9	8	8.2	8.1	7.8	8.2	8.1	0.41833
OS Monitor	5.3	5.3	5.3	5.9	5.4	5.44	5.3	0.260768
CPU Monitor	7	8	9	7	11	8.4	8	1.67332
Simple system monitor	6.42	5.89	8.85	7.72	9.26	7.628	7.72	1.470126
System Monitor	31.6	22.8	15.8	22.9	18.4	22.3	22.8	6.011655
Usage timelines free	14	16	15	17	17	15.8	16	1.30384
MK system monitor	37	42	46	40	23	37.6	40	8.792042
CPU Stats	20	26	25	22	15	21.6	22	4.393177

Applications’ Results, Standard Deviation and Mean value in YouTube scenario on Exynos 7420

Appendix2:
The table indicates the five iterations of the applications while they were monitoring the CPU performance, in YouTube scenario on "ARM Cortex A-9".

Application name	1st	2nd	3rd	4th	5th	Average	Mean	Standard Deviation
CPU Memory Monitor	9	11	10.5	11.5	8	10	10.5	1.457738
OS Monitor	10.6	11.7	11	12	11	11.26	11	0.572713
CPU Monitor	10	13	11	10	10	10.8	10	1.30384
Simple system monitor	6.19	10.2	6.47	5.62	6.76	7.048	6.47	1.811593
System Monitor	25	31.5	29	36	32	30.7	31.5	4.05586
Usage timelines free	54	55	52	61	57	55.8	55	3.420526
MK system monitor	37	39	41	44	39	40	39	2.645751
CPU Stats	41	46	41	45	51	44.8	45	4.147288

Appendix3:
The table indicates the five iterations of the applications while they were monitoring the CPU performance, in Skype scenario on "Exynos 7420".

Application name	1st	2nd	3rd	4th	5th	Average	Mean	Standard Deviation
CPU Memory Monitor	7.1	7.1	7.4	7.6	7.2	7.28	7.2	0.216795
OS Monitor	13.7	13.6	13.6	13.6	13.7	13.64	13.6	0.054772
CPU Monitor	16	16	17	13	17	15.8	16	1.643168
Simple system monitor	6.39	8.66	6.87	6.74	6.62	7.056	6.74	0.913909
System Monitor	36.6	34.9	30.4	32.5	33.4	33.56	33.4	2.354358
Usage timelines free	49	48	49	48	45	47.8	48	1.643168
MK system monitor	48	41	57	39	43	45.6	43	7.197222
CPU Stats	38	39	37	38	37	37.8	38	0.83666

Appendix4:
The table indicates the five iterations of the applications while they were monitoring the CPU performance, in Skype scenario on “ARM Cortex A-9”.

Application name	1st	2nd	3rd	4th	5th	Average	Mean	Standard Deviation
CPU Memory Monitor	36	33	35.5	33.5	35.5	34.7	35.5	1.350926
OS Monitor	27.8	31.6	33.2	34.9	33.9	32.28	33.2	2.777949
CPU Monitor	23	32	23	27	24	25.8	24	3.834058
Simple system monitor	29.16	30.75	36.34	33.79	33.64	32.736	33.64	2.812869
System Monitor	37	47.5	49	50	40	44.7	47.5	5.826663
Usage timelines free	65	52	41	45	47	50	47	9.273618
MK system monitor	44	46	51	53	51	49	51	3.807887
CPU Stats	62	62	60	61	62	61.4	62	0.894427

Appendix 5:
The table indicates the five iterations of the applications while they were monitoring the CPU performance, in COC scenario on “Exynos 7420”.

Application name	1st	2nd	3rd	4th	5th	Average	Mean	Standard Deviation
CPU Memory Monitor	7.1	7.1	7.4	7.6	7.2	7.28	7.2	0.216795
OS Monitor	13.7	13.6	13.6	13.6	13.7	13.64	13.6	0.054772
CPU Monitor	16	16	17	13	17	15.8	16	1.643168
Simple system monitor	6.39	8.66	6.87	6.74	6.62	7.056	6.74	0.913909
System Monitor	36.6	34.9	30.4	32.5	33.4	33.56	33.4	2.354358
Usage timelines free	49	48	49	48	45	47.8	48	1.643168
MK system monitor	48	41	57	39	43	45.6	43	7.197222
CPU Stats	38	39	37	38	37	37.8	38	0.83666

Appendix 6:
The table indicates the five iterations of the applications while they were monitoring the CPU performance, in COC scenario on “ARM Cortex A-9”.

Application name	1st	2nd	3rd	4th	5th	Average	Mean	Standard Deviation
CPU Memory Monitor	35.5	33	35.5	33.5	35.5	34.7	35.5	1.350926
OS Monitor	33.2	31.6	33.2	34.9	33.2	32.28	33.2	2.777949
CPU Monitor	24	32	24	27	24	25.8	24	3.834058
Simple system monitor	29.16	30.75	36.34	33.79	33.64	32.736	33.64	2.812869
System Monitor	47.5	47.5	47.5	50	47.5	44.7	47.5	5.826663
Usage timelines free	47	52	47	45	47	50	47	9.273618
MK system monitor	51	46	51	53	51	49	51	3.807887
CPU Stats	62	62	60	61	62	61.4	62	0.894427

Appendix 7:
The table indicates the SD of each application for each three test scenarios on each test device.

	Exynos 7420			ARM Cortex A-9		
	COC	Skype	YouTube	COC	Skype	YouTube
CPU Memory Monitor	0.294958	0.216795	0.41833	1.48324	1.350926	1.457738
OS Monitor	0.589067	0.054772	0.260768	4.427979	2.777949	0.572713
CPU Monitor	1.30384	1.643168	1.67332	1.581139	3.834058	1.30384
Simple system monitor	2.250784	0.913909	1.470126	4.068798	2.812869	1.811593
System Monitor	6.651842	2.354358	6.011655	2.893959	5.826663	4.05586
Usage timelines free	3.701351	1.643168	1.30384	4.97996	9.273618	3.420526
MK system monitor	8.729261	7.197222	8.792042	3.04959	3.807887	2.645751
CPU Stats	1.341641	0.83666	4.393177	2.44949	0.894427	4.147288