AALTO UNIVERSITY SCHOOL OF ELECTRICAL ENGINEERING
Department of Communications and Networking

Aashish Adhikari

MOBILE DEVICE IDENTIFICATION FROM NETWORK TRAFFIC MEASUREMENTS - A HTTP USER AGENT BASED METHOD

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Technology.

Espoo, August 7th, 2012

Supervisor: Heikki Hämmäinen
Professor, Networking Business

Instructor: Antti Riikonen
M.Sc. (Tech)

AALTO UNIVERSITY SCHOOL OF ELECTRICAL ENGINEERING
Department of Communications and Networking
ABSTRACT OF THE MASTER´S THESIS

Author: Aashish Adhikari

Subject of the thesis:
Mobile Device Identification from Network Traffic Measurements – A HTTP User Agent Based Method

Number of pages: 6 + 72	Date: 07.08.2012
Professorship: Networking Technology	Code of professorship: S-38

Supervisor: Professor Heikki Hämmäinen

Instructor: Master of Science (Tech.) Antti Riikonen

The proliferation of mobile Internet in recent years has created an increasing need to understand the usage of the mobile services. With widespread adoption of the Internet capable mobile handheld devices, knowledge about mobile Internet usage is beneficial from different aspects of the stakeholders.

A key challenge is that the factual information available on User Agent (UA) based device identification from IP traffic measurements is limited. The objective of our research is two folded; to develop a tool to identify mobile devices based on the HTTP UA obtained from the network traffic measurements and to profile mobile Internet usage in Finland. We observe that the tool can be developed by using a device description repository (DDR) and its API to interact with the repository and extract device related information. Moreover, the results from the DDR implementation can be improved by enhancing its original output. With the identification results, we provide descriptive statistics to aid in profiling the usage of the mobile Internet in Finnish mobile networks.

Wireless Universal Resource File (WURFL) DDR based tool produced accurate identification of the devices from the HTTP UA strings. However, identification of the devices from the UA strings generated by the applications other than the web browsers required additional programming. The resulting enhanced WURFL tool was able to improve the device identification results roughly by 15% points with our dataset. Based on the assessment of the enhanced WURFL tool, we observe that roughly 94% of the total UA strings subjected to the analysis were identified correctly. The share of incorrectly identified UA strings was about 0.5%. The data analysis results indicate that the majority of mobile handset traffic is generated by handsets with advanced capabilities such as 3G and the touchscreen, manufactured by numerous brands of mobile devices with different operating systems. The results from the identification of these devices and device features could be utilized by the operators to support the pricing and business development.

Keywords: Network Traffic Measurements, Mobile Internet, Device identification, HTTP User Agent, DDR, WURFL, Java API	Publishing language: English

Preface

This Master's Thesis has been written as a partial fulfillment for the Master of Science (Technology) degree at Aalto University School of Electrical Engineering, Finland. The work was conducted as a part of the MoMIE project in the Department of Communications and Networking at the Aalto University School of Electrical Engineering.

I would like to express my gratitude to all the people who have supported and encouraged me in this work. First and foremost, I want to thank Professor Heikki Hämmäinen for giving me the opportunity to work under his guidance. I am grateful to my instructor Antti Riikonen for his extensive support, motivation, and guidance throughout the whole research process.

I would also like to thank my colleagues from the Networking Business team for creating a lively working atmosphere. Team events with them were always fun and filled with excitement and enthusiasm. Furthermore, I wish to thank Benjamin for helping me with the scripts.

I owe my deepest gratitude to the Gurus, my mother Kalpana, and sisters Shalu, Shama, and Sangya for giving me all the love and wisdom in every step of my life. Finally, I would not want to miss thanking Merina, Nalin, Ujjwal, Prem, Saurav, and the entire Nepalese community at the Aalto University for their support and never ending motivation.

Espoo, August 7th, 2012

Aashish Adhikari

Table of Contents

List of Figures

List of Tables

Abbreviations and Terms

2G	2nd Generation
3G	3rd Generation
API	Application Programming Interface
App	Mobile application software
BABT	British Approvals Board of Telecommunications
CDR	Charging Data Record
DDR	Device Description Repository
Gb	Gb interface is a GPRS interface between a BSS and a SGSN
Gi	Gi interface is a GPRS interface between a GGSN and an external data network
GGSN	Gateway GPRS Support Node
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile Communications
GSMA	GSM Association
GTP	GPRS Tunneling Protocol
HTTP	Hypertext Transfer Protocol
IMEI	International Mobile station Equipment Identity
IMEISV	IMEI Software Version number
IP	Internet Protocol
IuPS	IuPS interface is a UMTS interface which links the RNC with a SGSN
MNO	Mobile Network Operator
MoMIE	Modeling of Mobile Internet Ecosystem, Aalto University research project
OS	Operating System
PC	Personal Computer
PDA	Personal Digital Assistant

PSP	PlayStation Portable
RIM	Research In Motion
RTOS	Real Time Operating System
SNR	Serial Number
SGSN	Serving GPRS Support Node
TAC	Type Allocation Code
TCP	Transmission Control Protocol
Tstat	TCP Statistic and Analysis Tool
UA	User Agent
UAProf	User Agent Profile
UMTS	Universal Mobile Telecommunication System
WLAN	Wireless Local Area Network
WURFL	Wireless Universal Resource File

1 Introduction

This chapter provides an introduction for our research. Motivation for our research is discussed first, followed by the research questions, the objectives and the scope of the research. After this, the research methodology is introduced, and finally structure of the thesis is outlined.

1.1 Motivation

The proliferation of mobile Internet in recent years has resulted in the emergence of numerous smartphones, tablets and other mobile Internet capable devices. On the other hand, the adoption of mobile Internet has also caused the number of mobile device users to increase, due to the convenience of information searching and guaranteed mobility. The type of mobile device used to access the Internet depends upon the context, purpose and demography of the user. Acquiring information on the type, model and features of the devices that the users prefer to access mobile Internet services is beneficial from different aspects of all the stakeholders. One way of extracting device related information is by analyzing the Internet Protocol (IP) traffic measurements from mobile operators' network, which provides the platform for this research.

Mobile device identification provides valuable information for measuring and analysing mobile Internet usage. From the mobile operator viewpoint, it is important to identify the devices that the users prefer to access their services. Kivi (2006) outlines that the sufficiently accurate subscriber terminal device (and device feature) identification could be utilized by mobile operators in a number of ways including price differentiation when separating modem traffic from truly mobile terminal originated usage. This would facilitate the sales of handsets and network operation services. Identification of the devices could also be beneficial in tailoring the existing services according to the needs of the users and at the same time introducing new ones to attract more users. In previous research, Huang et al. (2012) and Adzic et al. (2011) studied device identification for the development process of mobile web applications, whereas device identification

from the mobile network traffic measurements to analyze the mobile Internet usage remains overlooked.

Identifying the devices, used to access mobile Internet services, from the network traffic measurements data is an interesting area of research. There are different ways to identify devices from the measurement data. For instance, Kivi (2006) used Type Allocation Code (TAC) to identify devices with the data acquired from the Charging Data Record (CDR) data warehouses (or databases). Riikonen (2009) used TCP fingerprints acquired from the data with Transmission Control Protocol (TCP) and Internet Protocol (IP) headers. Likewise, another approach to identify devices is from the User Agent (UA) string, present in the Hypertext Transfer Protocol (HTTP) request header, available from the IP traffic measurements. To implement the identification with TAC, collection of comprehensive lists mapping TAC codes to specific terminal model, and detailing the supported features of specific terminal models is very laborious to obtain, provided such lists are not already available (Kivi, 2006). Similarly for TCP fingerprinting, a database of known fingerprints is required to compare the TCP fingerprints and determine the OS of the device. The process of updating the fingerprints requires time and resources, as well as the access to the latest devices and operating systems (Riikonen, 2009). All the approaches have some limitations of their own, however one good aspect about the UA based identification is the frequent update of its device repositories or databases, by the active community. Readily available Device Description Repository (DDR) with state-of-the-art device information is an added motivation to perform device detection with the use of the UA string.

The focus of the thesis is to study a convenient tool with comprehensive DDR to identify devices from the HTTP UA string. This thesis analyzes IP traffic measurement

data that has been gathered from Finnish mobile operator(s) in the MoMIE[1] project, conducted as a part of the Aalto University research. In addition to selecting and implementing UA based method for mobile device identification, this thesis provides statistical description on mobile Internet usage based on the identified devices.

1.2 Research Questions

The focus of our research is on developing a tool to identify device (and features) based on the HTTP UA. As content adaptation techniques in the web servers have conducted the identification of the devices in real time by analysing the UA string from the HTTP request header, few research have worked on the passive measurements from the mobile networks. This research is interested in implementing a DDR to extract device information and present descriptive statistics on the Finnish mobile internet usage.

MoMIE project has been conducting traffic measurements annually and some other methods for device identification have been found promising. Nonetheless, further development alternatives are to be classified and concrete recommendations provided, in this area of research.

The research questions which guide the course of our research work are:

Q_1: How can device and device features be identified based on HTTP User Agent from mobile Internet traffic traces?

Q_2: How can the identification of mobile devices (and features) aid in profiling the mobile Internet usage in Finland?

[1] This thesis has been conducted in the MoMIE project context. MoMIE (Modeling of Mobile Internet Ecosystem) is a national research project (2010-2012) funded by the Finnish Funding Agency for Technology and Innovation (TEKES), Nokia, Elisa, DNA Finland, Accenture, Sanoma, and Aalto Comnet

1.3 Objectives

In order to answer the above research questions, there is a need for the objectives to be defined. The objectives of our research are:

O_1: Develop a tool to identify device type, model (and features) based on the HTTP request header User Agent field (in MoMIE project's mobile network traffic measurements context)

O_2: Test the tool - Analyze the accuracy of UA based device identification and compare it with an existing tool

O_3: Provide descriptive statistics on the mobile Internet usage in Finland based on the identified devices

1.4 Scope of the Research

This research is structured around determining a method for the identification of mobile devices with the use of HTTP UA string from the network traffic measurement data. In order to have accurate results, the benefits and limitations of the tool used in device identification should be carefully measured. Smura et al. (2009) based the categorization of mobile devices on the physical size of the device, the capability to make 2nd and 3rd Generation (2G/3G) voice calls, and the OS of the device. According to these criteria, there are five types of mobile devices, namely mobile phones, smartphones and Personal Digital Assistants (PDAs), ultra-mobile Personal Computers (PC), laptops and tablet PCs, and other devices such as the Apple iPods and the Sony PSP.

In this thesis, the identification of the devices is limited to the analysis of the HTTP UA string. For the purpose of the analysis, devices are categorized as PC devices, handsets (mobile phones and smartphones), tablets, and other device and not on the basis of the parameters as used in Smura et al. (2009). PC devices include desktops, laptops and netbooks while handsets devices include pocket devices like mobile phones, and

smartphones, tablet includes devices like the Apple iPad, and others include gaming devices such as Xbox and PlayStation.

From the measurements data collection viewpoint, network access technologies are limited to mobile, i.e., cellular network technologies, as the centralized measurement point has been chosen from the 2G/3G core network (Riikonen, 2009). Connectivity to the Internet via other wireless access networks, such as WLAN, is out of the scope of this thesis. Mobile devices are not limited by any means, as long as they are mobile network capable devices. Thus, all devices from basic mobile phones to PCs are in the scope of the thesis.

As the IP measurements data is recorded from the Finnish mobile network operators, the scope is restricted to the Finnish mobile networks and mobile data subscribers. However, no geographical limitations exist, as also the roaming data of the measured operators' home subscribers is transferred via the home core network and the measurement point.

1.5 Research Methods

First, a **study** is conducted to find the available device identification tools and a comparison is performed to determine the most suitable of the tools. The comparison is based, for example, on the availability of device information, feasibility of the Application Programming Interface (API) implementation, and cost of the tool. In addition, a review of previous research in the area of the research subject is conducted.

Second, **the device identification tool is implemented** and the discussion on the accuracy and value of the results obtained is presented. Extraction of UA strings and other related entities from the measurements data is performed before the input is fed to the tool. Furthermore, descriptive statistics on the mobile devices used by the operators' user base is presented added with the results related to the characteristic features of the devices.

Last, the **main findings** of the thesis and **summary of the research** are presented. Along with this, the implications of the results are pointed out, the reliability and validity of the research are discussed, and suggestions for further research are provided.

1.6 Structure of the Thesis

The thesis consists of five chapters. The structure is illustrated in Figure 1.

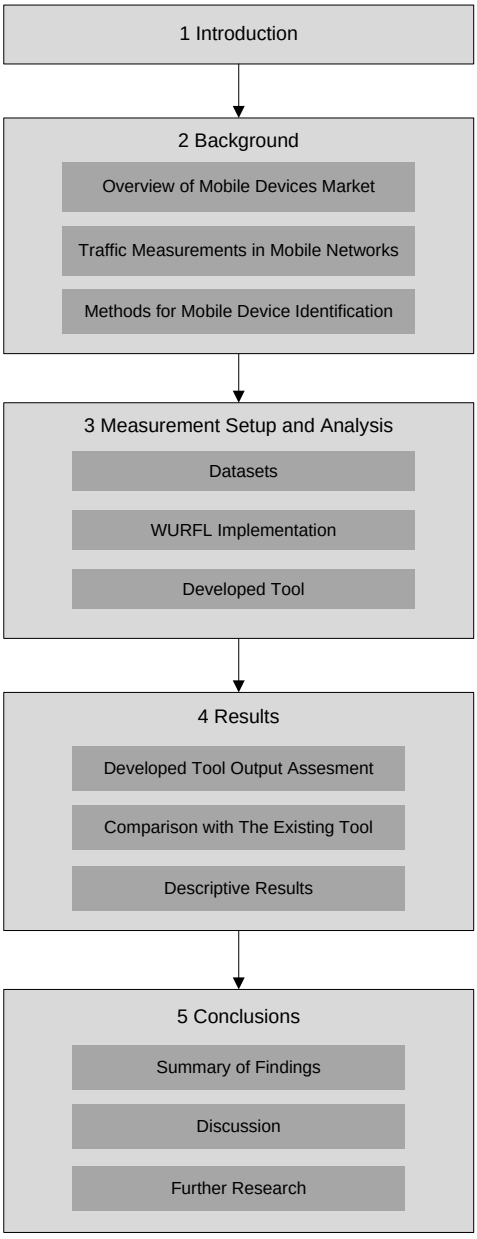

Figure 1-1: Structure of the thesis

After introduction, **Chapter 2** provides the background for the thesis. Overview of the mobile devices market is presented. General information about the measurements points and interfaces in the GPRS/UMTS networks, followed by the methods to identify devices from mobile network traffic measurements are presented and finally the HTTP UA based device identification is discussed.

Chapter 3 describes the measurement data and the analysis process used in the thesis. Measurement setup and data collection for the research and detailed description about the analysis process used in the thesis to perform device identification is provided.

Chapter 4 provides the results obtained from the device identification tool. Discussion on the results and accuracy of the tool is presented. General traffic characteristics are outlined, followed by the descriptive statistics about the devices in context of the mobile Internet usage in Finland. The results include information about shares of different device types as well as detailed description on the device brand, OS, model, and features.

Chapter 5 concludes the thesis by providing a summary of the main findings. In addition, implications of the results are pointed out, reliability and validity of the results are discussed and further research suggestions are given.

2 Background

This chapter provides the necessary background for our research. First, an overview of the mobile devices market is presented. Second, a brief introduction on mobile network traffic measurements is given. Third, methods to identify devices from the mobile network measurements data are discussed and finally an overview on the available methods and tools to detect devices specifically with the use of the UA string is presented.

2.1 Overview of the Mobile Devices Market

Understanding of the mobile devices market is an important step in the analysis of the mobile devices, applications and services. In this section, we will present the criteria for the classification of mobile devices used in previous research works, before giving statistics related to the mobile device market.

2.1.1 Categorization of Mobile Devices Market

In a past study, Omari et al. (2009) classified the mobile device market into three main segments; mobile phones, smartphones, and PDAs. They further mention that the different types of mobile devices have very close levels of capabilities and they offer rather similar services. For this reason, the distinction between mobile devices is blurring. In contrast, Georgieva & Georgiev (2007) mentioned that mobile devices differ vastly from each other by their hardware and software capabilities. However, both of the authors tend to categorize mobile market into segments summarized in Table 2-1.

Table 2-1: Types of mobile devices - summarized from Georgieva & Georgiev (2007) and Omari et al. (2009)

Mobile device	Operating system	Features
Laptops/Notebooks	Windows/Linux	- Visual capabilities: Screen resolution, Screen Mode and Supported Multimedia file formats - Internet access capabilities: Supported markup and script languages
PDAs	Windows Mobile/Palm OS/ Linux/Blackberry	
Smartphones	Windows Mobile/ Symbian/Apple iOS	
Mobile/Cell phones	real-time operating system (RTOS)	

Mobile devices have also been classified based on the criteria such as mobile technology generation (2G versus 3G) or on the property such as the capability to install third party applications to the device (Sugai, 2007). Smura et al. (2009) categorized mobile devices based on the physical size of the device, the capability to make 2G/3G voice calls, and the operating system of the device. According to these criteria, they classified mobile devices into five categories, namely mobile phones, smartphones and PDAs, ultra-mobile PCs, laptops and tablet PCs, and other devices such as Sony PSP and Apple iPod.

2.1.2 Mobile Devices Market Statistics

Mobile device in the form of a cellular phone dates back to early 1980s with the introduction of Motorola DynaTAC 8000X, the first commercial cellular portable phone (Kilpatrick et al., 2006). Since then, with the advancement of the commercial networks from 1G to 3.5G and 4G, mobile devices have evolved from simple phones to devices with capabilities more than just the possibility to make voice calls.

According to Gartner's (2012a), worldwide market shares for the handsets (mobile phones and smartphones) in 2011, Nokia (23.8%) was the market leader, followed by Samsung (17.7%), Apple (5%), LG Electronics (4.9%), ZTE (3.2%), and other brands (45.4%). These shares, when compared to 2010, show decrease in Nokia shares by 5% points whereas among others only Apple shows increase of about 2% points. Considering the market shares in 2011 by OS, Android leads the market with 50.9%, followed by iOS (23.8%), Symbian (11.7%) and RIM (8.8%). As compared to the market shares by OS in 2010, only Android and iOS market shares were found to increase by a heavy margin (around 20% point for Android and 8% point for the iOS), whereas the Symbian market shares decrease by 20%. From the OS shares, it is seen that Samsung remained the main contributor to Android's gain in the market share. In the handset market, Nokia is still the largest supplier in the world but it has suffered a decline in the overall market share (Budde, 2012). Statistics show that Samsung is the second largest manufacturer and it is pursuing Nokia rapidly. Along with the handsets, touchscreen tablet PCs have become very popular, with the most widely known being

the Apple iPad. In the tablet section of the mobile device worldwide sales, iPad is the market leader followed by the emerging Android tablet PCs (Gartner, 2012b).

The Finnish market which is analyzed in the thesis is roughly similar to the global market shares of mobile devices. However, it differs in terms of the OS shares among the mobile handsets. A MoMIE project report by Riikonen (2012) on the mobile handset population in Finland shows the longitudinal data on mobile devices. For 2011, it reports that mobile handsets account for 79% of the total mobile devices, while 14% consists of data terminals such as USB modems, data cards and tablets. The report also shows the growing share of data terminal devices while the share of mobile handsets decreased from 2007 to 2011. Nevertheless, decreasing share of mobile handsets does not imply the decrease in absolute number of handsets. This is because, the size of the device population has been growing and at the same time the shares of different device types have changed. In handset population, Nokia is seen to be the dominating brand with 81% while Samsung is second with 10% following by Sony Ericsson with 2% of the total population. The report does not provide full data on the Apple devices. Shares of mobile handsets by OS show Nokia Series 40[2] at the top with 45% (share in decreasing order from 2005 to 2011) and Symbian OS second with 26% shares. Again, the report does not provide full data on the iOS. According to the report, the advancement of Android OS in Finnish market was not seen until 2010. For more detailed statistics refer to the project report by Riikonen (2012).

[2] Series 40 is a software platform and application user interface (UI) software on Nokia's broad range of mid-tier feature phones

2.2 Traffic Measurements in Mobile Networks

Traffic measurement from the mobile networks is one of the fundamental methods for measuring Internet usage by the mobile subscribers. Other methods include data collection from the operator reporting systems and device-based usage measurements (Kivi, 2006). Packet data traffic measurements are typically passive measurements which can be conducted at several interfaces inside or at the edge of a mobile access network. Measurements can also be conducted at various intermediary nodes between terminals and servers (routers or gateways) where the traffic converges. These measurements are primarily used for characterization of user base and network usage, as well as modeling of traffic generated by the end user devices Kalden (2004).

This section provides an overview of the most commonly used mobile network measurement points and interfaces, and the type of information available from them. In the packet core network like General Packet Radio Service/Universal Mobile Telecommunication System (GPRS/UMTS), interfaces carry either control information alone or traffic data along with the control information. Common measurement interfaces such as Gn, Gi, and IuPS/Gb in the GPRS/UMTS network are depicted in Figure 2-1.

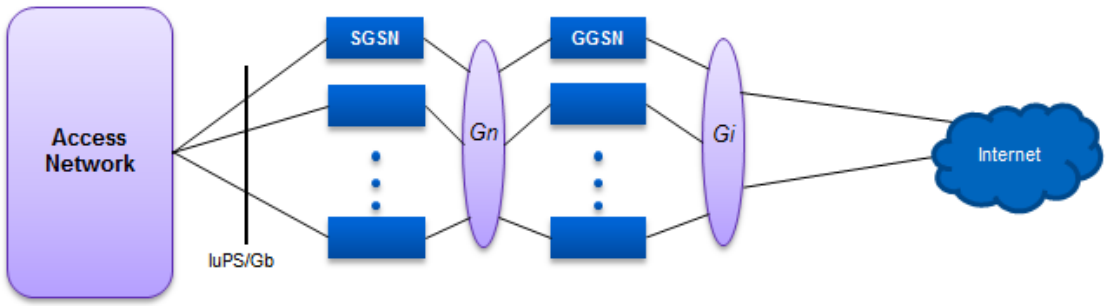

Figure 2-1: Commonly used measurement interfaces inside a GPRS/UMTS network

The interfaces that carry the packet switched data inside the cellular core are the Gn and Gi. The Gn interface is used to connect the network elements of the cellular core. Traffic in the Gn interface is GPRS Tunneling Protocol (GTP) traffic, including both control plane and encapsulated user plane IP traffic. The Gi interface is a gateway

interface to external networks (and the Internet). Measuring traffic at an access point (such as the Gateway GPRS Support Node, GGSN) to an external packet data network (such as the Internet) captures the IP traffic of all the subscribers of the operator. From the user traffic in the Gi interface, it is possible to identify mobile devices by using different techniques. The interfaces between the access network and the Serving GPRS Support Node (SGSN) are IuPS interface in UMTS and Gb in GPRS which carries traffic and signaling that can provide subscriber location and session information. For more information on traffic measurement points, tools and extractable information from different interfaces, refer Riikonen (2009).

2.3 Methods to Identify Mobile Devices

Different types of devices are generating traffic in the mobile network. Devices like handsets, tablets and laptops generate different volumes and profiles of traffic. To be able to understand the usage of mobile services, device type and model level identification of these devices is required. There are a few solutions to identify, for instance, the type, model, and the OS of mobile devices from different interfaces of the mobile network. We will explain and compare different types of device identification techniques from the mobile network measurements.

2.3.1 Type Allocation Codes (TAC)

Every mobile device is uniquely identified by its International Mobile station Equipment Identity (IMEI) number. IMEI number has is a 15 digit code consisting of an eight-digit Type Allocation Code (TAC) issued to the equipment manufacturers. In addition to the TAC digits, IMEI also includes an individual six-digit Serial Number (SNR) uniquely identifying each device within the TAC, and a spare digit, as shown in Figure 2-2. The GSM Association (GSMA) is responsible for the allocation of IMEI code (or TAC) ranges to equipment manufacturers (Kivi, 2006). TAC numbers are allocated on a per-company and per-model basis. Besides IMEI, there is also an IMEI Software Version number (IMEISV), which consists of a two-digit Software Version Number (SVN) instead of one spare digit like in IMEI. IMEISV is allocated by the manufacturer which identifies the software version number of the mobile equipment.

The first two digits of the TAC is the Report Body Identifier code which indicates the GSMA-approved organization that registered (or, before 2002, approved) a given mobile device. For instance, code *35* indicates BABT (British Approvals Board of Telecommunications). The following six-digits of the TAC are under the control of the Reporting Body Identifier which uniquely identifies each mobile device model. (GSMA, 2010)

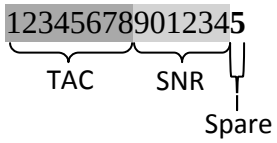

Figure 2-2: Structure of IMEI

In context of the measurements in mobile networks, the data related to terminal models and terminal radio technologies can be acquired from the time-stamped CDRs in GSM/UMTS networks. The CDR registers the mobile terminals that are using the chargeable services, such as packet data Internet service, enabling a way to identify the devices (Kivi, 2007). One way of identifying devices with TAC code is by collecting the data directly from the CDRs. In previous research, Kivi (2007) took into account each terminal creating a certain type of CDR during the measurement period. The TAC codes obtained from the CDR were mapped to appropriate device models using either listings published by BABT or data given by terminal manufacturers on their web sites. Implementation of TAC codes to identify devices can also be seen in the work done by Shafiq et al. (2011). Besides using the CDRs, the TAC codes can be collected also directly with network traffic measurements at specific interfaces in the packet core network, such as the Gn interface. No specific measurement setup is required to collect the TAC codes if the operator's reporting systems are used. Otherwise, if network traffic measurements are used, then the Gn interface should be measured to record IMEIs or TACs from the network traffic.

A database mapping TAC codes to specific terminal models is required to identify a device from the TAC code. This kind of database with the comprehensive mapping of the TAC codes and detailing the supported features of specific terminal models is very

cumbersome to collect. Provided that these databases are not updated timely, this could result in the lack of information for most recent devices. In addition to this, the TAC code ranges for a large terminal manufacturer are allocated for different assembly places and manufacturing facilities separately, which could also create added ambiguity to map the TAC codes (Kivi, 2006).

2.3.2 TCP Fingerprinting

TCP Fingerprinting, also known as TCP/IP stack fingerprinting or OS fingerprinting, is a process of determining the operating system of a remote host by analyzing the packets from that host. Every operating system's TCP/IP stack has its own stack-idiosyncrasies that differentiate them from each other. The working principle behind TCP fingerprinting with the traffic measurement traces is to compare certain IP and TCP headers to previously known signatures of different operating systems.

There are two types of TCP fingerprinting tools; active and passive. Active tools such as *nmap* (Yarochkin, 1998) and *Xprobe2++* (Yarochkin, 2009) send TCP packets to a port and notice how the TCP stack responds. Whereas, the passive techniques (e.g. p0f, Siphon, Ettercap) are based on the collection and analysis of the packet traces transferred in the network. The following steps are necessary to perform passive TCP fingerprinting:

1. A database with identified OS fingerprints has to be built to identify OSs (identification accuracy depends on the scope of the fingerprint database)
2. Traces of the packets sent by the remote system need to be captured and compared with the fingerprints stored in the database by inspecting TCP fields that include, Time to Live (TTL), Windows Size, Don't Fragment bit (DF), and Type of Service (ToS) (Smith & Grundl, 2002).

Previously, Riikonen (2009) used p0f tool for TCP fingerprinting to determine the OS of end user devices. The fingerprinting database used in his research, had been updated annually in the MoMI [3] project to include the latest fingerprints of the devices in Finland. While some level of OS identification is enjoyed by the use of fingerprinting, it comes with a few limitations as well. The main limitations pointed out by previous research are:

a. TCP/IP stack settings could be changed to either avoid identification or appear as some other OS. However, changing the settings is not by any means presumable by an average end user, as it requires certain knowledge on how to perform it (Riikonen, 2009).
b. Remote proxy firewalls rebuild TCP connections for the clients which prevent the identification of the real OS of the user (Smith & Grundl, 2002)
c. There could be bias in the accuracy of OS identification in the mobile network if the identification is only based on the uplink TCP traffic which corresponds to fewer amounts of flows and bytes out of the total traffic (Kivi, 2006).
d. The process of updating the fingerprints requires time and resources, as well as an access to the latest devices and operating systems (Riikonen, 2009).

[3] MoMI was an academic project of the then Helsinki University of Technology, Finland. It took a techno-economic standpoint on the evolution of the mobile service market. The work continues as MoMIE project during 2011- 2012. URL: http://www.netlab.tkk.fi/tutkimus/momi/

2.3.3 HTTP Header Based Approaches

Hypertext Transfer Protocol (HTTP) is an application-level request/response protocol for hypermedia information systems (RFC, 2616). HTTP traffic is based on a client-server computing model where a client, usually a mobile device's web browser, initiates (HTTP) requests that the web server responds to, offering tailored resources for that client (or device). Following is the header of a real request from a mobile handset Nokia 5300 to a server (Firtman, 2010).

GET / HTTP/1.1
Host: mobilexweb.com
Accept: application/vnd.wap.wmlscriptc, text/vnd.wap.wml,
*application/vnd.wap.xhtml+xml, application/xhtml+xml, text/html, multipart/mixed, */**
Accept-Charset: ISO-8859-1, US-ASCII, UTF-8; Q=0.8, ISO-10646-UCS-2; Q=0.6
Accept-Language: en
DRM-Version: 2.0
Cookie2: $Version="1"
Accept-Encoding: gzip, deflate
User-Agent: Nokia5300/2.0 (03.50) Profile/MIDP-2.0 Configuration/CLDC-1.1
x-wap-profile: "http://nds1.nds.nokia.com/uaprof/N5300r100.xml"

The request shown above contains User Agent and User Agent Profile (UAProf) information along with other attributes defined by the mobile device (browser). UA string and UAProf are the two potential attributes from which the server can acquire information about the device originating the request.

UAProf is a voluntary standard defined by the Open Mobile Alliance (OMA, formerly WAP Forum) that lists the abilities of the device, including its screen size, download features, and markup support. It is in the form of an Extensible Markup Language (XML) file. The XML file is defined by the vendor of the device such as Nokia and Samsung, or the carrier such as Vodafone, and the URL link to the XML file is included in the header, typically as *x-wap-profile*. As in the example of the request header stated previously, Nokia N5300 sends an x-wap-profile that tells the web server where to find the profile file, which is in the form; http://nds1.nds.nokia.com/uaprof/N5300r100.xml (Firtman, 2010). More information on UAProf specifications can be found in OMA (2006).

While UAProf provides a good amount of information about the device, for example the screen size, multimedia capabilities, detailed information for video, streaming, and MMS capabilities, drawbacks to relying solely on UAProf should also be considered. Some difficulties with adhereing to the UAProf as mentioned in literatures, Glover and Davies (2005) and (Firtman, 2010), are:

- Not all devices are supplied with a UAProf
- Lack of central repository; local repositories are often out of date
- Schema and data errors can cause the parsing to fail
- UAProf lacks the right granularity of information; for example, it can be read that Flash is supported, but no information about the version is included

The User Agent string from the HTTP request header is another attribute that provides information on the browser or the application software (app) being used, the underlying operating system, and in most cases, the client device model. Early Netscape products generated User Agent strings such as: *Mozilla/4.04 (X11; I; SunOS 5.4 sun4m).* These UA strings provided very little information and were unsuited to precise content personalization (Hoh et al., 2003). A possible solution to this was to extend the request header to accommodate additional client-capability information. RFC 2616 provides the details on how HTTP REQ header can carry sufficient client capabilities information.

One use of UA string is, for instance, in real-time where mobile handsets can be identified, from their user agent field in the HTTP request header, to be redirected to a mobile optimized web page with the equivalent content. On the other hand, from the mobile operators' viewpoint, UA string is useful for passive statistical analysis performed to identify the mobile devices from network measurements. The identification of the devices helps to gain an insight on the current mobile Internet usage among the operators' user base.

The UA strings carry a lot of information regarding the model, the OS and the browser of the requesting device. Typical examples (formats) of the UA string are:

i. *Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0; NP07)*

OR

ii. *Mozilla/5.0 (iPhone; U; CPU iPhone OS 4_3_3 like Mac OS X; fi-fi) AppleWebKit/528.18 (KHTML, like Gecko) Version/4.0 Mobile/7A341 Safari/528.16*

OR

iii. *NokiaN70-1/3.0546.2.3 Series60/2.8 Profile/MIDP-2.0 Configuration/CLDC-1.1*

UA string from #i gives us the information that the device has Windows OS and the browser is Internet Explorer 9.0. Similarly, from the UA string in #ii, it can be inferred that the device is an Apple iPhone with the OS version 4.3.3. Likewise from #iii, device can be identified as a Nokia N70 with the Nokia Series60 (Symbian) OS. Apart from the UA string generated by web browsers, there is a different and rather uncertain format of UA string generated by mobile applications. For instance, a YouTube application on Android generates a UA string as *Android-YouTube/2 (GT-I9000 GINGERBREAD); gzip*, from which we can infer that the YouTube application is generating the request from a device with the model name as GT-I9000, which has *Android* as the OS, and the OS version is *Gingerbread*. In this way, based on this information we can categorize mobile devices into different groups and in addition, identify the application generating the HTTP request as well.

Unlike the UAProfs, there are device repositories available that provide a comprehensive list of device capabilities. Device repositories are offline databases (or online web services) that take a UA string (or all of the request headers) and return the capabilities of the detected device, from screen size, device OS, OS version, to software platform for running applications based on Java programming language compatibility, to Ajax support and video codec compatibility (Firtman, 2010). In addition to the device repositories, APIs are available to interact with the repository and extract required information. The device description repositories are one solution to the difficulties imposed by the UAProf.

2.4 HTTP User Agent Based Device Identification

The User Agent field from the HTTP request header has been utilized to perform device identification by previous researches. For instance, Maier et al. (2010) identified the mobile handheld devices (MHDs) to study the HTTP usage by European DSL broadband users in their home Wi-Fi network. A similar study was conducted by Gember (2011), to present a comparative study of traffic for handheld and non-handheld devices in a campus Wi-Fi network. Gember (2011) used the UA strings in HTTP packets as the primary method to differentiate between those two devices types. They use a keyword list which is based on common knowledge, and published lists to identify the devices within the handheld group. Likewise, Erman et al. (2011) used the UA field to identify non-traditional sources of HTTP traffic like video game consoles, TV sets and smartphones among the residential Internet subscribers in the US.

As mentioned earlier, UA based device identification also requires a DDR, preferably the one which lists a comprehensive database of mobile devices and their features. A tool to interact with the DDR by parsing the UA string and extracting required device information is also required. There are a few DDRs available for this purpose and among them; four of the popular ones are outlined in the following section. But first, a general outline on how UA based device identification is conducted, is presented.

2.4.1 User Agent Based Device Identification - Basic Principle

Basic principle regarding the UA based device identification involves the exact string matching. Exact string matching includes searching a database, with the list of UA strings and the corresponding device, to find a match for the available UA string. The other approach can be, searching the UA string to find a device match from the list of devices.

2.4.2 Device Description Repositories

Table 2-2 presents a comparison between the four DDRs, namely WURFL (Wireless Universal Resource File), DeviceAtlas, DetectRight and Volantis. They are compared

based on the attributes such as the sources of device data, device coverage, cost and licensing options, to name a few important ones.

Table 2-2: Comparison of the four DDRs

Attributes	WURFL	DeviceAtlas	DetectRight	Volantis
1. Sources of device data	Solely by community	UAProf, device manufacturers, network operators, and DeviceAtlas community (dotMobi , 2011a)	UAProf, customer supplied data, own research	UAProf, device manufacturers, own research, community (W3C, 2006)
2. Device and capabilities coverage	- 15438 devices as of *wurfl v2.3* - 600+ capabilities (ScientiaMobile, 2012)	- 7,918 devices(dotMobi , 2011b) - 500+ capabilities (dotMobi , 2011c)	-21355 and 200+ -- PDA/GPS devices - 400+ capabilities (DetectRight, 2011a and DetectRightAMF, 2011a)	- 8,500 devices - 850+ capabilities (Antenna, 2011a)
3. For no exact UA match	Device Inheritance (Fall back mechanism) used	No fall-backs; Doesn't return data if unrecognized	Heuristics and module analysis used (Mobile Phone Wizards AS, 2007)	Not Found
4. Server Resources	Standalone or database backend	Standalone or cloud based options	Standalone or Cloud based	Standalone (Business Wire, (2006) or AMP server (Antenna, 2011b)
5. Cost	- Open Source - Commercial	- Enterprise: Contact and negotiation by email (dotMobi, 2011d) - Cloud and Premium: Standard ($399/year) (dotMobi, 2011e)	- Standard: €99 per month for on-demand updates - Cloud: From 399 Euros per month (DetectRight, 2011b)	- Limited: Free - Professional: $20k or $50k per annum VolantisSystems, 2009)
6. License	- Repository license: dual licensing AGPL and commercial - API license: dual licensing AGPL and commercial	- Repository license: commercial License - API license: commercial license	- Repository license: commercial License - API license: commercial license	- Repository license: commercial License - API license: commercial license
7. Updates	- Infrequent - Relies on voluntary help and manual patches	- Cloud and Standard: Weekly dotMobi (2011e) - Premium: Daily	Updates information not found.	- Live update over the Internet. every few days (W3C, 2006) - Time to time updates (Galoppini, 2008)
8. Accuracy	Not found	Self-reported: Accuracy rates in around 99 percent (Telecomlead, 2011)	Self-reported: near 100% accuracy (DetectRight, 2011b) - recognition rate of 97.3% from a	Not Found

			Russian research (DetectRight, 2007)	
9. Available APIs	.NET, PHP, Java (WURFL, 2011)	PHP, .NET, Java, Python, C++ (dotMobi, 2011f)	.NET, PHP, Java (DetectRightAMF, 2011b)	REST, PHP, Ruby (Antenna, 2011c)

Some other device description repositories are available, for example, from the OpenDDR (OpenDDR LLC, 2011-2012), 51Degrees (51Degrees.mobi Limited, 2010-2012) and MobileAware (MobileAware, 2010).

The data for the repository attributes shown in the Table 2-2 include data acquired from the respective websites of the DDRs, i.e. are self-reported. The figures shown in the table are solely based on the data available openly on the Internet. Hence, the table does not conclude on the superiority of one DDR to the other.

The open sourceness and the cost aspect of the WURFL repository is considered as the main driving force to use it. There is a vast amount of information provided by WURFL creators for the device repository, the API, the algorithms used in device identification and their implementation. However, in our opinion, criteria for the selection of appropriate DDRs should be selected based on comprehensive and up-to-date database of device and features information, widespread adoption of the tool, and availability of the APIs to interact with the database. Having said these, a short introduction to WURFL is now presented.

2.4.3 Wireless Universal Resource File - WURFL

ScientiaMobile (2012b) describes the WURFL DDR as a flat list of (device) elements. Fling (2009) describes WURFL as an XML configuration file which contains information about the capabilities and characteristics of most mobile devices. While WURFL as an open source entity, updates WURFL data every day on the WURFL DB, a publicly available *snapshot* of the DDR is produced and made available about once a month on the WURFL website (Firtman, 2010). The device information in WURFL is contributed by the active community along with the data mapped from the UAProf device profiles schema.

Architecture. Devices are grouped into a hierarchy of devices and attributes. Some devices are equivalent to other devices from the same series, possibly with some new features. For this reason, there is a fallback mechanism in WURFL allowing a device to extend its features based on the features of another device (Firtman, 2010). WURFL assigns unique device ID to every device model included in the WURFL repository. Moreover, each OS version of the device model is given a different device ID so the information is not repeated in the two records.

Patch file. A patch file stores modified or enhanced groups and capability lists for new or existing WURFL devices. Whenever making changes to the WURFL XML file is required, changing the original WURFL XML would be impractical. For this reason, the patch file is built with similar syntax to the main WURFL XML file which creates ease for building patch file. The WURFL API will merge the patch information with the information in the WURFL database whenever a reference to the patch file is made.

Capabilities. Every feature, ability, property, or attribute associated with the mobile device included in the WURFL repository is called a *capability*. Currently, WURFL has 32 device capabilitiy groups and more than 600 capabilities.

Algorithm. WURFL API uses Two-Step UA String Analysis algorithm. The algorithm has been described by the creator of WURFL, Passani (2007-2010) as:

1. The type of UA string is identified to categorize the UA string into Mozilla based or an iPhone based or a Windows based UA string, and subsequently,
2. The appropriate UA "handler" is selected from a pool of handlers, which is dedicated to the string belonging to a given family of UA strings.

After the handler is selected, improved RIS (Reduction In String) or LD (Levensthein Distance) algorithms is applied to identify the mobile device. The working principle of both the algorithms is discussed in detail by Passani (2007-2010). RIS is best applied to the UA string with the unique name of the device in the first part of the string. For instance, Nokia 6600 UA string *Nokia6600/1.0 SymbianOS/7.0s Series60/2.0 Profile/MIDP-2.0 Configuration/CLDC-1.0*. Whereas, LD is best applied to the UA

string starting with *Mozilla/* as in *Mozilla/5.0 (iPhone; U; CPU like Mac OS X;* ***en****) AppleWebKit/420.1 (KHTML, like Gecko) Version/3.0 Mobile/3B48b Safari/419.3*.

3 Measurement Setup and Analysis

This chapter describes the measurement data and the analysis process used in the thesis. The measurement setup for the data collection, the recorded dataset, and data processing tools along with a detailed description about the enhanced WURFL tool for device identification are provided.

The measurement setup is a continuation of the MoMIE project network measurements conducted by Aalto University in Finland since 2005. The measurement setup also includes recommendations made by Riikonen (2009). The MoMIE measurement setup for 2011 and 2012 can be seen in Figure 3-1.

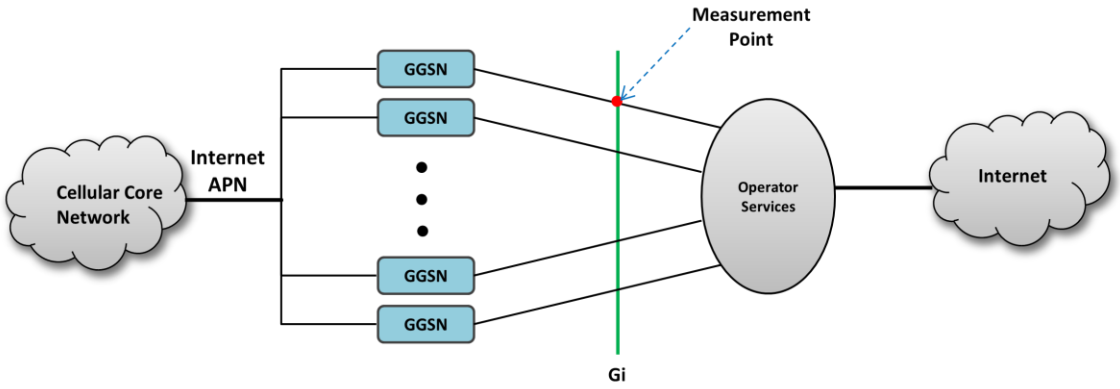

Figure 3-1: MoMIE network measurement setup for 2011-2012. (Adopted from Kivi & Riikonen, 2009)

The measurements were carried out in the GPRS/UMTS packet core networks of two Finnish mobile network operators (MNOs). Since these are major MNOs in the country, analysis could be made upon a significant amount of subscriber base. Traffic data for a period of one week was collected from both of the operators. Measurement of the IP traffic was performed on the Gi interface of the MNOs' networks, which is a gateway interface to external networks and the Internet and carries user plane IP traffic. IP traffic going through one of the GGSNs of each MNO's network was recorded.

Because we captured IP packet traces from a single GGSN only, the data does not directly cover all the traffic of the operator. However, assuming the traffic to be distributed equally among all GGSNs, the data acquired from a single GGSN was

scaled to represent the operators' total traffic. Furthermore, if the Finnish subscriber generated traffic profile is assumed to be similar between the operators (Riikonen, 2009), the measurements of the two MNOs can be assumed to represent the Finnish mobile Internet usage well. The measurement setup involved conducting passive measurements which did not obstruct the network. Measurement setup used a proprietary tool developed in Aalto and a modified version of the Tstat tool to capture and analyze the network traffic. Sensitivity of the data was also taken into consideration by masking user related fields and recording only the most frequently observed UAs based specified threshold value.

An outline of the analysis process is given in Figure 3-2. Input data are the WURFL repository and the list of HTTP UA strings. The first step of the analysis involves parsing of the UA strings through the WURFL API to obtain the output of WURFL alone. The output of this WURFL implementation is then enhanced with the application of custom WURFL patch and custom rules before integrating the String Matching results and applying manual updates. Selected features are mapped to the identified devices from the in-house handset feature list (mentioned as MoMIE Handset Feature List), as the final step of the analysis. The final output (arranged in separate columns) contains the UA strings, identified (including unknowns) devices and device features, corresponding traffic volumes (in bytes) and flows, along with the String Matching results. Detailed descriptions about the datasets and various phases of the analysis process are presented in the following sections.

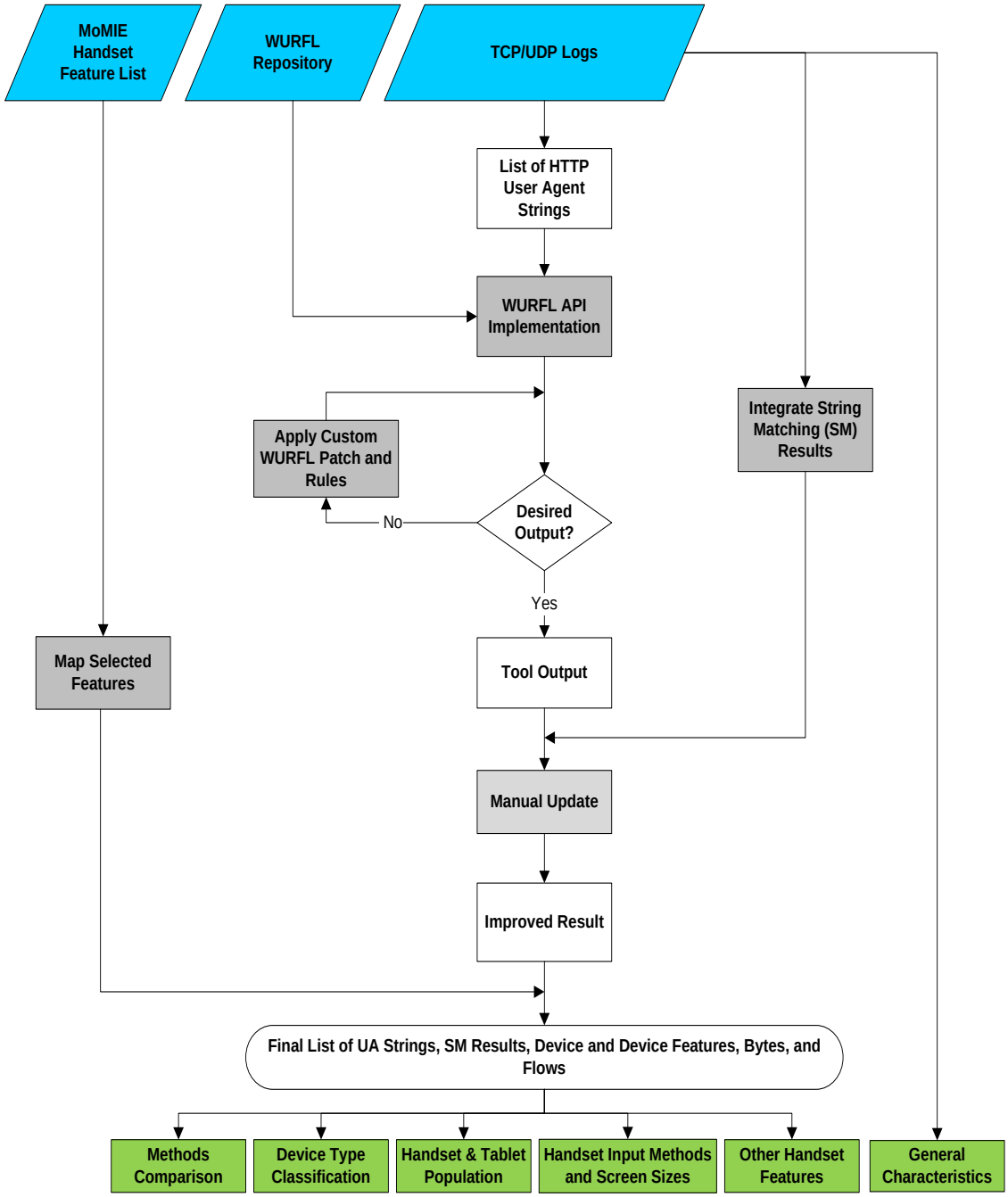

Figure 3-2: The analysis process

3.1 The Datasets

As depicted in Figure 3-2, three datasets were used for the analysis purpose, TCP and UDP logs, WURFL DDR, and mobile handset feature list maintained in the MoMIE project.

3.1.1 TCP and UDP Logs

Out of different types of traces collected in the measurement, UA strings from the HTTP header traces were used for the purpose of device identification in this thesis. The data was acquired from output of the *Tstat* tool specifically modified to provide HTTP header from the TCP log files recorded in the measurement. Tstat creates a set of text files where each row corresponds to a different flow and each column is associated to a specific measure. The columns can be grouped according to *C2S* - Client-to-Server and *S2C* - Server-to-Client traffic directions as required (Tstat, 2008). The columns represent measures such as client and server IP addresses, flow duration, start and end time, request and reply ports, transferred bytes, TCP connection type, as well as the UA string. Out of the different log files that the Tstat generates, we used the TCP related *log_tcp_complete* files. This file contains records of the complete TCP connections, which are connections which fully satisfy the three ways TCP handshake (RFC 793). Incomplete TCP connections were not considered in the analysis.

There are *log_tcp_complete* files for each hour of the day, for the whole measurement period. This allows us to determine the diurnal pattern for HTTP traffic with aggregated traffic volumes for each hour and day which is obtained by aggregating traffic volumes for the Connection type HTTP. The bitmask value stating HTTP protocol Connection type is ‘1’ (Tstat, 2008) in the *log_tcp_complete* files. Similarly, diurnal pattern for the total TCP traffic can also obtained by aggregating traffic volumes on *log_tcp_complete* files for the all the hours within the measurement period.

The TCP log files also included results from a benchmark tool, a tool that is being used to identify mobile devices in research within the MoMIE project. This tool operates in real time and identifies the mobile device model, device type and application from the IP packets being recorded from the operator's network. It is based on simple string matching algorithm. First the model name of the device is identified, and based on that, the type of device is determined. The type of device is categorized either as a *Handset*, T*ablet*, *PC*, or *Other* device. The database containing the devices, which is searched to find a match, has been prepared by listing previously identified device brand, model and OS names. From this point onwards whenever the term *String Matching* is used, it means this tool.

From the traffic traces and results from the String Matching tool in the *log_tcp_complete* files, only traffic identified as HTTP traffic was considered for further processing and analysis since it contains the UA string. The HTTP traffic traces were aggregated to calculate the sums of transferred bytes and flow counts for each combination of the UA string and the String Matching results. Thus, the processed data is a tab delimited text file with unique columns for UA string, String Matching results, and aggregated traffic volume and flows corresponding to the combination. Also, to generate general traffic characteristics related to TCP and HTTP traffic, the TCP and UDP logs were used. To generate these data for device identification and general traffic characteristics from the TCP and UDP log files, Perl, awk and bash scripts were used.

3.1.2 WURFL Repository

Another dataset used in the analysis process is the WURFL repository. As described in section 2.4.3, WURFL repository is an XML configuration file which contains information about the capabilities and characteristics of most mobile devices. Initially, the WURFL repository version 2.3 and the Java API version 1.3.1.1 (released in November, 2011) were implemented. Later when new versions of both the repository and the API were released (in May, 2012); the tool was updated to include the new releases, namely the WURFL repository version 2.3.1 and the Java API version 1.3.6.

3.1.3 MoMIE Project Handset Feature List

Handset feature list is a list of feature information for different mobile handsets. The features in the list are arranged for each handset model. The model name of the handset is a combination of the handset brand and handset model name. For instance, iPhone is listed as Apple iPhone, which is a combination of brand name Apple and handset model iPhone. The list includes handset features like GPRS, WLAN, Bluetooth, and GPS which were collected manually.

3.2 WURFL API Implementation

The development of the device identification tool by implementing the open source software components provided by WURFL requires the installation of the API. The API allows interaction with the DDR to identify devices by parsing the UA string. After the output was acquired by processing the TCP and UDP logs, WURFL DDR and Java API were implemented. Eclipse[4] IDE (Integrated Development Environment) application was used for programming the interaction between the API and the DDR along with the addition of the external libraries. For this purpose, the WURFL device repository version 2.3 (available in a zipped file) was downloaded from the WURFL website[5] and stored in the local machine. This repository can be accessed by the API either as a zip file or as an XML file zip obtained after extracting the zip file. The API is in the form of a jar (Java Archive) file named *wurfl-<version>.jar*, which is a collection of classes and interfaces that perform the programming tasks. This jar file was added as an external library to the IDE. In addition to that, a javadoc jar named *wurfl-<version>-javadoc.jar* file which was also added to the IDE contains documentation for the main Java code.

[4] http://www.eclipse.org/
[5] http://wurfl.sourceforge.net/

The API uses some kind of internal logging mechanism which required external libraries (jar files) to be added to the IDE. These external libraries added into the IDE were:

- backport-util-concurrent-3.0.jar
- commons-collections-3.2.1.jar
- commons-lang-2.4.jar
- commons-logging-1.1.jar
- servlet-api-2.4.jar
- slf4j-api-1.6.4.jar
- slf4j-nop-1.6.1.jar

After all the required jar files were added to the IDE, WURFL concepts were implemented in java code to parse the UA string through the API and interact with the repository. The programming concepts involved in the interaction are shown by the steps below:

Step1. WURFL Holder was instantiated with the repository information
WURFLHolder <Holder object name >=new CustomWURFLHolder("<repository file path>")

Step2. WURFL Manager was created
WURFLManager <Manager object name> = <Holder object>.getWURFLManager().

Step3. Device object was initialized to parse the user agent string
Device <Device object name> =
<Manager object name>.getDeviceForRequest(<user agent string>)

Step4. Device object was utilized to extract specific capabilities by using *getCapability* method
capability_name= <Device object name>.getCapability("<capability_name>")

For instance, considering a UA string such as *Mozilla/5.0 (iPhone; U; CPU like Mac OS X; en) AppleWebKit/420+ (KHTML, like Gecko) Version/3.0 Mobile/1A543a Safari/419.3*, following codes were written after the Step2:

Device dev = wm.getDeviceForRequest(Mozilla/5.0 (iPhone; U; CPU like Mac OS X; en) AppleWebKit/420+ (KHTML, like Gecko) Version/3.0 Mobile/1A543a Safari/419.3);

model_name=dev.getCapability("model_name");
brand_name=dev.getCapability("brand_name");

The execution of the above codes returned the brand name as Apple and the model name as iPhone. After testing the tool to execute a single UA string, necessary codes were written to enable the tool to parse a list of UA strings from the input file and produce selected device capabilities in a tab delimited output text file. Aforementioned steps were utilized for the purpose of extracting selected device capabilities as per their usefulness to the analysis. The selected device capabilities which were extracted for the output file are provided in Appendix A, Table A-0-1.

Although the WURFL repository is comprehensive in listing UA strings for the majority of mobile devices, there were cases when the device was either unidentified (returned as a generic device) or inaccurately identified. These inaccurate and generic identifications are undesirable. Thus, further improvements were made in terms of the addition of the custom WURFL patch and implementation of the custom rules.

3.3 Improvements to the WURFL Output

Improvements to the WURFL output were categorized into two on the basis of the types of unidentified UA strings. The first category includes addition of unidentified web browser UAs to a custom patch file to be used by the tool. The second category includes creating custom rule to extract device information from app-generated UA strings.

3.3.1 Integrating Custom Patch File

In addition to the device repository provided by WURFL, we created a custom patch file which stored modified and enhanced groups and capability lists for new or existing devices. The custom patch file was based on the cases of false positives, i.e., cases which are identified but show incorrectly identified devices, and unidentified devices. For instance, let us consider a case where the model name of a device was identified

incorrectly or was not identified at all. It was the case when there were some alterations in the standard format of the UA string or if other strings were found to concatenate with the standard UA string. These alterations on the UA string bypassed the WURFL device identification algorithms and the device was left unidentified (brand name was returned as *generic* or *generic web browser* in the output). Other cases may include a scenario where a new version of the device OS information is included in the UA string or the UA string is from a mobile device introduced in the market later than the latest release of the repository. The results regarding this scenario will be discussed more in detail later in the Results chapter.

The custom patch file format is similar to the format in which device capabilities are arranged in the WURFL XML file. In the case of false positives and unidentified UA even if the model is available in WURFL, first, the UA string was observed to determine the device model name. Second, a new device id was created for the UA string by following the same pattern of device id as the existing model in the WURFL XML had. Third, a fall back id for this device id was entered as the device id of the basic version (version1, device factory settings) of the existing WURFL device model. In addition to that, changes in the capabilities were also included if necessary. On the other hand, if the device model observed in the UA string has not yet been included in the WURFL repository, a new device id was created and the fall back id was pointed to the OS version that the model was installed with. All other required capabilities were manually entered for that device. These methods of creating a new entry in the patch file would accurately identify the device and return desired values of the capabilities.

The working principle of the patch files involves instantiating the WURFL Holder with the patch file along with the WURFL repository. When UA strings are parsed, the patch file is also imported to build a modified version of the device repository. In our case the

custom patch file was strictly targeted to map the UA strings generated by mobile web browsers. Multiple patch files can be added on a need basis. More information on how to create a custom patch file and the rules associated with it can be found at the WURFL website[6].

3.3.2 Creating and Implementing Custom Rules

Custom rules were created for the UA strings that were not in the standard format as defined by RFC 2616. It was observed that these UA strings are typically generated by applications other than the mobile browser. As WURFL is primarily created to detect mobile devices and their capabilities for web content optimization in real time, it does not contain repository entries and algorithms to handle the UA strings which are in non-standard format. For this reason, custom rules were created to identify devices from these kinds of UA strings.

For the purpose of creating custom rules in java programming language, app-generated UA strings were grouped together to determine if they followed similar format. UA strings were grouped by popular OS names and different formats of the UA string associated with the OS were determined. Also, application specific format of the UA strings for different OS were grouped together. The grouping of the UA strings does not follow any established guidelines or approaches from previous research. We basically grouped and implemented the codes as per our own inspection and for the ease of making as fewer custom rules as possible.

The code implementation of the custom rules involved manipulating the WURFL device id (devID) returned after initial parsing of the UA string. WURFL generates Device specific ID for all identified UA strings whereas unidentified ones are assigned

[6] http://wurfl.sourceforge.net/patchfile.php

with the device ID which includes the term *generic* in them. Let us consider the UA string *BBC News (AndroidApp; 1.2; 3) HTC Desire HD A9191 (Android 2.3.5, SDK 10)*, WURFL returns device ID as *generic_android_ver2_3*, brand name as *Generic* and model name as *Android*. This information about the UA string would not suffice in identifying the device to an extent required in the thesis. Thus, custom rules were first created by checking if the device ID contains the string *generic* and then required operation on the string were performed using java methods from the *StringUtils* java class. Executing the WURFL tool along with the custom rule returned brand name *HTC*, model name *Desire HD A9191* along with accurate values for device OS, and OS version. Besides the brand and model name of the device, we were also able to extract additional information such as the application name and application version using the custom rule. So, application name as *BBC News* and application version as *1.2* were also extracted from the UA string.

With the integration of the web browser patch (separate or merged with the main repository), WURFL is capable of identifying the devices as desktop (PC) devices based on the web browser UA string. Identification of the web browsers (as desktop devices) is marked by the *is_desktop* capability with value either TRUE or FALSE. It was observed that the brand name and model name for web browser UA strings were returned as the browser name and browser version respectively and the devID was returned with browser name and version concatenated by underscore. For instance, devID, brand name and model name for UA string *Mozilla/5.0 (Windows NT 6.0) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.52 Safari/536.5* were returned as *google_chrome_1.9, chrome and 19* respectively. As there were a number of web browsers and their corresponding UA in the data, we decided to harmonize them all into a common brand name with the term *Desktop*. For this purpose, custom rules were created with a concept to search the UA string for desktop PC related brand names such as Windows NT, Macintosh (MacBook), Linux i686 and some other brands. Thus, these custom rules were able to group the desktop devices to Windows, Mac, and Linux or Unix Desktop.

Along with the desktop web browser UA, rules were made to identify UA generated by the desktop applications other than the web browser. P2P clients, Messaging applications, Email clients were such Desktop applications to name a few. Rules for these applications were created for UA strings in similar format or by creating application specific format rules.

In this way, custom rules were implemented in the enhanced WURFL tool, for most of the non-standard UA strings along with a few modifications to classify the identified desktop devices web browser UA to desired categories. Hereafter, whenever the term *enhanced WURFL tool* is used, it shall mean the complete tool that was developed after the implementing custom WURFL patch and rules to the output of WURFL alone.

3.3.3 Incorporating New Releases

WURFL device repository increased in size as more and more device information is constantly contributed by the active community. WURFL API is also under constant improvements as new versions with improved device identification are released frequently.

In the course of implementing the tool and programming new rules, new versions of the device repository (v2.3.1) and the API (v1.3.6) were released by WURFL. Version 2.3.1 has the web browsers patch integrated in it. Previously, there were two separate xml files; WURFL repository v2.3 and *web_browsers_patch*. As one of the main reasons in implementing WURFL for device identification was because of its frequent update of device information and improved algorithms in the API, we had to make our code compatible to those frequent updates. For this reason, tool was upgraded to include the latest repository v2.3.1 and the new API v1.3.6, just by replacing the old repository and the API jar file references in the code with the new ones.

The custom patch file and the custom rules both were built with reference to the WURFL device ID. Updating the tool to include the newest released WURFL repository and the API induced minor errors in executing the program. Possible errors and their debugging are described below:

1. **Error related to incorrect *fall_back* and *device id* in the patch file**: The device information in the custom patch file can be referring the *fall backs* to unmatched device id. This is because; WURFL is constantly including new devices and re-arranging the old ones in their device repository. Thus, after running the program it might show error like *Orphan exception in hierarchy: [<device_id>]*.

The error was debugged by following the steps as mentioned below:

i. Search the custom patch file for the device_id shown in the error message and check the *fall_back* associated with it.

ii. Search the *fall_back* to the previous repository that you had used to find out the user agent string associated with it.

iii. Search that user agent string in the new repository to find out the corresponding device id

iv. Replace the fall_back id on the patch file with that new device id

If the search for the user agent in #iii returns null then we have to determine the nearest match for the model name and device OS version in the user agent string to update the *fall_back* id associated with it.

2. **Error related to duplicate user agent entry in the patch file**: This error was thrown when the user agent added in the patch file, had then been included in the newer version of the repository. The device information on the custom patch file was removed to debug this error.

3.4 Further Improvements to the Enhanced WURFL tool

The output of the tool improved after integrating a custom patch file and implementing the set of custom rules. However, there were cases when UA string was not recorded due to privacy issues or the UA string was recorded incomplete or the custom rules did not cover certain UA strings. To address these concerns, we combined the results from String Matching to the output of the tool with the output from String Matching. The String Matching results added the improvements to the output of the enhanced WURFL tool which includes the addition of a *Device Type* measure to categorize the devices into PC, Handset, Tablet, Handset/Tablet, and Others, and helps to decrease the share of unknown devices. The share of unknown devices is decreased by integrating String Matching results for the cases when String Matching detected the device which the enhanced WURFL tool could not.

3.5 Mapping Device Features from the Handset Features List

In addition to the features (capabilities) identified by the enhanced WURFL tool, selected features, from the handset features list were added to the output of the tool. Some of the added features were similar to the WURFL capabilities, which also enabled the comparison between these two results. As the features list is created around specific mobile handset model name, the features were mapped with reference to the device model name. Since the model name in the features list is the combination of the mobile device brand name and the device model name, following model name harmonization steps were performed to map the required features:

1. Brand name and model name from the enhanced WURFL tool were concatenated in to a new variable
2. The model name in the new variable was updated manually to match the one in the feature list

The features mapped from the features list are provided in Appendix A. The features were available for handsets only, so features for the device type *Tablet* were updated manually.

4 Results

This chapter presents the results of the analysis. First, the accuracy of the enhanced WURFL tool is discussed, along with its comparison with the String Matching results. Second, general traffic characteristic is described. Last, descriptive statistics on mobile devices and device features are presented. Descriptive statistics are generated from the final results obtained after the integration of the String Matching results to the output of the enhanced WURFL tool followed by few manual modifications as described in section 3.4 and 3.5.

4.1 Accuracy of the Enhanced WURFL Tool

To describe the accuracy of the enhanced WURFL tool, the results presented here are based on the share of the identified devices along with the shares of false positives. It is important to note that the results presented in this section are obtained from the enhanced WURFL tool where the input was a list of known UA strings. The traffic related to the UA strings that were not recorded, due to privacy reasons, is not included here (the share as mentioned in section 3.4).

Table 4-1 shows the shares of identified devices by the WURFL alone and the enhanced WURFL tool (WURFL Implementation + Custom Patch + Custom Rules). It shows that the enhanced WURFL tool was able to identify about 94% of the total UA strings subjected to the analysis whereas, the implementation of WURFL alone, was able to identify 80% of the UA strings. The custom rules and custom patches applied to the output of WURFL implementation improved the result roughly by 14% points.

Table 4-1: Device identification results

Device Identification	Shares		
	UA Strings	Bytes	Flows
WURFL Implementation Alone	80%	88%	90%
Enhanced WURFL Tool	94%	94%	97%

Out of the total number of UA strings for which the enhanced WURFL tool identified the devices, the share of UA strings for handheld devices (such as mobile handsets and tablets) was found to be 27%. This share of handheld UA strings includes the identification of precise device models (such as the HTC Desire S or GT-I9000) as well as devices identified as generic device (such as Generic Android, unknown Apple Handset/Tablet). Devices identified from the UA string as generic device refer to the identification of at least the correct OS or the brand of the mobile device. Roughly 26% of the total handheld device UA strings fall under this category of generic devices. Model name is identified as *generic <OS name>*, for the UA strings (mostly app-generated UA) which contain the app name and the device OS name or just device OS name (and version, in some cases) in them. One typical example of this kind of UA string format is *<app name>/<app version> Android/<OS version>*. Another format includes the Apple device app-generated UAs where, CFNetwork, a library used by the iOS for communication usually adds its name and version number to the end of UA string (Maier, et al., 2010). These types of Apple device UAs were identified as *unknown Apple Handset/Tablet* (again a generic identification). There were also a few cases when only brand name and the OS name were present in the UA string, this type of UA strings were identified as *generic <brand name>*.

The share of identified devices for the output of WURFL alone also includes the false positives. False positives refer to the UA strings that are identified incorrectly. The share of false positives was roughly only 0.5% out of the total number of analyzed UA strings. Some examples of false positives are; MeeGo app-generated UAs (with the format same as that of a browser, starting with *Mozilla/*) which were identified as Desktop devices. Also, UA generated by Internet Explorer mobile (IEMobile) browser, in Windows Phone, was detected as Windows Mobile. Also, there seems to be no explanation for the desktop web browser UA string being identified as Tablets such as Logitech Revue or EE Pad TF101. App-generated UA strings starting with NokiaN9 were detected as Nokia N90. Likewise, UAs from Nokia Suite (Ovi and PC both) updates are identified as Nokia 2605. If the UA string starts with an app name, consisting of a few initial characters that match with a device model in WURFL, then

the UA string is identified as that device model, which is incorrect. For instance, UA string *Touch/x.x.x CFNetwork/xxx.x.x Darwin/xxxx.x.x*, though it is from an Apple product, is identified as model *Touch7* with the brand name as *Emblaze*.

WURFL repository (v2.3.1 used in the analysis) also has a list of *spoof UAs* that the handsets use to alter their UA string. Some handsets allow the user to change the default UA string that is sent to the server which is often used to fetch contents equivalent to the content sent for the web browsers. This type of user altered UAs are known as spoof UAs. Nonetheless, WURFL was able to identify a few of those UA strings, which was only 0.1% of the total UA strings under analysis. This is a rather negligible share and hence do not have much effect on the accuracy as a whole. One example of the altered HTC Sensation UA string is *Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10_6_3; HTC_Sensation_Z710e; en-no) AppleWebKit/533.16 (KHTML, like Gecko) Version/5.0 Safari/533.16*, which is sent to the server as if it is from a desktop device (Safari running on an Intel Mac). These types of UA strings are included in WURFL repository which makes it capable to at least identify the device as HTC Sensation.

4.2 Comparison with the String Matching Results

Comparison of the results from the implementation of the WURFL alone, the enhanced WURFL tool, and the String Matching is provided in Table 4-2. The table shows the shares of unidentified UA along with the share of traffic associated with them. Results indicated the unidentified shares of UA string was roughly 20% for WURFL alone, whereas it was 15% for String Matching and only 6% for the enhanced WURFL tool out of total UA string provided as input.

Table 4-2: Comparison between the tools

Unidentified UA string	Unidentified Shares		
	UA Strings	Bytes	Flows
WURFL alone	19.9%	12.0%	10.4%
String Matching	14.6%	7.3%	4.4%
Enhanced WURFL tool	6.3%	6.4%	3.0%

Considering the unidentified shares of UA strings out of the total, it can be seen that the String Matching results are better than the results from WURFL alone. However, the results were improved by the enhanced WURFL tool as compared to the WURFL alone. Also, notable decrease in the share of unidentified UA strings is observed in the enhanced WURFL tool in comparison to the String Matching. Major contribution to this improvement was made by the identification of the UA strings where only the application and *CFNetwork* information was present. The device was detected as *unknown Apple Handset/Tablet* which at least provided the information that it is Apple iPhone, iPad or iPod. The share for these UA strings was roughly 6% out of the total UA strings. Along with the improvement in the identification shares, we point out a few important advantages of the enhanced WURFL tool over the String Matching.

- The enhanced WURFL tool facilitates manipulation of the output to provide room for added programming and analysis, whereas the String Matching results are obtained in real time and restrict further improvements
- Custom rules provide grouping of the app-generated UAs which provides the identification to become more efficient. For instance, a single rule can work for most of the UA strings following similar pattern
- Enhanced WURFL tool provides information on the device OS and OS version along with an elaborated list of device features, whereas String Matching results are at the moment limited to device model, device type and the application used
- Unlike String Matching which identifies mobile browsers only as *Mozilla* or *Opera* , the enhanced WURFL tool identifies the exact mobile browser
- Contribution of device information by the active community removes the task of manually updating the device information which is rather cumbersome in the case of the String Matching tool

Apart from the shares of unknowns as mentioned in Table 4-2, the tool was able to identify accurate model names for the cases where String Matching identified them only as generic devices based on the device OS. The share of accurate model names identified by the tool in this case was found to be roughly 70% out of the total generic

identification share of String Matching. This also implies that the accuracy of the enhanced WURFL tool is better than the String Matching for those cases.

Results are presented in the report about the false positives regarding the WURFL tool, but it was observed that String Matching has false positives as well. The numbers related to this result were not generated; however it was seen, for example, a Windows Live Messenger application operating on an Apple device which generates UA with the string *CFNetwork* in it, being identified as a Windows desktop device.

4.3 General Traffic Characteristics - HTTP Traffic by Day and Hour

An overview of general TCP traffic characteristics with the focus on the HTTP usage is provided in this section. Diurnal distributions of the total TCP traffic and the traffic for HTTP connection type are presented in Figure 4-1. We concentrate on the distribution of the HTTP traffic.

HTTP traffic shows large amounts of variation during the day. With low traffic volumes in the early hours of the day, which increases steadily as the day progresses and the HTTP traffic peaks in the night between 7pm to 9pm. The result shows high traffic volumes during the night, when most of the people may already be at home. This could be explained by the use of PC devices to access mobile Internet as they generate high volumes of traffic. It is also seen that the traffic remains fairly similar throughout the week. Out of traffic distribution for all the hours of a day, highest traffic is seen between 8pm to 10 pm. and the *Busy Hour* for the traffic is seen to be between 8pm to 9pm.

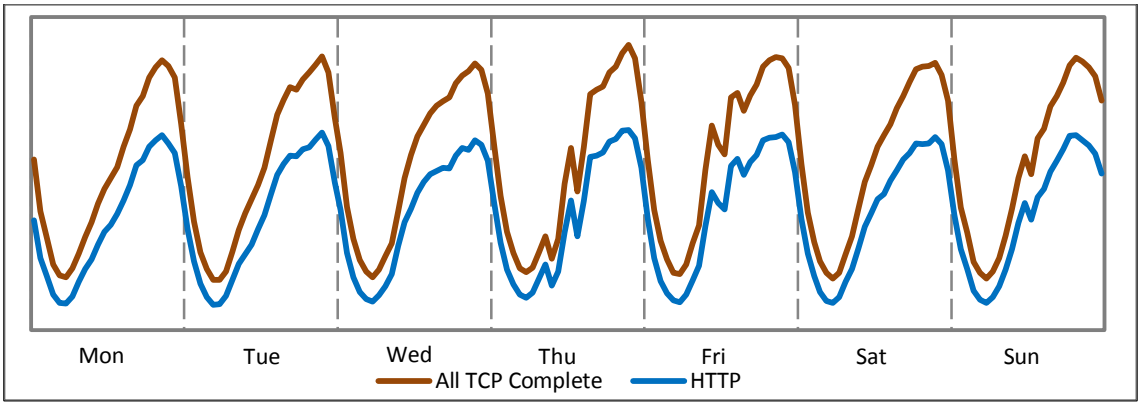

Figure 4-1: Distribution of traffic volume (bytes) by day and hour

A previous research by Riikonen (2009) showed decreasing amount of handset traffic during the weekend, whereas the computer (PC device) usage did not decrease throughout the week. Comparing the traffic distributions presented here with the results from Riikonen (2009), it is possible that the majority of devices generating HTTP traffic could be desktop (PC) devices.

4.4 Classification of Mobile Devices

The devices identified from the analysis give an idea about the type of devices that are generating the HTTP traffic. It was seen that different brands of mobile devices generated different formats of UA for different models of devices. Categorization of the results into different device types is based on the identified devices. Desktop devices are identified based on the web browser and other desktop application generated UAs. Desktop devices with more resource demanding operating systems like Windows OS, Mac OS X, and numerous Linux and UNIX variants are then classified as the device type *PC*. Device type *Handset* includes all identified mobile handset devices without further categorization into smartphones and mobile phones. All identified tablet PCs are categorized as *Tablet* device type. For both Handset and Tablet device type, device is identified from native mobile browser and mobile application generated UA. Handheld devices with no cellular radio capabilities (e.g. iPod) and gaming devices (such as Xbox) with or without 3G capabilities are classified into the *Others* device type category.

Handset and Tablet device type also includes proportionally distributed share of devices that were identified as either Handset or Tablet. Precise device brand name or model name was not included but device OS information was extracted from the UA string associated with this identification. The devices classified under this category were assigned with device type name as *Handset/Tablet*. The traffic associated with this device type was distributed among their corresponding devices under Handset and Tablet device based on their OS (or even brand in some cases) information.

The main focus of the thesis has been to identify mobile devices and present results related to their share of traffic. Figure 4-2 shows the share of traffic for each device type, in terms of transferred bytes and number of flows out of the total transferred bytes and flows. It can be seen that majority of flows are generated by PC devices (about 80%), PC devices also share a significant amount of traffic volume (85%) out of the total. Handset generated bytes is second on the list with 6% shares which is significantly low as compared to PC whereas, Tablet generated shares of traffic volumes are even less, only 2%. However, the share of Handset related flows is seen to be 14% out of the total, which is a considerable amount as compared with the device type other than the PC. This suggests that handsets show frequent usage of the mobile Internet even if the data transferred is less, which is probably due to the mobile optimized contents. The device type category, Others shows only 1% share of bytes and less than 1% out of all incurred flows which is insignificant amount of traffic.

Apart from the identified device types, 4% of the flows were left unidentified which is s about 5% of the total transferred bytes. This share of unidentified devices includes the share of enhanced WURFL tool unidentified devices and the shares of unidentified String Matching results observed specifically for the case when no UA string was recorded. From the figure, it is seen that the share of traffic by volume for the Unknown (those UA left unidentified) is similar to that of the Handset. Now, if we divide the traffic shares of Unknown based on the proportions of identified device types, it can be assumed that roughly 90% of the Unknown is PC generated traffic. This leaves the rest of 10% to the other categories.

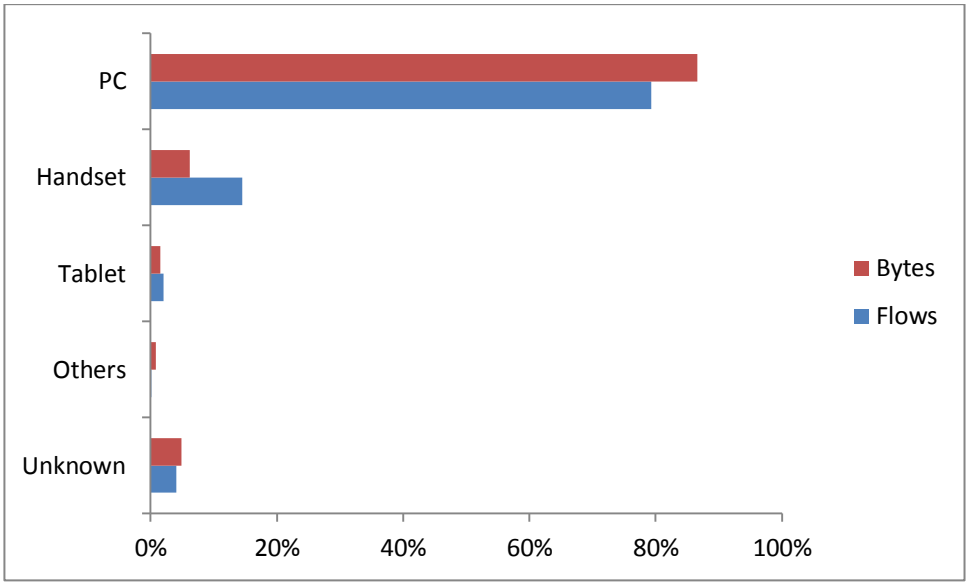

Figure 4-2: Share of all mobile devices generated traffic volume and flows

Based on this assumption and our interest in analyzing handheld devices such as Handsets and Tablets, the traffic shares related to the Unknown will not be considered for further analysis. In addition to this, shares related to the Others and PC device types are also excluded from the analysis. Therefore, from this section onwards, we present results related to the Handset and Tablet device types only.

4.5 Handset and Tablet Population

As Handset and Tablet device types share a notable amount of traffic, this section presents the statistics related to the identified devices that fall under these two device types in more detail. We use the term *Handheld devices* for the combination of the devices under Handset and Tablet device types. The Handheld devices include pocket devices like mobile phones and smartphones (e.g. Samsung Galaxy S II), and Tablets (e.g. Apple iPad).

4.5.1 Handheld Devices Brands

Share of traffic for the top brands of handheld devices is presented in Table 4-3. This data provides an insight on how Finnish mobile device market is evolving after the proliferation of smartphones and tablets.

Table 4-3 shows the share of the top handheld device brands based on the transferred bytes and flows. Out of the handheld devices identified correctly, considering the shares out of the total flows, around 28% of them are Samsung devices generating around 14% of the traffic volume (bytes). Apple with roughly 27% of the devices and Nokia with 23% are the next popular brands. Despite the low difference of roughly 1% between the shares of flows for Samsung and Apple devices, it can be seen that Apple devices generate nearly three times the traffic volume generated by Samsung devices. The table also shows traffic shares for Generic Android, this brand of devices include the generic identification of the devices from the UA strings with only Android OS information contained in them. They account for only 3% of the total flows whereas, these devices are found to generate traffic (nearly 25%) more than the combined shares of Nokia, Samsung, HTC and all others, except the traffic volume generated by Apple branded devices. Most of the devices in the list, Samsung, HTC, ZTE, Huawei and Sony Ericsson, use Android OS which suggests the explosion of Android based handheld devices.

Table 4-3: Share of handheld device brands

Brand	Bytes	Brand	Flows
Apple	37.2%	Samsung	28.4%
Generic Android	24.6%	Apple	27.2%
Nokia	13.9%	Nokia	23.2%
Samsung	13.8%	HTC	11.1%
HTC	7.1%	Generic Android	3.2%
Huawei	0.8%	ZTE	2.5%
ZTE	0.8%	Huawei	1.9%
Sony Ericsson	0.8%	Sony Ericsson	1.2%
Unknown	0.4%	Motorola	0.5%
Motorola	0.2%	LG	0.3%
Others	0.5%	Others	0.6%

Figure 4-3 also supports the results from Table 4-3 and the proliferation of the Android OS as mentioned earlier where almost 50% out of the total flows have their OS as Android followed by the iOS with the share of 27%. Symbian, MeeGo and Windows Phone OSs are the main contributors to the share (23%) of Nokia mobile devices.

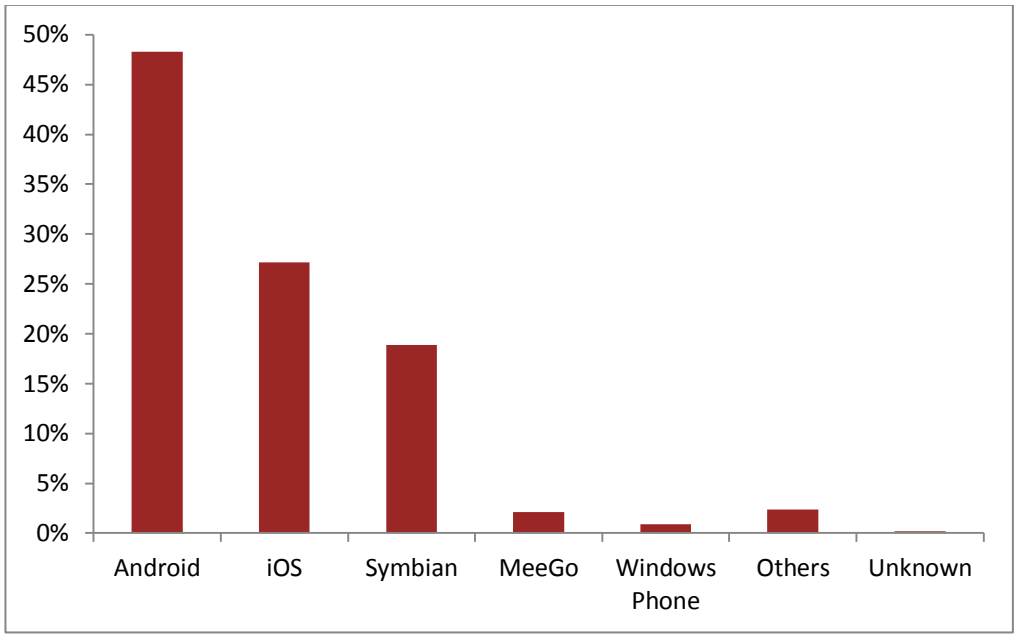

Figure 4-3: Operating system distribution (by flows) among the handheld devices

Share of traffic categorized as Unknown can be observed in the figure as well as the table. The UA strings that only contain mobile application (other than the browser) information in them are categorized as Unknown. These are handheld generated UA and classified as Handset/Tablet device types. However, these cases are later distributed among Handset and Table device types.

4.5.2 Operating System Shares of All Handset Traffic

From the illustration of handheld device top brands and highest number of flows generating device OS, we focus only on the traffic generated by mobile handsets in this section.

Figure 4-4 shows the shares of all handset OS generated HTTP flows and bytes. Similar trend can be observed in the case of handset OS as it was for handheld devices. Most of the identified devices are Android based devices with around 50% in terms of both transferred bytes and incurred flows. iOS is second with roughly 30% of transferred bytes and 20% of flows, followed by Symbian with 13% share of bytes and 22% flows. iOS share is contributed by the two Apple devices, iPhone and iPad. Out of the total handheld flows for iOS (27%) as shown in Figure 4-3, it is now clear that iPhone contributes 75% of the total flows for the iOS. The handsets with the Windows Phone OS account for roughly 1% out of the total flows, whereas the share for MeeGo OS is 2.5%. Nokia N9 is the most popular of the devices that use MeeGo OS and its share can be justified from the next section. There is a share of traffic that falls under unknown device OS as shown in the figure. The reason for this has already been outline in the previous section

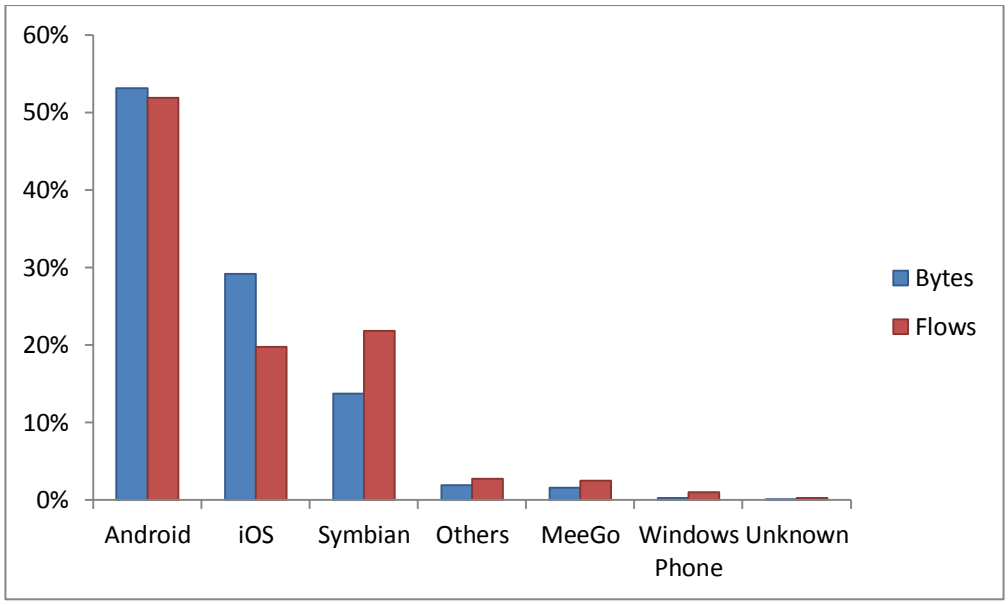

Figure 4-4: Shares of operating system generated handset traffic

In addition to the results obtained for the share of OS generated traffic, we further divide the OS share into two groups. UA generated by web browser is considered as one group and those generated by the applications other than the web browser is considered as the other. Figure 4-5 shows the proportions of browser and app-generated traffic, along with the share for the unknown, for each handset OS. App-generated traffic volume covers more than 60% of the total transferred volume for Android OS, whereas it covers only about 10% of the total flows. App-generated traffic volume for iOS, which is around 80%, is much higher than for any other OSs. The share of unknown for most of the OSs is found to be below 5%.

Symbian and MeeGo OSs show relatively high shares of browser generated traffic, which is more than 90%. This seems unrealistic as Symbian Apps are also popular in the market. One probable cause for this inaccuracy is due to the incapability of the tool to identify app-generated UAs from the browser generated UAs for Symbian OS, as most of the custom rules to identify app-generated UA string may have been focused on Android and iOS generated UAs. Another reason could be the case that the UA related to these devices fall to the device type *Unknown* (mentioned in section 4.4). From these arguments we can conclude that the application identification does not work well for all the platforms. Unidentified OS also shows browser and app-generated UA division, as a result of the UA strings that contain only application (other than the browser) or browser information and no other usable information in them.

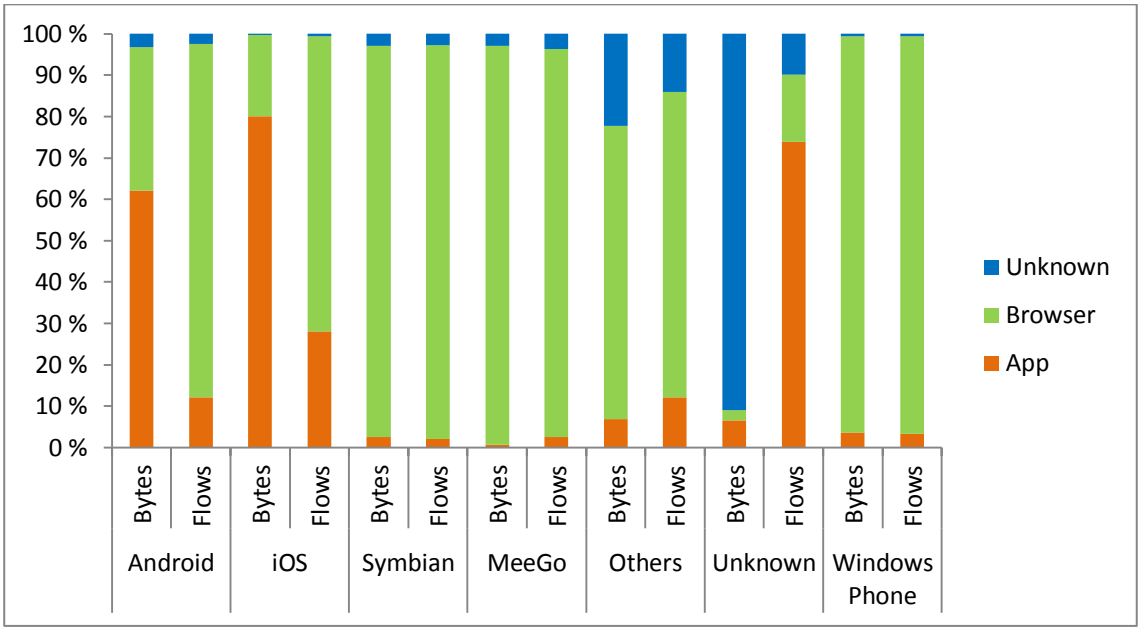

Figure 4-5: Shares of browser and app-generated bytes and flows

4.5.3 Popular Handset Models

The top 20 mobile handsets out of the total identified handsets shown are in Table 4-4. Apparently, most of the popular devices are using the Android OS, considering its large share among the handheld device OS. Although half of these popular devices use Android OS, the most popular model appears to be the Apple iPhone. It is then followed by Samsung Galaxy S II, Samsung Galaxy S, HTC Desire HD and Nokia E7. Results indicate that the handsets manufactured by Samsung are popular, with six out of the 20 handsets models in the popular handsets list. HTC handsets are also getting popular among the smartphone users and it is observed that three handsets model fall under the top 20 list. The most popular among the Nokia devices appears to be the Nokia E7. Based on the OS, out of the top 20 handset models, 11 are Android OS based devices, five models use Symbian OS, whereas only one handset each have Meego and Maemo as their OS.

Out of total transferred bytes, the results indicate that the Apple iPhone generates the largest amount of traffic which is roughly 30%. After the Apple iPhone, Samsung

Galaxy S II is found to generate the most traffic, around 6%. Considering the share of traffic volume, Nokia N8 with nearly 3% of the total transferred bytes has the largest share among the Nokia devices. Among the top 20 models, there are seven handset models from Nokia, five from Samsung and three the HTC. It is also interesting to note that even though the Generic Android handsets have the second largest amount of transferred traffic volume (19% out of the total); their share of out of total flow is quite low (below 1.5%).

Table 4-4: Share of bytes and flows for handset models

Handset model	Bytes	Handset Model	Flows
Apple iPhone	29.8%	Apple iPhone	20.0%
Generic Android Handset	19.0%	Samsung GT-I9100 Galaxy S II 16GB/32GB	8.9%
Samsung GT-I9100 Samsung Galaxy S II 16GB/32GB	5.6%	Samsung GT-I9000 Galaxy S	7.6%
Samsung GT-I9000 Galaxy S	4.5%	HTC Desire HD	3.4%
HTC Desire HD	3.0%	Nokia E7-00	3.3%
Nokia N8-00	2.6%	Samsung GT-S5660 Galaxy Gio	3.0%
Nokia E7-00	2.3%	ZTE Blade	2.9%
HTC Desire Z	2.0%	HTC Desire	2.7%
Samsung GT-I9001 Galaxy S Plus	1.9%	Nokia N8-00	2.6%
HTC Desire	1.7%	HTC Desire Z	2.5%
Samsung GT-S5570 Galaxy Mini	1.6%	Huawei U8800 Ideos X5	2.1%
Nokia N9-00 16GB/64GB	1.5%	Samsung GT-S5570 Galaxy Mini	2.7%
Nokia C7-00	1.5%	Samsung GT-I9001 Galaxy S Plus	2.7%
Nokia 5230	1.1%	Generic Nokia	1.7%
Nokia N900	1.1%	Nokia N9-00 16GB/64GB	1.6%
ZTE Blade	1.1%	Samsung GT-S5690 Galaxy Xcover	1.5%
Samsung GT-S5660 Galaxy Gio	1.0%	Generic Nokia Series 60 Handset	1.4%
Huawei U8800 Ideos X5	0.9%	Nokia C6-00	1.4%
Generic Nokia	0.9%	Nokia C7-00	1.4%
Nokia C6-00	0.9%	Nokia 5230	1.2%
Others	15.6%	Others	25.3%
Unknown	0.3%	Unknown	0.0%

This result of popular handset could however be applied a disclaimer as clear distinction between different models of Apple iPhone could not be achieved. Nevertheless, all the handsets listed in the top 20 most popular list support data standard of 200Kbps or greater (i.e. at least EDGE support). All handsets models have Wi-Fi except Nokia 5230, Generic Nokia and Generic Nokia Series 60. While most of the Android devices have FM radio feature, Apple iPhone does not have it.

4.5.4 Popular Tablet Models

Table 4-5 shows the shares of the top 10 tablet models, both in terms of the transferred traffic volume in bytes and the flows associated with them. Apple iPad dominates the top 10 tablet models in both the measures. It accounts for roughly half of the total traffic volume in bytes and 70% share of the total flows. Other than the Apple iPad, all other tablets among the top 10 have OS as Android. While roughly 30% of the total flows are generated by the Android based devices, they account for almost half of the total transferred traffic volume. The tablet model mentioned as Generic Android is comprised of the devices that are identified from UA strings where only device OS (Android) and the application name is present.

Table 4-5: Share of bytes and flows for tablet models

Model	Bytes	Model	Flows
Apple iPad	51.1%	Apple iPad	70.7%
Generic Android Tablet	42.0%	Generic Android Tablet	13.2%
Samsung Galaxy Tab GT-P7500	2.3%	Samsung Galaxy Tab GT-P7500	4.8%
Samsung Galaxy Tab GT-P1000	1.0%	Samsung Galaxy Tab GT-P1000	3.7%
Samsung Galaxy Tab GT-P7300	0.9%	Samsung GT-N7000 Galaxy Note	2.4%
Samsung GT-N7000 Galaxy Note	0.8%	Samsung Galaxy Tab GT-P7300	1.7%
Acer A500	0.4%	Asus Eee Pad Transformer TF101	0.5%
Asus Eee Pad Transformer TF101	0.2%	Acer A501	0.2%
HTC Flyer	0.1%	Opera Tablet (Android)	0.3%
Opera Tablet (Android)	0.1%	Acer A500	0.2%
Others	0.5%	Others	1.2%
Unknown	0.4%	Unknown	1.2%

4.6 Handset Features

As defined by Kivi et al. (2009), mobile handset features are new technologies that diffuse within the existing population of mobile handsets. These features enable new mobile services that complement and substitute the traditional voice calling and text messaging services. In this section, we present results related to the features of mobile handsets. We start by presenting results from the analysis of the handset characteristic such as the input method (e.g. navigation with either a keypad or a touch screen) and handset screen sizes. Then, the share of traffic generated by the handsets with selected feature in the Finnish mobile networks is presented.

4.6.1 Input Method

One of the most prominent features related to the usage of handsets is the way how the guided user interface (GUI) navigation is done. As variety of software and hardware features are being added to the handsets, the GUI navigation is also enhanced alongside, to adjust with the addition of new features. Thus, it is important to study the type of input method that the mobile handsets have for the GUI navigation.

Figure 4-6 shows the share of handset based input methods. Out of the total flows, roughly 77% is generated by the devices that are solely touch screen devices. Around 11% is generated by the devices that have both the touch screen and QWERTY keyboard as input methods. In the report related to the handset population in Finland from 2005 to 2010, Riikonen (2010) showed that out of total handsets in Finland, the highest shares of handsets had 3x4 numeric keypads. He further mentioned that the touch screen, QWERTY keyboard and other input methods were emerging to substitute the numeric keypad in the handsets. The results comply with the prediction made by Riikonen (2010) on the rapid penetration of touch screen and the replacement of numeric keypads.

Of the popular handset models presented in Table 4-4, 14 handset models have touch screen alone, and four have Touch + QWERTY as the input method. This suggests that

the handset devices that are mostly being equipped with touch screen capabilities and also with the addition of QWERTY keyboard generate most of the network traffic.

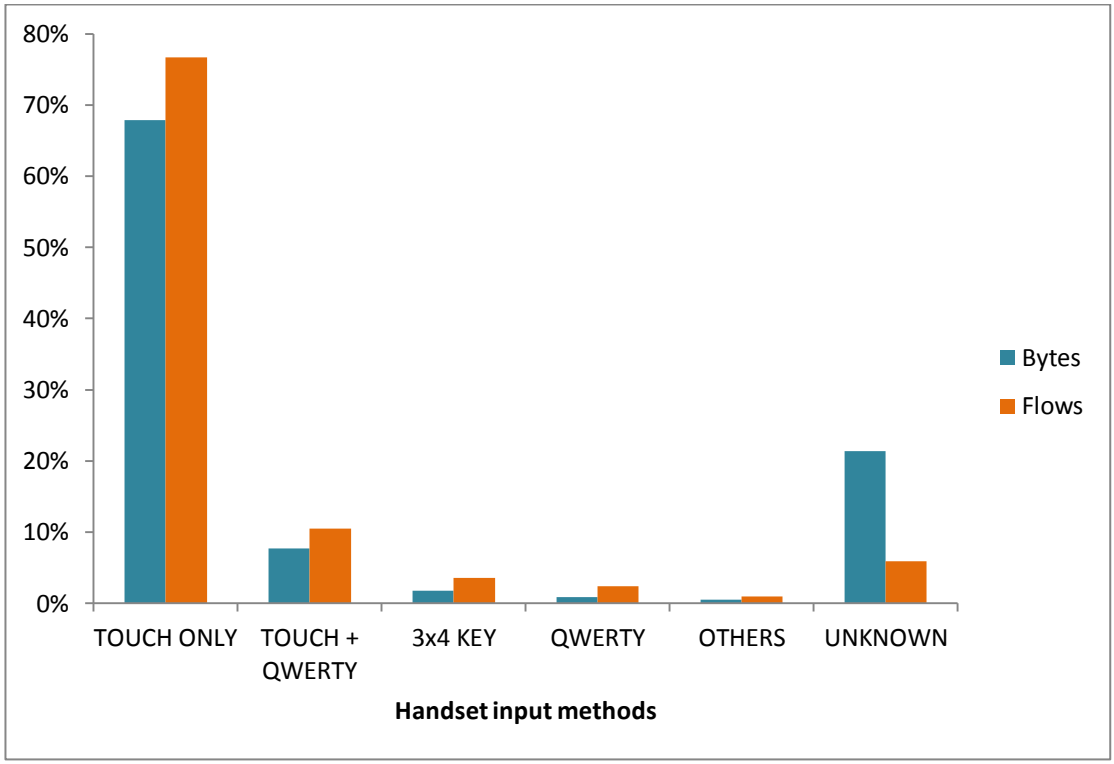

Figure 4-6: Share of bytes and flows for handset input method

4.6.2 Screen Size

To complement the use of mobile Internet and the evolving GUI input methods, suitable screen sizes are preferred. Most of the devices in the market now have rather large screen sizes as compared to the feature phones.

The share of devices with different screen sizes and their corresponding traffic volume is presented in Figure 4-7. Devices with the screen size of four inches generate roughly 70% of the total flows, and devices with screen size of three inches generate about 16%. Of the most popular handset models, Nokia C6-00, Nokia 5230, Samsung Galaxy Mini and Samsung Galaxy Gio have screen size of 3 inches, whereas all the other handset models, except the generic and unknown models, have screen size of 4 inches. This

information supports the results presented in this section to a considerable extent. There are around 6% of handsets which still have screen size of 2 inches. The figure also present the traffic volume associated with the different screen sizes.

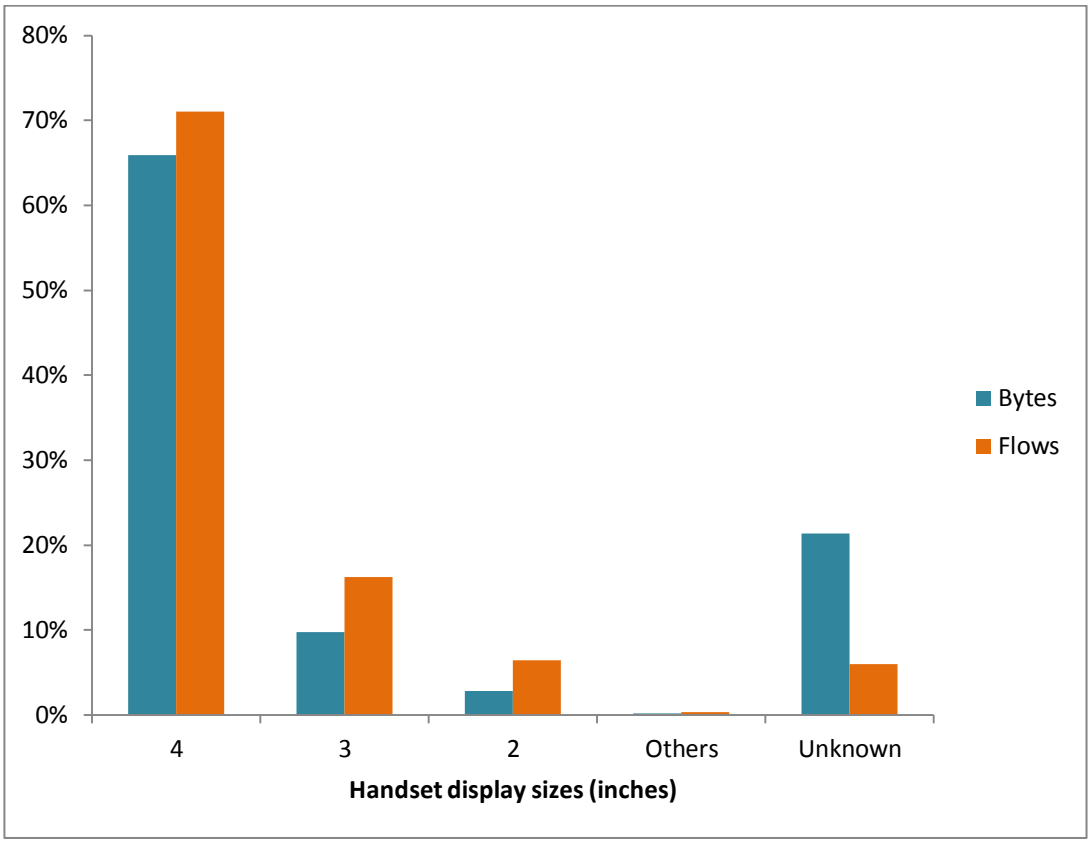

Figure 4-7: Handset display size (in inches) shares

4.6.3 Other Handset Features

The share of traffic volume based on the selected handset features is presented in this section. Selected features include the handset capabilities related to mobile data technologies and other features such as the wireless local area network (WLAN) or Wi-Fi, Global Positioning System (GPS), Bluetooth, FM Radio, Java support, Email and Camera as shown in Figure 4-8. The share of traffic volume for each of these features was calculated separately and put together in the same figure for comparison. The error bars present in the figure resulted from the terminals that do not have the feature or for which the data were not available.

Most of the identified devices had information regarding the features presented here. There are devices identified as Generic, such as Generic Android, Generic Nokia and Generic Nokia Series 60 handsets which do not provide information on these features. For this reason, initially the error margins in the handset penetration were huge. Later, the basic compatibility related to these generic handset models was identified and features relevant to our analysis were included as true for these devices. For instance, as 99% of the handsets within the Generic Android model have Android OS version 2.1 or greater, we assume the device has the basic hardware compatibilities as of the OS version 2.1. So, from the Google Inc. (2010) lists of basic hardware compatibility features for v2.1, information on the features relevant to our analysis were acquired. Features related to Generic Android acquired from the compatibility lists include Wireless data standard capability of 200Kbps or greater, Camera, Email, Java, Bluetooth, GPS, WLAN. Similarly, assuming the handsets identified as Nokia Series 60 has OS S60 2nd Edition or later, the handsets should have the basic hardware compatibilities as support for GPRS/EDGE/WCDMA, Email, Java, Camera and Bluetooth as defined by Nokia Corporation (2008).

Figure 4-8 shows the share of traffic volume for the selected features. Each bar separately represents the share of traffic volume for the handsets with that feature. Features like GPRS and EDGE are grouped together as all handsets that support EDGE also have GPRS support capability. It can be observed that many features are close to saturation which includes the 3G technologies such as WCDMA and HSDPA, messaging feature like the email, wireless connectivity features like WLAN and Bluetooth and multimedia related built-in camera feature. Apart from these features close to saturation, it is interesting to learn about the penetration of FM Radio in the handsets. As Apple does not incorporate built-in FM radio in its iPhone handset, the results show high share of handset population without this feature. This could raise a question about the saturation level for the FM radio feature. The penetration of the capability of downloading and running third party Java applications (e.g. games), has not quite reached the saturation limits. Penetration of the Java feature is seen to be nearly 65%.

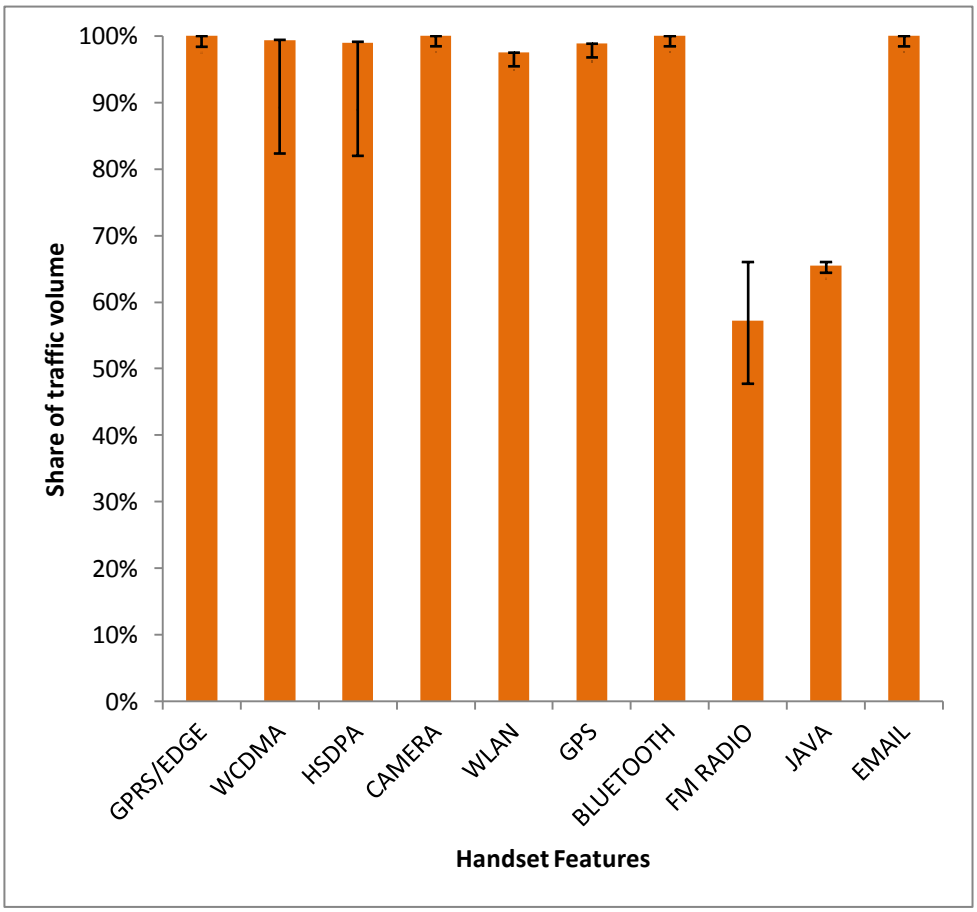

Figure 4-8: Share of traffic volume for selected handset features

5 Conclusions

This chapter provides the summary of findings and discusses the implication of these findings. Reliability and validity of the research are then evaluated and finally, suggestions regarding the extension of the current research work are presented.

5.1 Summary of Findings

The thesis worked on developing a tool to identify devices based on HTTP User Agent strings obtained from the mobile network traffic measurements. Based on the results of the tool, descriptive statistics related to the mobile device usage in Finnish mobile networks were provided.

We were able to observe that the tools used for the identification of mobile devices in the web servers for the purpose of content adaptation could also be used to identify devices from mobile network traffic traces. Out of the available tools, the tool developed in this thesis (the enhanced WURFL tool) used an open source DDR called the WURFL. The DDR was chosen based on its widespread use, comprehensive and state-of-the-art device repository, and availability of the APIs (suitable to different development platforms) to extract the information from the repositories. The enhanced WURFL tool was able to identify about 94% of the total UA strings subjected to the analysis. Results of the enhanced WURFL tool were the combination of the results from the implementation of WURFL alone and the enhancements made to certain results. Since, WURFL alone did not identify devices from the HTTP UA strings generated by applications other than the web browsers; custom rules and patches to the WURFL tool were created as those enhancements. The original WURFL tool was able to identify only about 80% of the UA strings. Out of the total UA strings parsed, about 0.5% of UA strings were identified inaccurately. The results for this inaccurate identification were addressed by the custom enhancements made to WURFL to produce accurate results. Thus, the custom enhancements made to the output of WURFL implementation improved the result roughly by 14% points.

General HTTP traffic characteristics showed the period of high traffic between 7pm to 9pm. Descriptive statistic from the device identification results showed that most of the traffic was generated by PC devices (roughly 87%), whereas handheld devices originated only about 6% of the total traffic. Out of the total handheld traffic, mobile handsets generated roughly 88%. Beside PC and handheld devices, less than 1% of the traffic was generated by other device types like the Apple iPod and Xbox.

In terms of the traffic volume, mobile device brand Apple seems to be consuming the network bandwidth with 37% out of total handheld traffic, whereas Samsung has the most number of flows. The share of Nokia is third with 14% and 21% both in terms of traffic volume and number of flows respectively. As there were UAs which did not include any specific device brand or model name in the string, they were categorized as Generic devices based on the OS information included in the UA string. Out of these, Android generic devices showed a significant amount of traffic volume and flows. From the operating system distribution among the handheld devices, the results indicate that Android with 48% of total flows has the highest share, followed by iOS with 27% and Symbian with 18.9%. Similar was the case for the handset OS, where handsets with Android OS dominates other OSs with its share of 52% out of the total flows. Results related to the distinction between the browsers generated UA and the app-generated was found to be effective for Android and iOS where these two UA types could be separated rather straightforwardly. However, the distinction for other OSs such as the Symbian and MeeGo seem unrealistic. Device identification results showed the most popular handset model as the Apple iPhone with 20% share out of total flows followed by Samsung Galaxy S II, Samsung Galaxy S and HTC Desire HD. Among the Nokia handsets, Nokia E7 (fourth on the list) is the most popular one with the share of around 3% out of the total flows whereas Nokia N8 generated the most traffic which is sixth on the list of highest traffic generating handset models. Results from the identification of the tablets showed that the most popular models is the Apple iPad with over 70% of the total flows and 50% of total traffic. The results also show the gaining popularity of Android tablets, especially the models from Samsung.

In addition to the results of device identification, selected features from WURFL and the MoMIE handset feature list were also extracted for the handset models. The identified handsets which generated the most traffic volume (77%) and flows (68%) had touch screen as the only input method. The share of traffic volume and flows for the devices with other input methods such as touch + QWERTY, 3x4 numeric keypad, QWERTY keyboard were very low. The results related to the screen size of the handset were also presented which was then justified with reference to the top 20 handset models results. Along with the results obtained for the input method and the screen sizes, share of traffic volume for selected handset features such as the supported mobile data technologies and other features such as WLAN, Bluetooth, FM Radio, Email, GPS and Java were also presented. The shares could also indicate that most of these features are close to saturation except FM Radio and capabilities related to Java.

5.2 Implication of the Results

This thesis had three objectives, developing a tool to identify mobile devices based on the HTTP User Agent, analyzing the accuracy of the tool and comparing it with String Matching results, and providing descriptive statistics to aid in profiling the mobile Internet usage in Finland.

The results of the enhanced WURFL tool show that it is possible to implement DDR and the related API to identify devices from the UA obtained by passive mobile network traffic measurements. Moreover, the output from the DDR implementation can be manipulated to improve the identification and get the results in desired forms. The results from the accuracy of the tool imply that it is reliable to implement open source and community contributed (WURFL) DDR and API for the purpose of mobile device identification. From the mobile operators' viewpoint, learning about general traffic characteristics provides them with the understanding of how the traffic varies throughout the day or week and helps in provisioning the network resources accordingly. Knowledge about the type and model of devices that access their network also helps in generating ideas for new strategies. Subscriber device identification could also be used in price differentiation when separating modem traffic from truly handheld

device originated usage. Significance of device identification is not just limited to identifying the device model, identifying the features related to the device models (most importantly the mobile handset models) also provides valuable insights to the need of network operator to tailor its services.

The results related to the overall device type classification were similar to the results from Erman et al. (2011), whereas, the handset features related results are more in line with the results from Smura et al. (2011). Results show the rapid emergence of Android based devices, out of which Samsung is the most popular among handsets. Apple products such as the Apple iPhone and the Apple iPad are the devices that consume most bandwidth in both handset and table device type categories. Based on this (or on any device that generated the most traffic), mobile operators and content providers can tailor their service to meet the existing users' demand and attract new users. In addition, device identification may possibly enable mapping of usage data with, for example, retail sales data to provide pricing information.

5.3 Reliability and Validity of the Research

The reliability and validity of the research methods and results are discussed in this chapter.

Reliability. The extent to which the results of a research can be reproduced under a similar methodology describes the reliability of the research. Like every other research, this research also has potential sources of error that affect the reliability. Several research phases might have included random error. In general, it should be remembered that network traffic measurements always include certain amount of uncertainty and no analysis methods are available which provide 100% accuracy. The mobile devices now-a-days provide the users with a facility to alter the UA that the device sends while communicating. As the research methodology and analysis were solely based on the identification of the device from the UA string, this alteration results in false identification of the devices. In the research, we had very limited detection of those types of UA strings which could make the results unreliable to some extent. On the

other hand, the analysis made use of the community contributed device data and there are possibilities of it being inaccurate. Post processing analysis which is done after results are obtained from the develop tool, also has potential places for human error. Measures such as transferred bytes and flows are taken into account to calculate the share of devices and generating statistics, where there is a possibility of mathematical calculation errors.

The concerns on the reliability of the input data, analysis of the data, and final results were addressed by providing detailed descriptions of the analysis phases and links between different statistical results. The case of altered UA strings in the analysis is the place where reliability of the develop tool could most likely be questioned.

Validity. The extent to which a research accurately reflects or assesses the objectives that the research is attempting to measure describes the validity of the research. Research results in general, should be assessed with both internal and external validity. This research provided descriptive results without attempting to make strong statements on the causal relationships behind the results. Thus, assessment of internal validity is not very relevant. However, while presenting the descriptive statistics, relationships between the results were outlined in the description to some extent. This included, for instance, validating the shares of the iOS operating system for handsets to the shares of Apple iPhone handset as it is the only one handset in the market that uses the iOS.

The data analyzed in the thesis were collected from two Finnish mobile network operators' network. One week worth of data was collected from a single GGSN for each operator, representing the Finnish mobile Internet usage rather well. The enhanced WURFL tool for device identification was able to correctly identify devices from roughly 95% of the total UA strings analyzed.

5.4 Further Research

The results presented in the thesis are limited to mobile devices types, brands, models, and features. The tool developed for device identification was also able to identify the applications that were generating the HTTP request. Results related to the application identification were unrealistic in the case of certain OSs as most of the applications identified were either for Android or the iOS. Nevertheless, the identified applications give early indications of the possible future developments. Further improvement to the enhanced WURFL tool can be made in this regard as more information about the actual market status can be acquired by examining the actual usage levels of the various applications enabled by the devices and networks (as described in Smura et al. (2011)).

Results in the thesis are only related to device identification which aids in profiling the mobile Internet usage. An interesting extension to this research work would be, combining the user session logs (from RADIUS) with the IP traffic measurement data to evaluate the characteristics of mobile Internet user sessions based on the device type, model, OS and device features. This can be done by mapping the IP address of the device and the corresponding timestamps to the IP traffic flow data that is used for device identification. In this way, device identification results and user session level data combined could provide insights on user behavior as well. Moreover, user session characteristics based on the applications could also be analyzed.

In addition to further research on identified applications and mapping user session data, business perspective could also be added to the analysis. Learning about the mobile services used by the mobile devices along with their usage session can help in developing better services which can bring added value.

References

The references are divided into three groups: Bibliography, Online Resources and Software Tools.

Bibliography

Adzic, V. K. (2011). A survey of multimedia content adaptation for mobile devices. *Multimedia Tools and Applications* , 379-396.

Adzic, V., Kalva, H., & Furht, B. (2011). A survey of multimedia content adaptation for mobile devices. *Multimedia Tools and Applications* , 379-396.

DetectRight. (2010, June). *White Papers and Research.* Retrieved June 7, 2012, from DetectRight: http://www.detectright.com/services/files/Device%20Detection%20And%20Databases.pdf

Erman, J. and Gerber, A. and Sen, S. (2011). HTTP in the Home: It is not just about PCs. *ACM SIGCOMM Computer Communication Review, Vol. 4, No. 1* , 90--95.

Fielding, R. and Gettys, J. and Mogul, J. and Frystyk, H. and Masinter, L. and Leach, P. and Berners-Lee, T. (1999). *RFC 2616 - Hypertext Transfer Protocol -- HTTP/1.1.*

Firtman, M. (2010). *Programming the Mobile Web.* O'Reilly Media, Inc.

Fling, B. (2009). *Mobile design and development: Practical concepts and techniques for creating mobile sites and Web apps.* O'Reilly Media, Inc.

Gartner. (2012b, April 10). *Gartner Says Worldwide Media Tablets Sales to Reach 119 Million Units in 2012.* Retrieved April 29, 2012, from http://www.gartner.com/it/page.jsp?id=1980115

Gartner. (2012a, February 15). *Gartner Says Worldwide Smartphone Sales Soared in Fourth Quarter of 2011 With 47 Percent Growth*. Retrieved April 29, 2012, from http://www.gartner.com/it/page.jsp?id=1924314

Gember, A., Anand, A., Akella A. (2011). A comparative study of handheld and non-handheld traffic in campus Wi-Fi networks. *Proceedings of the 12th international conference on Passive and active measurement, PAM'11* (pp. 173-183). Berlin: Springer-Verlag.

Georgiev, E & Georgieva, T. (2007). Methodology for mobile devices characteristics recognition. *Proceedings of the 2007 international conference on Computer systems and technologies* , 71.

Glover and Davies. (2005). Integrating device independence and user profiles on the Web. *BT Technology Journal* , Vol. 23, No. 3, pp. 239--248.

Google Inc. (2010). *Android Compatibility Program.* Retrieved May 31, 2012, from Android Open Source Project Web Site: http://static.googleusercontent.com/external_content/untrusted_dlcp/source.android.com/en//compatibility/2.1/android-2.1-cdd.pdf

GSMA Association. (2010, September 01). *IMEI Allocation and Approval Guidelines.* Retrieved May 06, 2012, from IMEI Database: http://imeidb.gsm.org/imei/DG06.pdf

Huang, J. J. S., Yang, S. J. H., Chen, Z. S. C., & Wu, F. C. C. (2012). Web content adaptation for mobile device: A fuzzy-based approach. *Knowledge Management & E-Learning: An International Journal* , 4(1), 102-122.

Huang, J. J. (2012). Web content adaptation for mobile device: A fuzzy-based approach. *Knowledge Management & E-Learning: An International Journal* , 4(1), 102-122.

Kalden, R. (2004). *Mobile Internet traffic measurement and modeling based on data from commercial GPRS networks.* Doctor's degree dissertation in University of Twente, Kassel, Germany.

Kilpatrick, J. A. and Cyr, R.J. and Org, EL and Dawe, G. (2006). New SDR architecture enables ubiquitous data connectivity. *RF DESIGN, Vol. 29, No. 1* , 32.

Kivi, A. & Riikonen, A. (2009). *IP Traffic Measurements 2008 - Mobile Internet Usage Pattern. MoMI project report.* Espoo: Dept. of Communications and Networking, TKK.

Kivi, A. (2009). Measuring mobile service usage: methods and measurement points. *International Journal of Mobile Communications, Vol. 7, No. 4* , 415-435.

Kivi, A. (2007). Measuring Mobile User Behavior and Service Usage: Methods, Measurements Points, and Future Outlook. *Proceedings of the 6th Global Mobility Roundtable, 1st--2nd June.* Los Angeles, California.

Kivi, A. (2008). *Mobile Data Service Usage Measurements - Results 2005-2007.COIN project report.*

Kivi, A. (2006). *Mobile Internet Usage Measurements - Case Finland.* Master's Thesis. Networking Laboratory, Helsinki University of Technology, Espoo, Finland.

Kivi, A. (2006). *Mobile Internet Usage Measurements - Case Finland.* Master's Thesis. Networking Laboratory, Helsinki University of Technology.

Kivi, A., Smura, T. and Toyli, J. (June 2009). Diffusion of mobile handset features in Finland. *In Proceedings of the Eighth International Conference on Mobile Business* , 209.

Maier, G. and Schneider, F. and Feldmann, A. (2010). A first look at mobile hand-held device traffic. *Proceedings of the Passive and Active Measurement* (pp. 161--170). Springer.

Nokia Corporation . (2008). *Resource information.* Retrieved May 31, 2012, from Nokia Developer: http://nds2.fds-forum.nokia.com/p/d/fds_forum/961c0d01-8932-4623-aeb0-162ecdcbbe85/S60_Platform_Developers_Introduction_v1_0_en.pdf/S60_Platform_Developers_Introduction_v1_0_en.pdf?fdptoken=1338557217_fdeead4e7f0e8764ded961af06d625dc

Nokia. (2011, October 14). *Detecting Mobile Devices on Web Services*. Retrieved June 03, 2012, from NOKIA Developer: http://www.developer.nokia.com/Community/Wiki/Detecting_Mobile_Devices_on_Web_Services

OMA. (2006, February 06). *User Agent Profile.* Retrieved June 03, 2012, from Open Mobile Alliance: http://www.openmobilealliance.org/technical/release_program/docs/UAProf/V2_0_1-20070625-A/OMA-TS-UAProf-V2_0-20060206-A.pdf

Omari, R. and Feisst, M. and Christ, A. (2008). Analysis of the Important Mobile Devices Features to Improve Mobile Web Applications. *iJIM, Vol:2, Number:2* , 33.

Passani, L. (2007-2010). *New WURFL API and why it is important to upgrade*. Retrieved December 15, 2011, from ScientiaMobile Web site: http://wurfl.sourceforge.net/newapi/

Postel, J. (September 1981). *RFC 793 - Transmission control protocol.* USC/Information Sciences Institute.

Riikonen, A. (2012). *Mobile Handset Population in Finland 2005-2011. MoMIE project report.* Technical report in Aalto University, Espoo.

Riikonen, A. (2009). *Mobile Internet Usage - Network Traffic Measurements.* Master's Thesis. Networking Laboratory, Helsinki University of Technology, Espoo, Finland.

Shafiq, M. Z., Lusheng, J., Liu, A. X. & Wang, J. (2011). Characterizing and Modeling Internet Traffic Dynamics of Cellular Devices. *Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems* (pp. 305--316). San Jose, California, USA: ACM SIGMETRICS '11.

Smith, C. & Grundl, P. (2004, March 02). *Know Your Enemy: Passive Fingerprinting*. Retrieved November 23, 2011, from Honeynet Project - http://project.honeynet.org: http://old.honeynet.org/papers/finger/

Smura, T. and Kivi, A. and Toyli, J. (2011). Mobile data services in Finland: usage of networks, devices, applications and content. *International Journal of Electronic Business* , Vol. 9, No. 1, pp. 138--157.

Smura, T., Kivi, A., Töyli, J. (2009). A Framework for Analysing the Usage of Mobile Services. *INFO - The journal of policy, regulation and strategy for telecommunications, information and media* .

Sugai, P. (2007). Exploring the impact of handset upgrades on mobile content and service usage. *International Journal of Mobile Communications, Vol.5 No. 3* , 281-299.

Tstat. (2008). *Measurement; Torino, Telecommunication Networks Group - Politecnico di*. Retrieved April 28, 2012, from http://tstat.tlc.polito.it/measure.shtml

W3C. (2004, February 10). *RDF Primer.* Retrieved June 03, 2012, from W3C Recommendation: http://www.w3.org/TR/rdf-primer/

Yarochkin, F. (1998, October 18). *Remote OS detection via TCP/IP Stack FingerPrinting*. Retrieved May 04, 2012, from nmap.org: http://nmap.org/nmap-fingerprinting-article.txt

Yarochkin, F.V. and Arkin, O. and Kydyraliev, M. and Dai, S.Y. and Huang, Y. and Kuo, S.Y. (2009). Xprobe2++: Low volume remote network information gathering tool. In *IEEE/IFIP International Conference on Dependable Systems \& Networks, 2009. DSN'09.* (pp. 205--210). IEEE.

Online Resources

51Degrees.mobi Limited. (2010-2012). *Device Data*. Retrieved June 07, 2012, from 51Degrees.mobi: http://51degrees.mobi/Products/DeviceData.aspx

Antenna (2011a). Antenna Expands Device Database. Retrieved November 25, 2011 from Antenna Press Releases: http://www.antennasoftware.com/news/press-releases/2-17-2011-antenna-expands-device-database

Antenna (2011b). AMP Server. Retrieved 25, 2011, from Antenna Software: http://www.antennasoftware.com/products/mobile-platform

Antenna (2011c). About. Retrieved November 25, 2011, from Antenna Software: http://gateway.volantis.com/docs/archives/fw/fw_6_0_0/drws_admin/drws_about.html

Budde, Paul Budde Communication Pty Ltd. (January, 2012). *Global - Mobile - Smartphones, Touchscreen Tablets and Handset Market Insights.* Retrieved May 5, 2012, from Research and Markets: http://www.researchandmarkets.com/reports/1877183/global_mobile_smartphones_touchscreen_tablets

Business Wire (2006). Volantis Mobile Device Database Expands to 3000+ Devices; 95% Of US Devices Now Covered; Database Now Available via Standalone Licensing Program. Retrieved November 25, 2011, from TheFreeLibrary: http://www.thefreelibrary.com/Volantis+Mobile+Device+Database+Expands+to+3000%2B+Devices%3B+95%25+Of+US...-a0148666362

DetectRight (2007). How accurate is DetectRight?. Retrieved November 28, 2011, from DetectRight: http://detectright.typepad.com/detectright/2007/12/index.html

DetectRight(2011a). Landing Page. Retrieved November 25, 2011, from DetectRight: http://detectright.com/page/index

DetectRight(2011b). Landing Page. Retrieved November 28, 2011, from DetectRight: http://www.detectright.com/products_and_services.html

DetectRight(2011c). DetectRight Product and Services. Retrieved November 28, 2011, from DetectRight: http://detectright.com/page/products

DetectRightAMF (2011a). AMFDetectRightFilter. Retrieved November 25, 2011, from ApacheMobileFilter Wiki: http://wiki.apachemobilefilter.org/index.php/AMFDetectRightFilter

DetectRightAMF (2011b). AMFDetectRightFilter. Retrieved November 25, 2011, from ApacheMobileFilter Wiki: http://wiki.apachemobilefilter.org/index.php/Mobile_web

dotMobi (2011a). Help. Retrieved November 20, 2011, from DeviceAtlas™: http://deviceatlas.com/help#1_6

dotMobi (2011b). Devices. Retrieved November 25, 2011, from DeviceAtlas™: http://deviceatlas.com/resourcecentre/explore+deviceatlas+data/devices

dotMobi (2011c). Data Explorer. Retrieved November 25, 2011, from DeviceAtlas™: http://deviceatlas.com/resourcecentre/explore+deviceatlas+data/data+explorer#_/

dotMobi (2011d). DeviceAtlas Products. Retrieved November 28, 2011, from DeviceAtlas™: http://deviceatlas.com/products/enterprise

dotMobi (2011e). DeviceAtlas Products. Retrieved November 28, 2011, from DeviceAtlas™: http://deviceatlas.com/products/cloud

dotMobi (2011f). About Our APIs. Retrieved January 3, 2012, from DeviceAtlas™: http://deviceatlas.com/resourcecentre/Get+Started/About+Our+APIs

Galoppini, R. (2008). Open Source Mobile: Volantis eventually released Mobility Server under the GPLv3. Retrieved November 28, 2011, from Roberto Galoppini's blog: http://robertogaloppini.net/2008/03/24/open-source-mobile-volantis-eventually-released-mobility-server-under-the-gplv3/

MobileAware. (2010). *MobileAware*. Retrieved June 07, 2012, from MobileAware: http://www.mobileaware.com/

Mobile Phone Wizards AS (2007). DetectRight – how does it compare? Retrieved November 25, 2011, from DetectRight Library: http://library.detectright.com/DetectRight_Compared.pdf

Nokia. (2011, October 14). *Detecting Mobile Devices on Web Services*. Retrieved June 03, 2012, from NOKIA Developer: http://www.developer.nokia.com/Community/Wiki/Detecting_Mobile_Devices_on_Web_Services

OpenDDR LLC. (2011-2012). *openddr*. Retrieved June 07, 2012, from openddr: http://www.openddr.org/

Passani, L. (2007-2010). *New WURFL API and why it is important to upgrade*. Retrieved December 15, 2011, from ScientiaMobile Web site: http://wurfl.sourceforge.net/newapi/

ScientiaMobile (2012a). Help. Retrieved May 29, 2012, from ScientiaMobile-WURFL: http://wurfl.sourceforge.net/help_doc.php

ScientiaMobile(2012b). *Online Contibutors Help*. Retrieved June 7, 2012, from ScientiaMobile: https://db.scientiamobile.com/static/contributors_help.htm#gettingmostoutofuseragentstrings

ScientiaMobile, Inc. (2011-2012). *WURFL Licensing Terms*. Retrieved June 7, 2012, from wurfl: http://wurfl.sourceforge.net/wurfl_download.php

Telecomlead (2011). dotMobi unveils low-cost DeviceAtlas device detection service. Retrieved November 28, 2011, from Telecom Lead: http://www.telecomlead.com/inner-page-details.php?id=1392

VolantisSystems (2009). Volantis Mobility Server™. Retrieved November 28, 2011, from VolantisSystems: http://www.volantis.com/files/Volantis_Mobility_Server_Overview.pdf

W3C(2006). Volantis Systems Ldt. Position Paper. Retrieved November 20, 2011, from W3C MWI DDWG - Workshop on the Implementation of a Device Description Repository: http://www.w3.org/2005/MWI/DDWG/workshop2006/papers/Volantis/

WURFL (2011). WURFL Official APIs. November 28, 2011, from ScientiaMobile: http://wurfl.sourceforge.net/

Software Tools

WURFL Repository version 2.3, available online:

<URL: http://sourceforge.net/projects/wurfl/files/WURFL/2.3/ >

WURFL Java API version 1.3.1.1, available online:

<URL: http://sourceforge.net/projects/wurfl/files/WURFL%20Java%20API/ >

Eclipse IDE, available online:

<URL: http://www.eclipse.org/ >

Appendix A

Table A-0-1: Device capabilities/features extracted from WURFL and the Feature List

Capabilities/Features	Name Description
brand name*	Mobile device brand name (e.g. Apple)
model name *	Mobile device model (e.g. iPhone)
device os *	OS name of the device
device os version *	Version of the device os
mobile browser*	Information about the device browser (e.g. Opera, Mobile Safari)
mobile browser version *	Which version of the browser
marketing name *	Some devices have a marketing name (e.g. HTC Ace) in addition to Brand and Model
pointing method *	Input method is either a joystick, stylus, BlackBerry-style clickwheel or touchscreen
has qwerty keyboard *	If the device has a full qwerty keyboard. This can also be a virtual keyboard
is tablet *	If a device is a tablet computer (iPad and similar, regardless of OS)
has cellular radio *	Device has cellular technology (most probably a phone, but not necessarily. May be a data-only device such as iPod touch or N800
max data rate *	Maximum bandwidth reachable by the device. HSDPA = 1800/3600/7200/14400, UMTS(3G) = 384, EGPRS/EDGE = 200, GPRS = 40
resolution height *	The screen height in pixels
resolution width *	The screen height in pixels
built in camera *	If the device has a built-in camera
is wireless device *	If a device is wireless or not. UA strings from web desktop browsers have this feature FALSE
Input method**	GUI Navigation with either 3x4 + QWERTY, 3x4 KEY, QWERTZ/QWERTY, TOUCH ONLY, TOUCH + 3x4, TOUCH+GAMINGKEY or TOUCH+QWERTY
Camera**	If the device has built-in camera
GPRS**	If the device supports GPRS
WLAN**	If the device can access Wi-Fi
EDGE**	If the device supports EDGE
WCDMA**	If the device supports WCDMA

HSDPA**	If the device supports HSDPA
GPS**	If the device features GPS
FM Radio**	If the device has built-in FM radio
Bluetooth**	If the device has Bluetooth capability
Java**	If the device OS supports Java
Manufacturer**	Device manufacturer's name
Display size (in inches)**	Display size expressed in inches
Email**	If the device provides a full email client

* Extracted from WURFL
** Extracted from the handset feature list