An Openflow-based Wireless User Management System

By

Eduardo Luengo

A Thesis Submitted in Partial Fulfillment

Of the Requirements for the Degree of

Master of Applied Science

In

The Faculty of Engineering and Applied Science

Department of Electrical/Computer Engineering

University of Ontario Institute of Technology

January 2016

Acknowledgment

I would like to express my sincere gratitude to my supervisors Dr Shahram Heydari and Dr Khalil El-Khatib for their expert guidance, involvement and patience during the elaboration of this thesis. Moreover, I would like to thank my family and friends for their love, encouragement and support. Lastly, I would like to acknowledge my colleagues at the Advanced Network Technologies and Security (ANTS) Research Lab for the friendly and cooperative environment they always provided.

Abstract

The degradation of the end-to-end Quality of Service (QoS) on wireless users is becoming a common issue for IEEE 802.11 infrastructure-based networks in crowded areas. In response, network providers usually increase coverage and capacity to serve the ever-increasing simultaneous connections and traffic loads. However, wireless users still experience unacceptable QoS at the application layer (i.e.: video and audio streaming). In many cases, this issue is due to the nature of the IEEE 802.11 standard that makes wireless users select the target AP to camp on based on signal strength, regardless of traffic load supported by the AP at the moment. This thesis tackles the issue from a different perspective than previous works, by employing an SDN framework on an integrated wireless/wired environment. Thereby, we present the development and implementation of a system that performs user management by analyzing the network load from the Openflow statistics, as well as, the wireless information collected from the associated users. In order to analyse the behaviour of the proposed user migration algorithm, we evaluate the system under scenarios with different traffic load and user session duration. From the experiments, we observed that in several cases, wireless users get a considerable QoS improvement at the application layer when the system is activated. Based on the results, we present an analysis on how and when user migration in multi-AP IEEE 802.11 networks is most effective.

Publications

The following conference paper has been published as part of the work of this thesis:

Luengo, Eduardo, Christopher Weber, Shahram Shah Heydari, and Khalil El-Khatib. "Design and Implementation of a Software-Defined Integrated Wired-Wireless Network Testbed." In Proceedings of the 13th ACM International Symposium on Mobility Management and Wireless Access, pp. 47-52. ACM, 2015.

The following manuscript has been submitted and is currently under review:

Luengo, Eduardo, Shahram Shah Heydari, Khalil El-Khatib, "OpenFlow-based Wireless User Management System for Improved User QoE", IEEE ICC'2016 workshop on Quality of Experience-based Management for Future Internet Applications and Services", 23-27 May 2016, Kuala Lumpur, Malaysia

Table of contents

1. Introduction 1
1.1. Background 1
1.2. Problem Statement 3
1.3. Contribution 4
1.4. Thesis Organization 6

2. Related Works 7
2.1. Traditional Approaches 7
2.1.1. User-driven Approaches 7
2.1.2. Network-driven Approaches 8
2.1.3. Standard IEEE 802.11 Approaches 11
2.2. SDN in Wireless Networks 12
2.3. Summary 20

3. System Architecture 22
3.1. System Overview 22
3.2. Considerations and Assumptions 24
3.2.1. System Design Considerations 24
3.2.2. Wireless Considerations 25
3.2.3. SDN Considerations 26
3.3. Software Defined Wireless Network Controller 26
3.4. Wireless User Application 38

4. User Migration Algorithm 40

5. Emulation Model 55
5.1. Model Description 56
5.2. Implementation Considerations 62
5.3. Proof-of-concept 65
5.4. Technical Details 69

6. Performance Evaluation 70
6.1. Scenarios 70
6.2. Results 73

7. Conclusion and Future Works 85

8. References 87

Figures

Figure 1. WUMS architecture 23
Figure 2. Message diagram of SDN topology information 28
Figure 3. Message diagram of SDN Port Statistics for a given AP 29
Figure 4. Message diagram of SDN Flow Statistics for a given user 30
Figure 5. Message diagram of Available Networks and Current Network information 32
Figure 6. Message diagram of user migration and confirmation after migration 34
Figure 7. UML representation of the SDWNC software 37
Figure 8. Available networks and current network information from WUA 38
Figure 9. Process chart of the UMA 42
Figure 10. Flow chart of the Monitoring process 44
Figure 11. Flow chart of the Balancing Factor Analysis 47
Figure 12. Flow chart of the User Analysis 49
Figure 13. Flow chart of the Selecting users to migrate process 52
Figure 14. Flow chart for determining the target AP 54
Figure 15. Emulated model 58
Figure 16. Network connectivity in the testing environment 60
Figure 17. Description of internal connectivity for the SDWNC, web services/applications and wireless users 61
Figure 18. Emulation of wireless users in a single laptop 62
Figure 19. Architecture design of the flows update solution 64
Figure 20. Flow update after user migration in SDN switch 65
Figure 21. Configuration of the bandwidth limitation policy on an AP 65
Figure 22. Balancing factor indicator before and after the WUMS migrate the wireless users 68
Figure 23. Load distribution on each AP during the test 68
Figure 24. Scenario of new light and old heavy users in over-loaded AP 71
Figure 25. Scenario of new heavy and old light users in over-loaded AP 72
Figure 26. Overall UDP traffic rate improvement in 80H/20L with and without BT . 74
Figure 27. Overall delay improvement in 80H/20L 75
Figure 28. Average number of ICMP responses received by wireless users in 80H/20L with and without BT 75
Figure 29. Disconnection gap per wireless user for 80H/20L with and without BT ... 76
Figure 30. Overall UDP traffic rate improvement in 50H/50L with and without BT . 77
Figure 31. Overall delay and jitter improvement in 50H/50L with and without BT ... 78
Figure 32. Average number of ICMP responses received by wireless users in 50H/50L with and without BT 79
Figure 33. Disconnection gap per wireless user for 50H/50L with and without BT ... 79
Figure 34. Overall UDP traffic rate improvement in 20H/80L with and without BT . 80
Figure 35. Overall delay and jitter improvement in 20H/80L with and without BT ... 81
Figure 36. Average number of ICMP responses received by wireless users in 20H/80L with and without BT 82
Figure 37. Disconnection gap per wireless user for 20H/80L with and without BT ... 82
Figure 38. Example of overall delay and jitter of heavy and light users obtained in 80H/20L scenario 84

Tables

Table 1. Example of sorted users list 50
Table 2. Comparison of SDN environments 56
Table 3. Wireless users association after migrations 67
Table 4. Different testing scenarios for the WUMS 73

1. Introduction

1.1. Background

Wireless technologies have revolutionized human life allowing suitable, fast and reliable ways of communication anywhere. Starting with voice call and text messaging services, wireless communications have evolved to provide ubiquitous voice and data access for humans and machine-to-machine communications. Frequently, it is common for wireless users roaming around a city to get Internet connections not only from cellular networks but from nearby Wireless Local Area Networks (WLANs) deployed in the area. In particular, the ever-increasing presence of public/private WLANs allows high-speed connections to transient users with portable devices such as laptops, phones and tablets. These WLANs are commonly implemented with the IEEE 802.11 standard. This standard is a set of specifications that was originally designed to provide convenient high-speed wireless communications to a reduced number of computers by utilizing unlicensed radio-frequency spectrum [1]. The IEEE 802.11 standard was essentially developed for small-office-home-office (SOHO), with radio ranges of up to 100 meters without extenders [2]. Since then, the standard has been revised in order to enhance technical aspects related to modulation techniques, frequency range of operation, as well as, to introduce spatial diversity capabilities. In the case of the IEEE 802.11n amendment, the enhancements allowed data rates transmissions of up to 600Mbps and radio ranges of up to 250 meters in outdoor environments [3].

Usually, IEEE 802.11 user devices at hotels, airports, coffee shops, and universities scan the wireless channels in the area for nearby IEEE 802.11 access points (APs), in order to join a network with Internet access. These IEEE 802.11 networks are defined by the standard as infrastructure-based, in contrast to ad-hoc-based networks, whose links are established between users (peer-to-peer) without any additional device. The former requires that wireless users associate with an AP before being able to communicate. Other infrastructure-based IEEE 802.11 networks, commonly known as hotspots, are found in crowded areas. Hotspots can be accessed by transient or stationary users on a public or private basis. Usually, municipalities or service providers own these networks [4] and [5]. For instance, mobile companies usually

deploy hotspots to avoid traffic congestion in their cellular networks offering higher-speed connections to their customers in a given area [4]. Moreover, aside from the different types of IEEE 802.11 infrastructure-based networks, there is a sustained interest to extend coverage and capacity of WLAN domains. For instance, service providers usually make agreements with their SOHO broadband customers or other WLAN providers of a given area, in order to offer better connection services to transient customers. As a result, numerous APs are part of a large topology [6] and [7].

However, the predominant adoption of the standard, which was originally developed for confined environments, and the high demands of traffic generated by wireless users raise concerns in terms of performance and scalability. In this regards, common issues encountered in crowded IEEE 802.11 networks are related to user management, user mobility, radio frequency interference from IEEE 802.11 and non-IEEE 802.11 sources, security and quality of service (QoS) [8].

Apart from adoption and related issues of IEEE 802.11 networks, an emerging technology, originally applied to data centers, promises to revolutionize the design and management of network architectures as a whole. This technology, referred to as Software Defined Networks (SDN), aims at changing the traditional network design and management by achieving more flexible, low-cost, open standard, and scalable infrastructures. Specifically, the main concept of the framework is to achieve physical decoupling of the control plane from the forwarding plane of network devices, and thus eliminating the historical model of proprietary control plane implementations [9]. The SDN approach fosters the creation of less computational complexity devices, resulting in the deployment of relative low-cost network topologies. Moreover, SDN facilitates network management due to its centralized control (SDN controller) plane, in contrast to traditional network architectures, in which the control plane is distributed among the nodes.

In a SDN architecture, the network is managed by a SDN controller. The SDN controller allows the communication with high-level components by means of standardized interfaces (northbound interfaces), as well as, with lower-level components by means of other standardized interfaces (southbound interfaces). Usually, a SDN controller has knowledge of the entire topology and network status by retrieving traffic measurements from the network devices. This information serves as

an input to makes dynamic traffic handling decisions and installs forwarding rules in network devices. Based on the forwarding rules installed by the SDN controller, a network device simply applies the best matching rule to each packet received. By doing so, the data traffic is forwarded in and out the network through the forwarding device ports. Moreover, SDN switches can be dedicated hardware (i.e.: Lanswitches and APs) or software instances running on top of multi-purpose computer hardware. The interaction between the SDN controller and the network devices (SDN switches) is carried out by means of a standardized southbound SDN protocol, such as Openflow. Among all its capabilities, this protocol allows the collection of statistics and the configuration of Quality of Service (QoS) and traffic shaping [10]. Moreover, the northbound interface of an SDN controller is usually develop as Representational State Transfer (REST) APIs. In this way, most of the network complexity is hidden, so that upper services/applications get information from the network and manage certain aspects of the traffic in a standard and flexible way.

Over the years, the applicability of SDN in IEEE 802.11 networks has been gaining popularity. One reason for this approach is to discover simple, flexible and standardized solutions to deal with the traditional IEEE 802.11 management issues mentioned before. Another reason is that researchers have always pursued the need of a framework capable of integrating wired and wireless environments in a straightforward manner to foster continuous innovation with low investments [18] and [50].

1.2.Problem Statement

Crowded areas have several APs with spare capacity for wireless users to connect to. These APs may belong to the same administrator or they are part of a cooperative framework of different administrators. However, regardless of the network capacity distributed among the available APs in the area, wireless users experience poor connection services. The reason is that a single AP is usually supporting large volume of traffic from several wireless users while other APs remain idle. Hence, when the over-loaded AP can no longer accommodate all the traffic demands from the associated users, congestion occurs. From the user's perspective, this leads to the degradation of the QoS, affecting the end-to-end throughput, delay, jitter and packet-

loss of services/applications (i.e.: video/audio streaming) [38], [51], [52], [53] and [54].

The fact that many wireless users connects to the same AP is due to the inherent mechanism of the IEEE 802.11. The standard allows wireless user devices to decide which AP to associate with, based on partial information such as the strongest signal-to-noise ratio and theoretical transmit rate. A clear example of this, is a scenario where there is an empty room with overlapping wireless coverage provided by distributed APs. In this case, as IEEE 802.11 users enter the room, it is likely that users will associate to the closest AP to the entering door (due to its highest signal strength), despite the available capacity on other APs. Hence, letting users un-deliberately perform this task, without a complete view of the network topology and status, leads to the allocation of large amount of traffic in some APs, while others remain idle. Thus, the mere presence of under-loaded APs in the vicinity does not solve the problem.

In order to tackle the issue, a user management system is required. User management solutions distribute the user traffic load across nearby APs so that network congestion can be avoided. In fact, commercial solutions that utilize standard protocols (ie: CAPWAP/LWAPP) are available to mitigate the problem [30], [31] and [32]. Nevertheless, these solutions are being developed in a proprietary manner, favouring the selection of specific hardware and software. Moreover, several approaches using traditional architecture were considered in the past [25], [26], [27], [28], [38], [39], [40] and [41]. However, all these approaches required additional entities/agents and/or specific protocols (i.e.: SNMP) that makes the architecture more complex to manage. Lastly, with the advent of SDN, research regarding user management in IEEE 802.11 networks has been conducted. However, again, these efforts follow non-standardized approaches to solve the issue [11], [12], [13], [14] and [15]. This is due to the lack of out-of-the-box SDN protocols capable to support IEEE 802.11 channel parameters and statistics. Thereby, the need for open, standard, flexible and vendor-independent approaches remains unaddressed.

1.3. Contribution

Considering the fact that an SDN architecture maintains a general view of the network topology, IEEE 802.11 network issues can now be tackled from a different

perspective than with traditional wireless solutions. This thesis presents the design and analysis of a dynamic wireless user management system for infrastructure-based IEEE 802.11 networks. The novel approach uses the SDN architecture in order to mitigate load-balancing issues in IEEE 802.11 networks. The proposed system collects statistics from the SDN network, as well as, wireless information from associated users, without introducing modifications to the southbound interface (Openflow protocol) for that purpose. In this way, the system detects traffic load asymmetry on APs, and consequently makes user migration decisions to distribute the load among nearby APs.

As in [24] and [38], the system aims at improving the user QoS by maximizing the end-to-end throughput and, at the same time, reducing the end-to-end delay, jitter, packet-loss, and the disconnection time elapsed of wireless users during migrations.

Apart from its forwarding decision tasks, the SDN architecture assists the system in the collection of network information such as the topology database and traffic statistics from every AP connected to the network. Moreover, the system interacts with associated users for both getting network-scanning information from the IEEE 802.11 wireless medium, and migrating users to other nearby APs.

The main contribution of this thesis includes the design and analysis of:

- **System Architecture:** A description of the components involved in the system, the definition of their roles and interactions between each other.
- **Algorithm:** The design and implementation of a wireless user migration algorithm to mitigate the load asymmetry detected by the system.
- **Emulation:** A detailed description regarding technical aspects of the real environment used to test the system
- **Performance evaluation:** A study of different unbalanced scenarios and the results obtained when the user management approach is applied. In addition, this thesis analyse the impact of the user migration algorithm on the QoS

1.4. Thesis Organization

The remainder of this thesis is organized as follows. Chapter two tackles the related works in the field. Chapter three explains the design and requirements of the system. Chapter four presents the proposed algorithm. Chapter five describes the technical aspects related to the emulated model. Chapter six presents the results obtained from the experiments. Finally, chapter seven concludes the thesis and discusses future works.

2. Related Works

This chapter discusses previous work in the field of user management for IEEE 802.11 networks. The prior work in this field is primarily limited to load balancing among APs through user migration, which is one of the primary functions of a wireless user management system. Besides, the chapter provides a walk through the research related to traditional user management solutions in IEEE 802.11 networks, and the recent load-balancing approaches utilizing the SDN framework. Throughout the years, state-of-the-art solutions have been suggested to solve the load-balancing issue in IEEE 802.11 infrastructure-based networks. Many solutions employed traditional networking architectures while, recently, others used the SDN framework. Nevertheless, a common aspect of all user management approaches is the fact that they all share the same design principles. These are the definition and monitoring of metrics, an algorithm to determine the asymmetry and the causes, and finally, a mechanism to distribute the load among other nodes of the network. Moreover, approaches can be preventive or reactive in response to unbalanced conditions in the network. The former usually utilizes access control policies to reject new associations in over-loaded APs. The latter dynamically accommodates the current load of the associated users by means of association mechanisms applied from the AP or centralized wireless controller.

2.1. Traditional Approaches

In this section we review some research on wireless user management prior to SDN. Though the approaches are all based on traditional networking architectures, differences exist among them in terms of system design. In general, the literature classifies these solutions according to the location in which the user management decisions take place; be it a network-driven or client-driven [24].

2.1.1. User-driven Approaches

Over the years, user migration algorithms based on user decisions were suggested [25], [26], [27], [28] and [37]. These approaches provided users with meaningful

wireless information, other than the signal strength, in order to accomplish a smarter connection choice from available APs. For example, Virgil [25] was presented as a user-level application running on the user device. Virgil made the user device to associate to every AP in the vicinity, and performed a series of test to determine bandwidth and round-trip-time values, by contacting TCP/UDP servers in the Internet. Later, Virgil stored the performance evaluation results of each AP in a local database, and selected the AP with the highest performance. This approach resembled a brute-force method in which network resources were wasted while wireless users found their best AP to finally camp on.

Another example of user-driven was the Application Layer Load Distribution protocol (ALDP) [37] in which an SNMP server assisted users in the task of collecting wireless metrics. The architecture required that both APs and user devices run a SNMP agent. The dynamic user migration approach measured congestion to detect an over-loaded condition in the network. The SNMP server provided users with information regarding the degree of congestion on each AP, so that users could re-associates to a different AP if needed. The server collected information about the utilization of the AP wireless interface and the current number of users on each AP. Consequently; the server calculated the remaining capacity on each AP (residual bandwidth). Both residual bandwidth and number of users were sent to the user device, which utilized that information to compute a normalized value of the remaining capacity on each available AP. Based on the obtained value, the user selected the AP with sufficient capacity that would not affect its current load. In order for ALDP to work, it required each user to know the server IP address in advance (or learn from any entity on the network) and establish a registration. The solution fitted on small WLAN scenarios. However, the amount of traffic generated by user registrations would not go unnoticed in crowded environments.

2.1.2. Network-driven Approaches

Alternatively, many other user management approaches were positioned inside the network [38], [39], [40] and [41].

In [38], a network-driven load-balancing scheme was presented. The architecture consisted of overlapped APs configured in different IEEE 802.11 channels and equipped with additional software called Load Balancing Agent (LBA). In addition, the architecture did not require any central server or extra modification on user devices. Basically, APs shared load information between them (using the wired network) and performed an analysis to determine whether user management actions are required or not. This user migration approach was reactive but also preventive in nature because it rejected upcoming association on over-loaded APs. The algorithm considered the uplink/downlink throughput per AP as the load metric for the analysis, identification of the cause of asymmetry and, finally, for load distribution. Furthermore, the algorithm utilized a fairness formula based on AP throughputs to detect asymmetry in the network. In case of imbalance, it compared the load on each AP with the average load of all APs (L), in order to classify APs as over-loaded, balanced or under loaded. Consequently, the algorithm proceeded by moving the excessive load (in the over-loaded APs) to under loaded APs. In order to select users to migrate from an over-loaded AP, the algorithm considered those users whose load contributions reduce the current AP load close to L. The mechanism used to perform a user migration was simply a disassociation message sent from the AP to the user device, allowing the user to re-associate to a nearby AP. The authors presented their results based on a scenario with two AP and two users. The results shown improvements on user's packet delay and throughput in the long term, when load balance was activated. However, in the case of migrated users, the data traffic was interrupted for approximately three seconds while the user was re-associating with the new AP. This fact, considerably increased the delay, and it was due to the unassisted handoff mechanism utilized.

Sawma *et al.* [40] presented a network-driven load-balancing algorithm called ALBA (Autonomic Load Balancing Algorithm). The architecture required a central server to collect measurement from the wireless environment such as the channel utilization, the spatial distribution of users and APs, as well as the QoS profile of each user. The authors explained that the algorithm and central server could be conceived as part of a Knowledge Plane that retrieved status information, measurements and statistics from traditional network architecture. Therefore, the network could utilize this information to tackle issues in a dynamic and automatic manner. This load-balancing algorithm,

worked by migrating existing users from a given AP to other nearby APs. This was to provide enough capacity to a user that was previously experiencing low performance. In other words, once the algorithm was aware that there was an affected user in the network, the algorithm selected a better AP for the user to camp on. Before migrating the user to the target AP, ALBA migrated current associated users from the target AP to other nearby APs. By alleviating traffic load on the target AP, ALBA accommodated user traffic loads throughout the network in an even manner. The algorithm was tested in an overlapping IEEE 802.11b network with wireless users generating UDP VoIP traffic. The results of this work shown enhancement in terms of global end-to-end delay (55% improvement) and throughput on the network (13.6% improvement).

A more recent user management approach was introduced in Le *et al.* [39]. The architecture was based on a centralized server and multiple overlapping WLAN APs. In the architecture, the centralized server performed AP association/disassociation for wireless users based on the results of two algorithms. The first algorithm focused on a proactive scheme that assigned each user to the lightest AP in the area, by performing the association process in a certain order. In this way, the algorithm first organized sets of users by the number of APs that the users could reach. Then, the algorithm started the re-association process of users, whose set had the minimum number of APs reachable. For each user, the algorithm found the candidate AP that had the minimum current load. When an AP was previously assigned to a user, that AP was no longer available for other users in the same set. Once finished with a set, the algorithm continued the association process of the next set of users. This procedure provided control on the distribution of users during the association stage, by assigning an AP to a user and, by checking the current load on each AP. On the other hand, the second algorithm worked as a reactive user management algorithm. The goal of the second algorithm was to reduce the load on the over-loaded APs. For each user on an over-loaded AP, the algorithm analyzed the current load of user`s candidate APs, and selected the AP with the minimum load as target AP. Once users on the over-loaded AP reached lighter APs to switch to, the algorithm terminated. The results shown an improvement on user throughput when the approach was compared with the traditional IEEE 802.11 mechanism. In addition, measurements of the Jain fairness index returned values closer to 1. This indicated a fair share of network resources

between wireless users, when the approach was activated. This study did not include results of other network performance metrics such as delay, jitter and packet loss.

2.1.3. Standard IEEE 802.11 Approaches

Moreover, there were some standardization efforts related to the concept of centralized architecture and user management in IEEE 802.11 networks. For instance, the CAPWAP [29] was an IETF protocol for control and provisioning of WLAN APs. Apart from AP devices, the CAPWAP architecture required a central controller. The protocol ensured a standard for a controller and APs to communicate with each other, regardless of vendor hardware specifics. In this way, functions of authentication, policy enforcement and collection of wireless channel information were localized in a central entity in the network. Moreover, with centralized control functions, CAPWAP fostered the reduction of computer power per AP. The standard presented two modes of operation: Split MAC and Local MAC. The former decoupled the data frames and management frames, by processing data frames locally and forwarding the management frames to a controller. The latter supported IEEE 802.11 data frames and management frames locally. Many commercial solutions utilized CAPWAP [30], [31] and [32]. However, as mentioned in the literature [12], [14] and [15], there was no definition of central entity that could offer an open interface for network application in order to manage and control the WLAN infrastructure. Commercial solutions based on the CAPWAP protocol were developed in a proprietary manner, and they often lacked of interoperability among other CAPWAP vendor solutions [43].

Another example of standardization in WLAN environments was the IEEE 802.11k amendment [33]. It was developed as a standard way to provide detailed reports of physical (radio) and data link layer of IEEE 802.11 environments, so that upper layer applications could make use of it. IEEE 802.11k allowed local and remote measurements from wireless users, as well as, measurements on different channel than the one the user was connected to. The performing of measurements was possible without the need for user re-association. IEEE 802.11k offered different metrics that could be used to mitigate load asymmetry in the network [34]. For instance, IEEE 802.11k beacons reports provided a list of all APs in range for a given user, which could be utilized by a central controller to build a map of users and APs in the area.

Moreover, channel load reports informed the channel load utilization, which could be used to estimate the AP load where the user was associated to. Lastly, station statistics returned the AP average access delay that could be utilized as one parameter to measure the performance before and after a user management action was executed. Despite IEEE 802.11k brought new capabilities to acquire detailed information about the wireless environment, it required changes on both the AP and user device software to support new frame format and procedures [35] and [36], which is not always supported in commercial wireless cards. In addition, the traffic overhead generated by user reports might affect the network throughput in scenarios with several wireless users [35].

2.2. SDN in Wireless Networks

With the adoption of SDN architectures in networks of large companies such as Google, Facebook, Yahoo, Microsoft, Verizon, and Deutsche Telekom [16], the SDN framework has been gaining reputation constantly in the research community, as well as, in the industry. As a natural step, researchers are currently analysing SDN potential to solve common issues on other environments. In this way, research regarding the integration of SDN in IEEE 802.11 networks has emerged as an alternative approach to overcome the limitation of current network infrastructures.

OpenRoads [42] was an open-source platform developed by Stanford University that integrated an entire real wired and wireless (IEEE 802.11 and 802.16) SDN network. The OpenRoads focused on slicing a production network in order to perform independent and concurrent test on the same infrastructure. User-level software was used to equip access points (APs) with Openflow and tunnelling capabilities.

Suresh *et al.* [11] presented Odin, a programmable SDN framework for WLANs. The architecture consisted of OpenWRT APs and a Floodlight SDN controller, which collected wireless statistics and parameters from each AP (i.e.: signal strength, bit rate, last packet timestamp, etc), by employing a separate southbound API different from the Openflow protocol. In order to accomplish that, the architecture required two software modules; a master module running on the controller, and an Agent module

running on each AP. One of the key components introduced by Odin was the Light Virtual Access Point (LVAP) that allowed a wireless user to stay associated with a unique virtual BSSID, while moving along the overlapping APs zone. This was to permit users to switch from one physical AP to another seamlessly (without service disruption). In this way, this approach aimed at solving handoffs on overlapped WLANs, without the need of re-association and re-authentication. Besides, the authors defined Odin as a distributed WiFi split-MAC architecture. The reason was that physical AP performed some of the layer 2 wireless operations (i.e.: IEEE 802.11 ACKs), while the SDN controller helped in performing others operations such as, user association management. Therefore, the authors stated that Odin was different from other approaches like CloudMAC [12], which concentrated many of the AP operations in the cloud. The Odin architecture implemented events to handle the dynamic changes in the wireless network, which was a valuable feature to reduce unnecessary traffic when collecting AP information. In addition, the paper described a series of application that were developed on top of the framework to manage mobility, interference, dynamic channel selection, energy efficiency on APs, policy enforcement and user management. Though many of these applications are not directly related with the topic of this thesis, an interesting point to highlight is regarding the mobility management approach followed by the authors. Basically, the mobility application collected the user`s signal strength values from all AP (LVAPs). In this way, the application built a general view of the user`s signal strength, and thus, it selected the AP with the highest signal strength for that user. However, Odin still rose some concerns. For instance, the LVAP overhead per associated user was approximately 80 bytes. This could limit Odin in terms of scalability, since SOHO APs are usually built with constrained hardware resources. Furthermore, regarding the ACK mechanism implemented on each AP, the authors highlighted a practical limitation. This was the fact that, in some cases, the verification of the destination address received from associated users (BSSID) might lead to the incorrect generation of ACKs messages, and consequently, these messages might be destined to BSSIDs that were not hosted by the LVAP in the first place. Lastly, though Odin allowed seamless handoffs, APs needed to be configured on the same IEEE 802.11 channel. Otherwise, Odin would require the support of IEEE 802.11h; an amendment developed for the 5 GHZ frequency spectrums.

Dely *et al.* [12] developed the CloudMAC architecture for WLANs using SDN technologies. The architecture consisted of OpenWRT APs, Openflow switches, a POX SDN controller and VMs running virtual instances of IEEE 802.11 access points. The concept behind CloudMAC was to withdraw part of the MAC layer functionality from physical APs and perform these MAC layer operations on virtual APs (running on standard servers) in the cloud. This approach differed with Odin in that, the physical APs served just as relaying elements between wireless users and virtual APs. In this way, MAC layer operations such as encryption and control headers were performed on virtual APs rather than in the physical APs. However, physical APs performed some strict-time operations independently, such as ACKs and frame retransmissions. In a way, the CloudMAC resembled the split MAC mode of the CAPWAP specification. Besides, the CloudMAC relied on SDN technologies to forward the users' downloading and uploading data traffic. One of the advantages of CloudMAC was the flexibility to accommodate users on different channel in the same AP. In this way, the author described a scenario where a user connected to physical AP with multiple wireless cards could be switched from one crowded channel to an uncongested one (in the same AP) without re-association required. As in Odin, this feature relied on the IEEE 802.11h amendment. Next, the authors highlighted the reduction of beacons frames transmitted over the wireless medium by implementing an on-demand approach. Hence, beacons frames were only transmitted when probe requests (sent by users) were detected. Furthermore, the CloudMAC architecture introduced an increment in the round-trip-time (from 1.79 to 2.28 ms for median values). This was due to additional processing between network elements (Openflow processing and tunnelling). Unfortunately, the author did not provide an analysis on network latency, considering that CloudMAC elements (physical and virtual APs) could be conceived to work in different physical locations. In fact, strict-time messages like ACKs and frame retransmissions were implemented locally on each AP in order to avoid the latency introduced by distance. In contrast to the Odin architecture [11], it is worth mentioning that CloudMAC provided a smooth integration of wireless and wired SDN elements because it did not require an agent (additional software development) in the APs nor extra southbound API to handle the wireless issues.

Ethanol [15] was another SDN architecture for IEEE 802.11 infrastructure-based networks. It was conceived to provide the following features: mobility, user management, security, QoS and localization of wireless users, the collection of wireless links statistics, and the virtualization of SDN wired/wireless deployments. Though the article did not include a complete set of experiments of all these features, it provided some preliminary results on QoS, ARP overhead and load-balancing. The latter will be discussed later in this chapter. In essence, Ethanol consisted of a SDN controller (POX) and commercial home APs running user-level Openflow software (Pantou 1.0). Each AP was equipped with an additional software module that allowed the collection of IEEE 802.11 wireless links statistics and QoS management from the SDN controller. The authors highlighted the fact that the Ethanol architecture supported QoS programmability in a per-flow basis; by utilizing a modified version of Pantou (Openflow software for commercial APs) that gave the SDN controller complete management of QoS parameters. The paper shown a simple study case in which a different bandwidth policy was applied to a wireless user and to two wired users connected to an Ethanol AP. In this way, Ethanol provided evidence that the Openflow protocol could be used to enforce QoS policies in IEEE 802.11 networks. Moreover, the work included results regarding a reduction on the ARP traffic in the wireless medium. This was possible by employing an algorithm on the SDN controller that compared the IP address of the ARP traffic received with its internal ARP database. Consequently, the comparison would determine whether the traffic had to be flooded to all ports or just forwarded to specific ports. Unfortunately, this first development of Ethanol controller did not consider the definition of a northbound API, which discouraged upper services/applications to collect IEEE 802.11 information from the controller on a standardized manner.

Other researches shown considerable efforts on extending the Openflow protocol for IEEE 802.11 networks. Patro *et al.* [13] developed the Coordination framework for Open APs (COAP), which was implemented with SOHO APs utilizing OpenWRT and the Floodlight SDN controller. COAP was primarily destined for IEEE 802.11 networks in large building complex. The authors defined a software architecture and southbound API, in which an SDN controller could retrieved wireless channel information from APs. The information exchanged between the SDN controller and APs was the airtime utilization, user and neighbouring APs statistics and beacon

statistics among others. The paper shown improvements in the available throughput of a given AP, due to the reduction on the airtime utilization on the wireless channel. By identifying the traffic type on the wireless channel, the SDN controller could apply a throttled or slotted technique, so that high priority traffic was not affected by low priority traffic. The slotted technique accommodated high priority traffic on certain time slots while left low priority traffic on other time slots. This technique helped avoiding congestion on the wireless channel. In fact, the results shown that when utilizing the slotted technique between two APs, with a hidden user affecting the traffic of a non-hidden user, the throughput of the non-hidden user was improved four times. Another important feature to point out in this architecture was that the SDN controller could keep track of previous wireless information regarding the AP surroundings (from IEEE 802.11 and non-IEEE 802.11 sources). The authors referred to it as context information that could be used to develop channel congestion avoidance algorithms and interference management solutions.

Kim *et al.* [14] presented the OpenFlow AP architecture (OFAP) to provide seamless handoff and improve network throughput by mitigating the performance anomaly issue in overlapped wireless environments. As part of the OFAP architecture, a wireless extension for the Openflow protocol was developed. The approach did not require any changes to user devices. Moreover, each AP (OFAP) was registered to a SDN controller and performed user management, wireless channel monitoring and hardware control. The implementation was achieved by employing the Kulcloud MuL controller and wireless APs with the Openflow 1.0 protocol. As in Odin, users associated to a virtual AP instead of a physical one. In order to estimate the data rate of a given user to migrate, the controller periodically collected the user signal strength levels and data rates from all APs. In this way, the system computed the correlation between user signal strength values and user data rates. As a result, the controller selected the most convenient AP for the user to camp on, based on the current user data rate. When the algorithm was activated, the results shown no degradation of throughput when a user moved from one AP to another. However, the OFAP architecture required all APs to be configured with same BSSID, SSID and IEEE 802.11a channel, so that users perceived that only one AP was presented. Thought that configuration scheme might be applicable to some cases, it is not the most commonly scenario found in crowded places where BSSID, SSID and channels may vary from

one network administrator to another. Finally, the paper did not mention the impact of traffic load in the network due to the monitoring traffic, considering that each AP was required to compute and report the signal strength to the SDN controller periodically.

Murty *et al.* [23] presented an architecture for WLANs called Dyson, under a joint effort of Microsoft with Harvard University. Despite the fact that the architecture used a proprietary southbound protocol, the authors claimed that Dyson had been inspired by the same principals of SDN and that the compatibility with Openflow was being considered. Dyson allowed the implementation of customized policies on top of the architecture in order to manage common WLAN issues (i.e.: interference, user management, VoIP handoffs). Dyson provided a set of API, in which APs and users could send their wireless information such as, radio channel conditions, performance measurements and location to a central controller. In addition, the central controller of Dyson was aware of historical knowledge from users regarding their space and time patterns. With all these information the central controller could build a general view of the network that later could be used to develop several policies for management and control. Despite the client modification requirements imposed on Dyson, which has been pointed out by the literature [14], Dyson provided accurate wireless channel information from users, which served to create complete view of the network. In this way, 802.11 issues such as the hidden terminal, handoff and user migration could be tackle in a more precise way.

So far, this section has discussed the general approaches encountered regarding the SDN framework in IEEE 802.11 networks. Next, we present the early efforts addressing the load-balancing problem in IEEE 802.11 networks employing the SDN framework.

Suresh *et al.* [11] and [50] introduced a reactive user migration algorithm for overlapped WLANs that run on top of the Odin framework. It simply collected the associated users of each AP and their respective signal strength values. With that information the user migration algorithm built a map that indicated the candidate AP for each user. In this way, the application re-distributed the LVAPs (users) based on the number of users that they were currently associated on each physical AP. In order to determine the candidate AP for a given user, the application selected the AP with the highest signal strength that would not affect the current user throughput after migration. By applying this algorithm, the results shown that 50% of the clients got

better throughput in comparison with 15% that resulted when the user management mechanism was disabled. Though the issue was mitigated, the metric used to balance the network (the number of user on each AP), was not adequate to apply in data networks, since the load contribution usually differs from one user to another [38]. Moreover, the criterion of choosing a target AP only by its signal strength value, cannot be considered a smart decision in terms of user management. Hence, the simplicity of the algorithm may lead to incorrect user migration results in some cases. To sum up, further analysis should be provided to determine the robustness of the proposed algorithm.

Furthermore, Moura *et al.* [15] proposed a user management solution that worked on non-overlapping WLANs in the Ethanol architecture. Instead of querying the network periodically, like Odin did, this approach relied on the reception of events sent by APs in order to detect load asymmetry on the network. During the association and authentication of a new user, an Ethanol AP generated events to the SDN controller. As a response, the SDN controller applied an access control policy to allow/deny or redirected a new user to another AP. In this case, the metric used for user migration decisions was the number of users on each AP (which was compared with the predefined minimum number of users allowed per AP). The authors focused on a preventive algorithm that left the user with the freedom of choice of camping on a different AP, in case of rejection from a crowded AP. To conclude, this approach just served as a mechanism to easily control user association in IEEE 802.11 networks. Despite the fact that the algorithm worked in a SDN environment, it followed the same approach as many commercial solutions based on traditional network architectures [19] and [20].

In the case of the OFAP, the paper did not provide a user management mechanism. However, it is interesting to point out the criterion chosen to perform handoff of user with low data rates. Giving that the OFAP considered a scenario where APs are configured on the same IEEE 802.11 channel, the traffic of all users competes for the wireless medium. In a scenario with these characteristics, the performance anomaly issue becomes relevant [21] and [22]. The issue occurs when a low-data-rate user captures the wireless medium for a given period of time, and current high-data-rate users are forced to negotiate a lower speed (with the AP), which eventually turns out in equal chances for all users to access the medium. The authors of OFAP explained

that when there was no clear candidate AP (due to similar signal strength values as the current AP) to move a low-data-rate user, the target AP was selected by looking at the most crowded and heavy loaded AP. Though this approach may seem contradictory at first, the authors shown that this approach actually avoided the degradation of user throughput. Crowded and heavy loaded APs, usually leaves upcoming users with low channel occupancy; hence a new low-data-rate user would not affect current users due to the low medium occupancy required by the user. However, a low-data-rate user in a light loaded and uncrowded AP, would likely occupy the medium for long periods of time affecting the current high-data-rate users. The authors compared the results of the performance anomaly reduction handoff technique with a traditional handoff technique based only on the AP signal strength. The results shown an improvement of 26.7% in the throughput of all users. The performance anomaly was also analysed in Dyson [23] to solve the user migration problem. In this case, the authors analyzed the asymmetry observed when few upload users attempted to use the wireless medium of several download users. In this case, both AP and upload users were the only parties waiting to take control over the medium in order to transmit. This led to unfairness for all download users that were required to wait until the AP took control over the medium again. The user migration policy presented in Dyson, aimed at achieving fairness among upload and download users in a given AP by a client cooperation technique. Basically, the policy sorted upload users by their throughput, in decreasing order, and then throttled their traffic until the upload traffic reduces its predominance in the medium. The experiment was carried out with the policy activated, as well as, with the policy deactivated. The results evidenced the presence of fairness among users in terms of network capacity when the policy is activated. Despite this technique tackled the performance anomaly issue on a given AP without the need of user migration, it did not consider user management issue in the network as a whole. However, this policy presented in Dyson can be considered as complementary to user management solutions in an effort to achieve fairness in an intra-AP basis, as well.

2.3. Summary

The traditional IEEE 802.11 approaches previously mentioned utilized coupled network architecture to address wireless user management. The common limitation factor of these approaches was the lack of flexibility and integrity regarding wireless and wired networks. Else, they required the presence of a particular IEEE 802.11 amendment and/or new network entity in the network, or the customization of existing network protocol to mitigate the problem. Thereby, these solutions made the network architecture more complex and disjoined.

On the other hand, SDN show promises to address IEEE 802.11 issues from the network perspective in a standardized and flexible manner. However, the issue is not fully covered in SDN based-architectures. Though some efforts regarding user management with SDN architecture were proposed [11], [15] and [50], the network metrics utilized in these approaches (number of users associated to an over-loaded AP) were not adequate to reflect the load in data networks.

However, this thesis aims at developing a user management system for IEEE 802.11 networks by utilizing the Openflow statistics to detect network asymmetry. Besides, this approach provides a broader vision to tackle IEEE 802.11 issues (i.e.: mobility management, security, etc.), rather than tackle the particular load balancing problem. The proposed approach does not require protocol/standard customization in order to collect wireless information to mitigate the issue. Moreover, our design is not limited to an enterprise environments like commercial solutions [30], [31] and [32], in which all APs belong to the same administration. Rather, it is conceived to address the issue in scenarios where multiple APs from the same or different services providers may be present. In these scenarios, where the network load may be distributed among APs of different service providers a reactive approach is more realistic. This is in contrast to load balancing approaches, commonly applied to enterprise environments that use admission control mechanisms in order to solve the problem. In those approaches, the user management system operates during the association phase by rejecting or allowing new users to join APs. Hence, once a wireless user has been authenticated and associated to a given AP, the user management mechanism does not perform further operations with that particular user.

Moreover, instead of utilizing the number of users as a load metric, the proposed algorithm utilizes APs and users' traffic rate, as well as, the users session duration to distribute load across APs. Also, the system assists wireless user during migrations. This is to assure that the user performance is not significantly affected by unnecessary delay during the migration process. Lastly, due to the fact that using the same IEEE 802.11 channel on all APs might lead to a physical wireless channel congestion in crowded scenarios, the system presented in this work does not limit APs to be configured on the same channel like Odin [11] and OFAP [14] do.

Finally, the proposed approach differs from others, in the way that it is conceived to work in the application layer in a standard and straightforward manner by utilizing the information already available in the network and without introducing modification/customization in the architecture. Working in lower layers provides fine-grained information of the medium that definitely helps solving issues related with, for example, the dynamics of wireless environments. However, working at the application layer provides the flexibility to mitigate issues from an end-to-end perspective as part of an integral QoS solution for wireless users.

3. System Architecture

As mentioned in the previous chapter, the IEEE 802.11 user management issue has been addressed in the past, by using traditional network architectures. However, this thesis studies the introduction of a novel approach that uses SDN technologies to mitigate the issue. This chapter describes aspects regarding the system design and functionality.

3.1. System Overview

The proposed approach is a Wireless User Management System (WUMS) for WLANs under an integrated SDN architecture. The WUMS corrects unbalanced conditions, by moving users that generates excessive load on a given WLAN AP, to other under-loaded APs in the vicinity. By doing so, the WUMS maximizes the network throughput in order to improve the wireless user QoS.

The way the WUMS works is reactive in nature. Essentially, it monitors the network periodically in search for network asymmetry. Once an unbalanced condition arises, the WUMS determines the amount of load that must be redistributed in order to counteract the effect. However, moving the excessive load in a WLAN means migrating the wireless users that are causing the issue. Hence, the WUMS selects wireless users on the over-loaded AP and, consequently, migrates them to other under-loaded APs. Eventually, the balanced condition is re-established in the network.

The WUMS requires a mixture of wired and wireless network information for its operation. On the one hand, it periodically contacts the SDN controller via the northbound REST API. This is to collect wired network information. However, in order to learn about the wireless environment, the WUMS acquires wireless information from the associated users by contacting them directly.

Regarding migrations, the WUMS provides assistance to wireless users. As a result of this approach, the disconnection gap experienced by wireless users during the migration process can be reduced. This is mainly because the tasks related to the selection of the target AP (i.e.: network scanning) are performed while the wireless user is still associated to the over-loaded AP. Therefore, wireless users do not need to waste time in scanning and re-association attempts in order to connect to another AP.

The WUMS is made of two components working together. One is the Software Defined Wireless Controller (SDWNC) located on top of the SDN controller. The other one is the Wireless User Application (WUA), which is installed on every wireless user device. Figure 1 depicts the SDN environment for this WUMS.

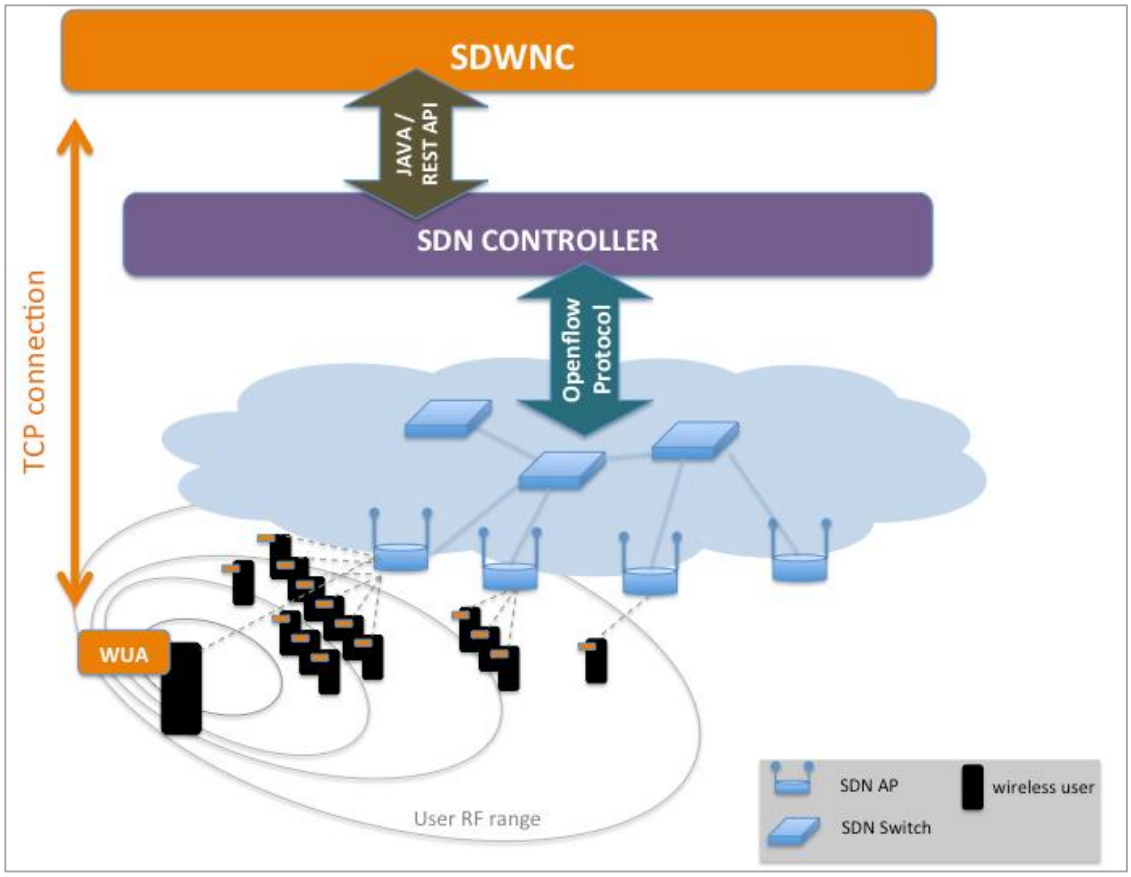

Figure 1. WUMS architecture

Notice in Figure 1 that the SDN network consists of switches and APs that are equipped with Openflow protocol. In this way, a SDN controller manages the forwarding decisions in the network. On the other hand, SDWNC is a central component in the architecture that communicate with both the SDN controller via the northbound interface (REST API) and with the multiple WUAs using TCP connections. Finally, WUAs inform the SDWNC about wireless information collected on their RF range.

3.2. Considerations and Assumptions

In this section we summarize design considerations and assumptions for our WUMS architecture, wireless environment and SDN controllers.

3.2.1. System Design Considerations

- Reactive user migration approach: The WUMS was developed as a reactive mechanism for unbalanced conditions in SDN wired/wireless networks. Hence, wireless users must be associated to APs in order for the WUMS to perform user migration
- The WUMS allows the administration of more than one SDN controller domain. The WUMS was initially conceived to collect network information from several SDN controllers, though this work considers only one SDN controller
- The WUMS is entirely developed in JAVA programming language, due to its platform-independent, object-oriented and multithreading capabilities, as well as its compatibility with current SDN controller implementations
- The WUMS requires connectivity with the SDN controller and the wireless users at the same time
- In order to achieve reliable control message exchange, the WUMS uses the connection-oriented protocol TCP
- The WUMS monitors the load of those APs that belongs to the SDN topology only. It does not monitor other devices
- In this work, the WUMS considers the traffic rate in the downlink direction, essentially assuming that the uplink traffic is, for all practical reasons, negligible comparing to downlink traffic
- Although WUA was developed in Java and can run on a platform independent engine as Java Runtime Environment (JRE), the command syntax from the wireless card utility is platform dependent. Hence, the WUMS was developed considering always the same operating system environment running on the user wireless device (Windows 7 32/64bits). Chapter 5 (Emulation Model) provides further description and technical details.

- The WUMS does not consider inactive users. Inactive users are wireless users that have been associated with an AP for a long time, but they are currently not sending/receiving traffic load.
- However, the WUMS considers active users. Active users are wireless users that have been associated with an AP, and they are currently generating traffic load. In this work, users are classified in the following manner:
    - Heavy users: users streaming video from a media server (UDP traffic). In this case, users with traffic rate greater than 2Mbps are considered heavy users. A predefined threshold of 2Mpbs was considered in the system design. This is due to the user traffic rate observed on users performing video streaming. An alternative (but much complex) way to detect heavy users might be by analyzing layer 3 and 4 information collected from the Openflow statistics of a user. In this way, the system would be able to find out the user traffic type. However, this approach was not implemented for the purpose of simplicity.
    - Light users: users generating web pages requests, emails, etc. (TCP traffic). In this case, users with traffic rate less than 2Mbps are considered light users. This is due to the user traffic rate observed on users performing audio streaming which is usually 200kbps in our scenarios.

3.2.2. Wireless Considerations

- IEEE 802.11n is utilized between APs and wireless users. No modifications to the IEEE 802.11 standard was introduced in our WUMS
- According to the IEEE 802.11 standard [1], the wireless networks in this work are categorized as infrastructure-based
- APs are configured with different SSIDs in order to emulate crowded scenarios (i.e.: coffee shops, hotspots, airports, etc) in which several networks are advertised
- APs operate in different wireless channels on the range of 2.4 Ghz. This is to avoid the issues that result when the wireless medium is crowded, and thus concentrate the study in the analysis of load redistribution among APs

- Wireless users are stationary, and they are randomly distributed in different locations within the APs radio frequency range, aiming at emulating real scenarios such as coffee shops, hotspots, airports and other crowded places. There is no user mobility model consider in this work.
- Each AP has one wireless interface where users associate to
- SSIDs are all visible to wireless users. There are no hidden SSIDs configured in the APs
- It is required that the BSSID of a given AP has the same ID as the Mac address of the AP WLAN interface. In this way, the WUMS can link the AP information coming from the SDN environment (AP Mac Address) with the one collected from the wireless environment (BSSID)

3.2.3. SDN Considerations

- The SDN controller must provide a northbound REST API interface for the WUMS to interact with the SDN network
- The SDN controller works in dynamic forwarding mode. This means that, based on the network status and traffic demands, the SDN controller install, remove and update flows on each SDN wired and wireless switch in an automatic manner.
- Openflow 1.3 is the SDN southbound protocol: This protocol is used between the controller and wired/wireless switches. No modifications were introduced to the protocol for the purpose of collecting wireless information from users.

3.3.Software Defined Wireless Network Controller

The Software Defined Wireless Network Controller (SDWNC) is the principal component of the WUMS. This application is in charge of monitoring the network, detecting the asymmetry and redistributing users associated to over-loaded APs to other under-loaded AP in the vicinity.

In general, the SDWNC receives wired and wireless network information via TCP connections either with the SDN controller or WUAs. The advantage of choosing this scheme is the flexibility towards various implementations. In fact, no modifications

are required in the wired and wireless network protocols and standards to provide the WUMS with the required network information. Particularly, the SDWNC contacts the SDN controller to get the topology information and traffic rate measurements from the APs and associated users. However, IEEE 802.11 network information is not supported by current open-source SDN controllers. For this reason, the SDWNC collects the current association information and the available wireless networks that are seen by the wireless user from the WUA directly.

A major part of the SDWNC is an algorithm that performs wireless users migrations. Although the details of the algorithm will be explained in the next chapter (User Migration Algorithm), the following describes all the tasks and control messages exchanged between the SDWNC, SDN controller and WUAs in order to obtain the network information required by the algorithm:

- **Topology information:** Usually, a SDN controller maintains an updated network topology database to make forwarding decisions. The SDWNC requires this source of information in order to perform all the user management operations. In order to get the required information, the SDWNC contacts the SDN controller via the northbound RESTCONF API interface. This interface supports HTTP operations such as OPTIONS, GET, PUT, POST and DELETE Moreover, the RESTCONF API allows access to the SDN controller database to configure and retrieve information [56]. Therefore, upon starting, the SDWNC sends a request to the SDN controller. The reply contains the topology database of the SDN controller, which provides the SDWNC the required data to retrieve contact and status information from the APs. Finally, the SDWNC builds an AP list to store Mac Address, IP addresses, Openflow IDs, and Openflow ports IDs from all the APs in the SDN topology. Figure 2 shows the messages exchanged between the SDWNC and SDN controller during this stage.

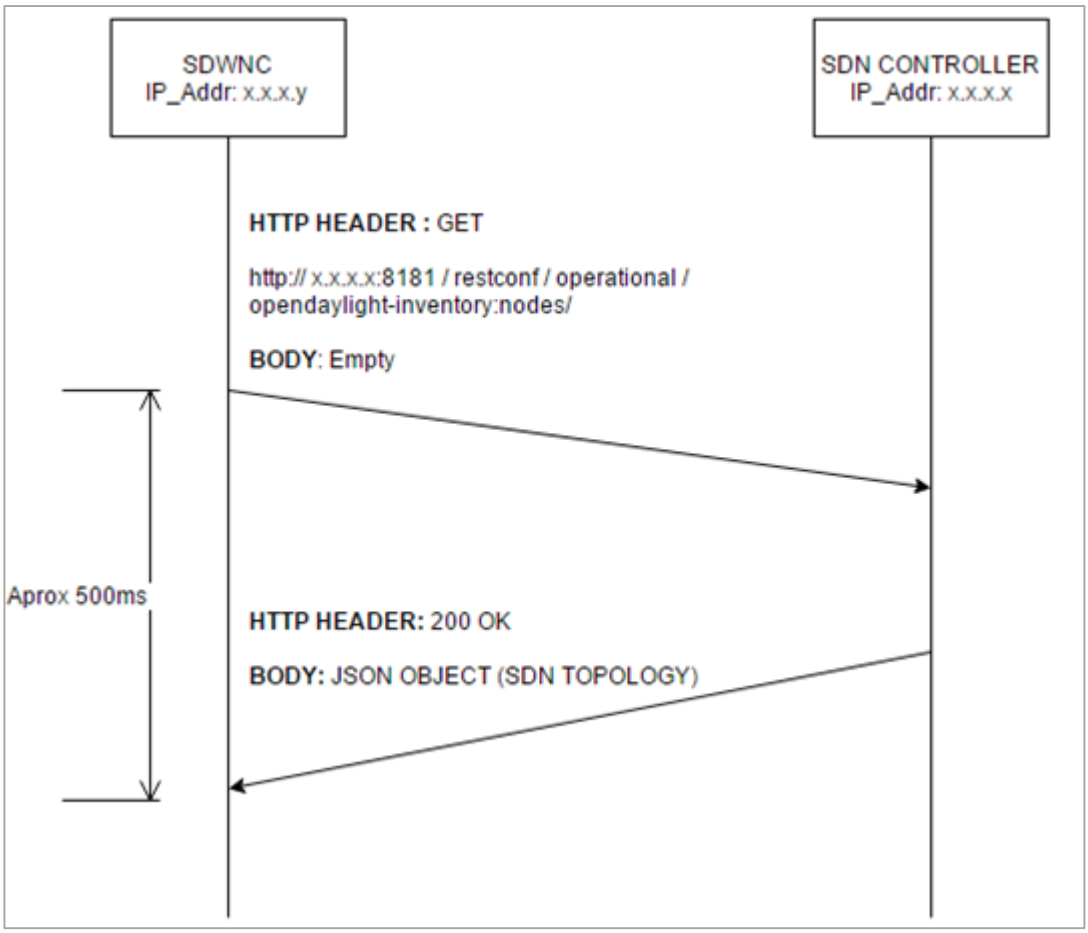

Figure 2. Message diagram of SDN topology information

The SDWNC sends a GET message to the SDN controller on port 8181 with a URI that ask for the SDN topology information. Notice that port 8181 is specific to the SDN controller used (Opendaylight) and may differ on other implementations. The SDN controller replies by sending the HTTP message 200 OK. The body of the message includes the SDN topology as a JSON object format. After receiving this information, The SDWNC filters and parses the JSON object, using the pre-defined "Javax.json" package.

- **Monitoring:** Once the SDWNC builds the AP list of the SDN topology, the monitoring process starts. During this stage, the SDWNC contacts the SDN controller to get the current traffic load information of the network. Essentially, the SDWNC gets Openflow port statistics of each AP of the topology. These statistics contains network status indicators that can be used to calculate the current traffic load on a given AP.

Figure 3 shows the messages exchanged between the SDWNC and SDN controller during this stage.

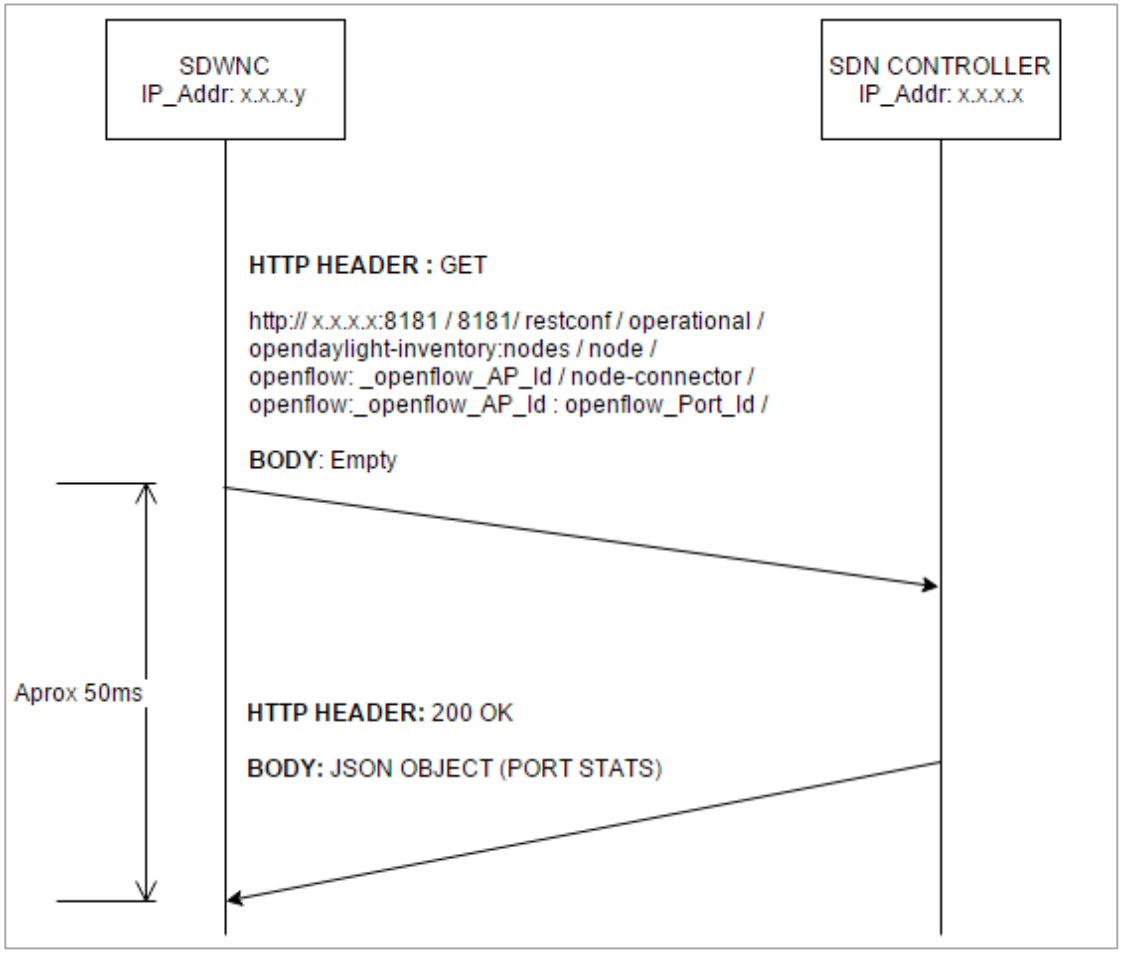

Figure 3. Message diagram of SDN Port Statistics for a given AP

The SDWNC sends a GET message, which URI specifies the Openflow ports statistics from a given AP of the topology. The SDN controller replies by sending the message 200 OK. The body of the message includes the Openflow ports statistics of the particular AP as a JSON object format.

For each AP in the topology, the SDWNC periodically carries out the message scheme shown in Figure 3. By doing so, the SDWNC may detect an imbalance condition in the network. If that is the case, the SDWNC compares the traffic load of each AP in order to identify the over-loaded one. Additionally, Openflow ports statistics contains contact information (IP and Mac address) of all the wireless users associated to the port. Next, user analysis is carried out on the over-loaded AP.

User Analysis: This stage identifies the users' load and session duration on the over-loaded AP. In order to get this information, the SDWNC queries the SDN controller for Openflow flow statistics related with the wireless users associated with the over-loaded AP. After taking a number of samples, the SDWNC identifies the flows that belong to each user. Figure 4 shows the messages exchanged between the SDWNC and SDN controller during this stage.

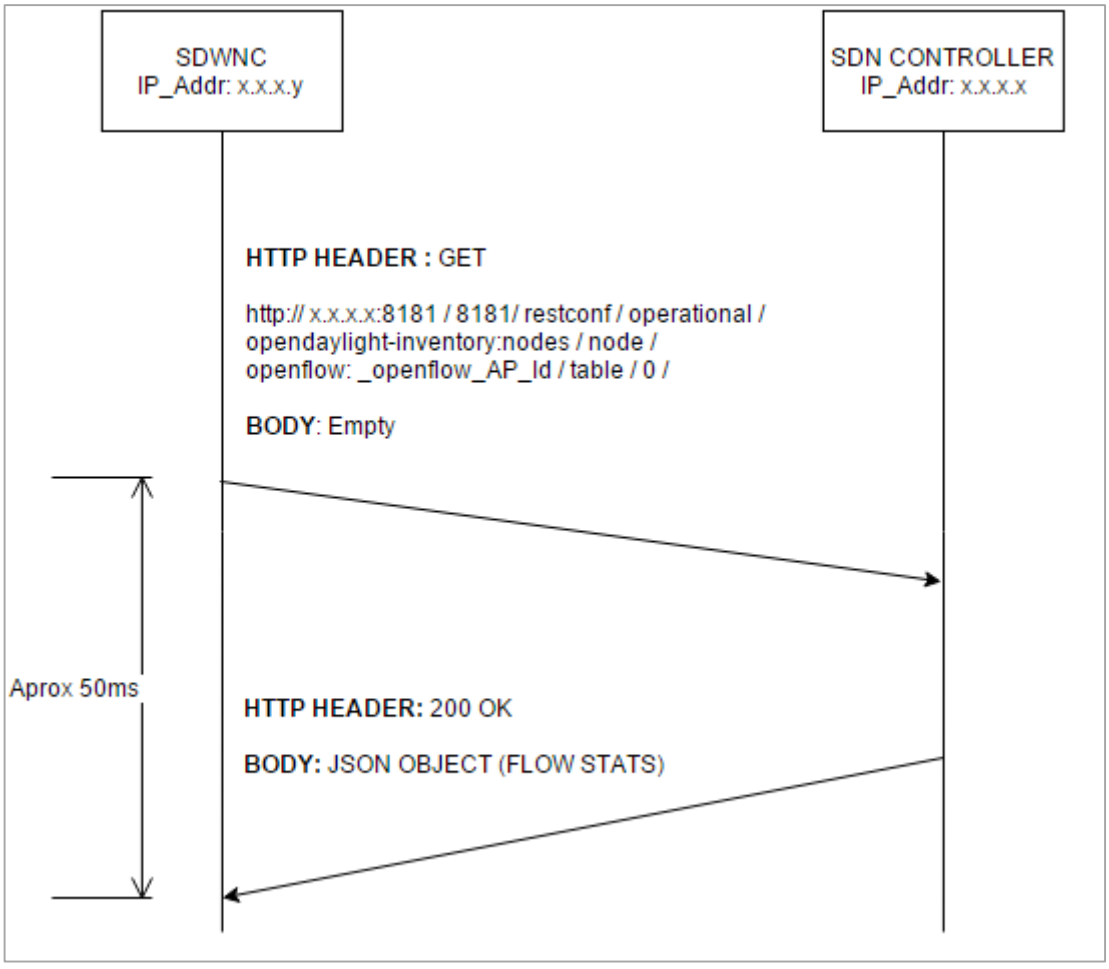

Figure 4. Message diagram of SDN Flow Statistics for a given user

The SDWNC sends a GET message, which URI specifies the Openflow flow statistics from a given user in the over-loaded AP. The SDN controller replies by sending the message 200 OK. The body of the message includes the

Openflow flow statistics of the particular user in the over-loaded AP as a JSON object format.

For each user in the over-loaded AP, the SDWNC carries out the message scheme shown in Figure 4. By doing so, the SDWNC gets the traffic load and session duration of each user. This user information serves as an input for the next stage

- **Decision-making:** In this stage, a set of wireless users are eligible for migration. The stage considers the average traffic load of all APs, as well as, the traffic load and session duration of each user in the over-loaded AP. As a result of this analysis, the SDWNC provides a sorted list of wireless users to migrate. The sorted list includes the IP and Mac address of the selected users for migration. Due to the fact that there is no additional information required to perform this analysis, there are no control messages exchanged. A detailed description of this part is provided in the next chapter (User Migration Algorithm).

- **Contacting wireless users:** Using the user list from the previous stage, the SDWNC contact the WUA on each user. This is to retrieve information of the wireless environment from each user. The information exchanged is the current association and the available networks seen by the user. Figure 5 shows the messages exchanged between the SDWNC and the WUA (of a particular user) during this stage.

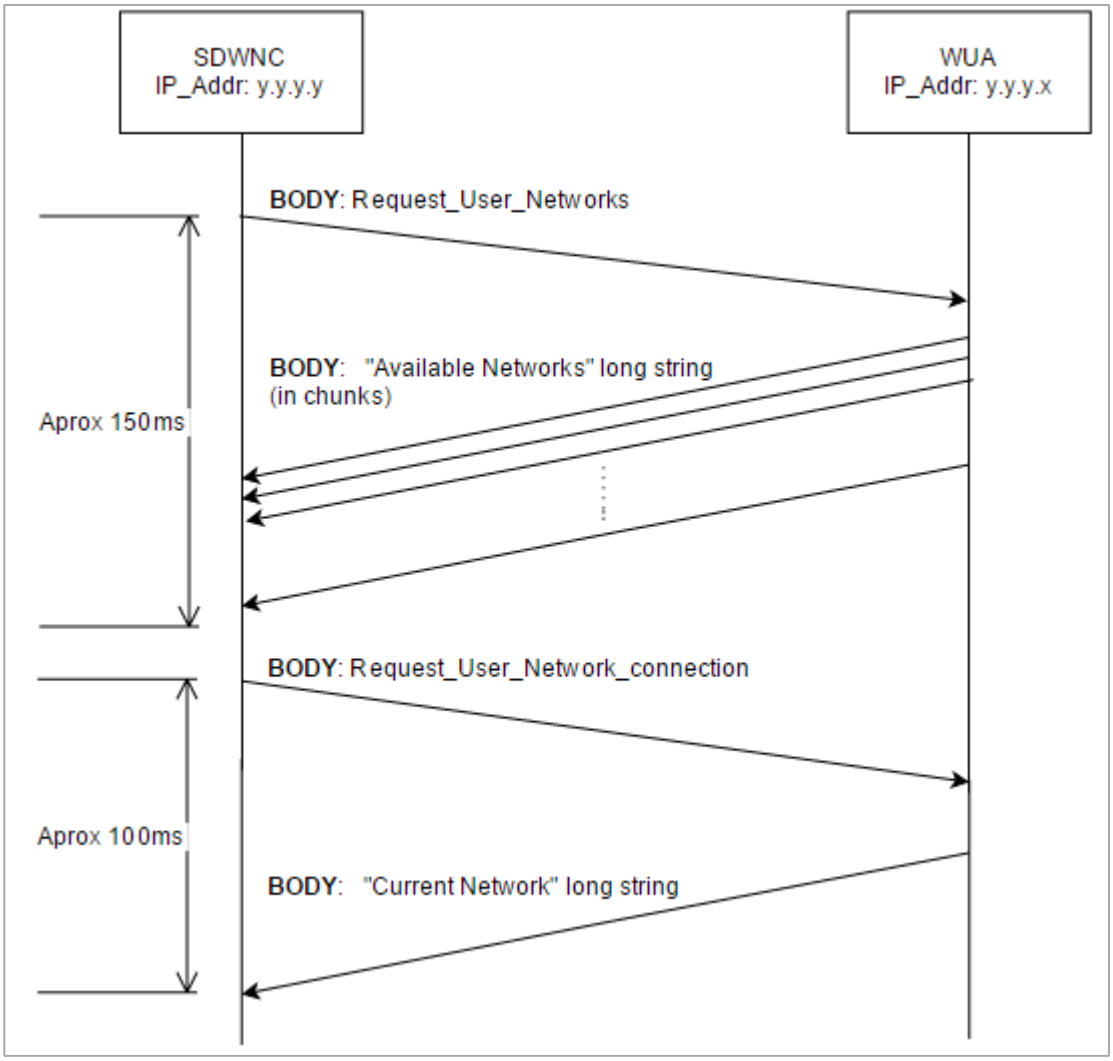

Figure 5. Message diagram of Available Networks and Current Network information

The SDWNC sends a request with the message “Request_User_Networks” in order to get the available networks seen within the RF user range, and a separate request with the message “Request_User_Network_connection” to collect the current association details of the wireless user.

In the case of the available networks information, the WUA send the long response in multiple chunks. In fact, depending on the number of available networks seen by the user, the response is broken in a few or several chunks.

- **Data processing from the user:** The SDWNC processes the responses received from the WUA in order to obtain the following information: SSID, AP MAC ADDR (BSSID), signal strength, and wireless channel. Due to the

fact that there is no additional information required to perform these tasks, there are no control messages exchanged in this stage.

- **User migration and confirmation:** The WUMS assists the wireless users during migration. In this process, the SDWNC open a TCP connection with the WUA to send the target AP authentication information. Once the authentication information are sent to the WUA, the SDWNC wait for 10 *secs* to contact the WUA again. Hence, the SDWNC verifies that the wireless user is currently associated to the target AP. The waiting gap configured in the SDWNC (10 *secs*), provides the WUA with enough time to:

    - execute the command to switch the user device to the target AP
    - wait for the completion of the re-association and authentication process
    - wait until wireless connectivity resumes (in the target AP)
    - and execute the "current networks" command that will returned the information of the target AP

  It was observed during tests that setting the waiting gap to values less than 10 *secs*, makes the wireless card utility (in the user device) halt or work erratically (when the WUA executes the current networks command after migration). However, as it will be shown in section 6.2 (Results), the actual disconnection time for the user connectivity occurs independently of this issue, and it takes considerable less time than the waiting gap mentioned here.

  Figure 6 shows the messages exchanged between the SDWNC and the WUA (of a particular user) during this stage. The SDWNC sends a "Request_User_Network_connection" message to the WUA. The SDWNC processes the response received from the WUA and gets the SSID in which the user is currently associate to. Lastly, the SDWNC checks the SSID received in the response with the SSID sent previously to the WUA (on the target AP authentication information message). In this way, the SDWNC verifies whether the migration of successful or not.

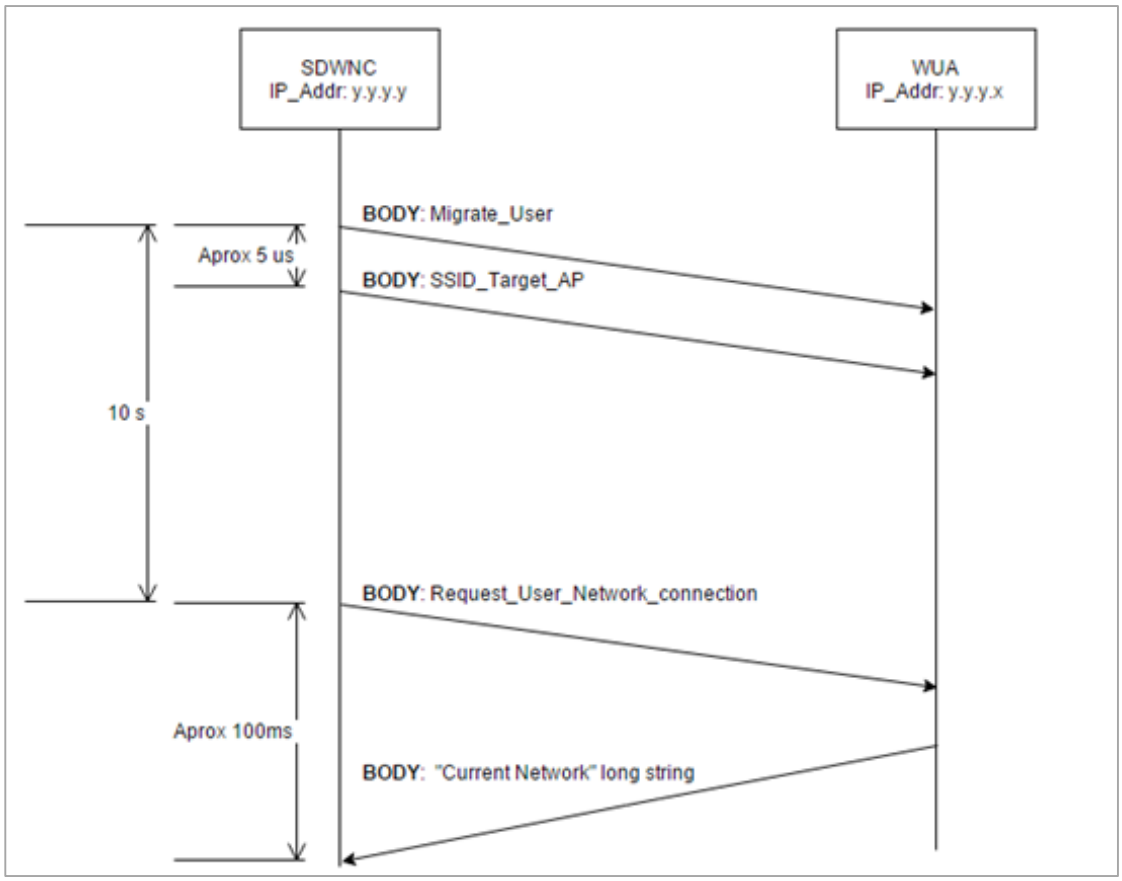

Figure 6. Message diagram of user migration and confirmation after migration

In regards to the software design, the SDWNC was developed in several modules all linked to the main program. This modular approach is convenient for code debugging, as well as, it provides the flexibility required to apply future changes in the WUMS. Regardless of the Java language definition of classes, the SDWNC software design considers three types of elements in terms of functionality. These are: classes' declaration and definition, external functions and the main program. Below there is a description of the modules:

- **SDWNC_Main1:** This is the main program that runs the SDWNC code. It performs all the tasks described previously in this section. The main program calls external functions, gets/sets data from/to Java object, and executes the algorithm code to perform user migrations. Moreover, it provides an output to the standard console with runtime information of the tasks being executed at the moment.

- **Modules that start with the keyword *Class_*:** All these modules declare and define classes that provide attributes and methods for the external functions and the main program. These modules provide methods to get and set the input data collected from the SDN network and wireless users.

- **REST_API_Connection:** This module is an external function that is in charge of sending and receiving REST_API messages. All messages between the SDWNC and the SDN Controller of the SDWNC stages that were explained before, are processed by this module.

- **SDN_Topology:** This external function is in charge of retrieving the topology from the SDN controller. However, this modules requires the REST_API_Connection module in order to send out the request to the SDN controller. Later, the topology data received is filtered in order to get the contact information of the AP in the SDN network such as the APs Mac Address and APs Openflow IDs and ports.

- **FlowStatistics and PortStatistics:** Both are external functions that manage the collection of samples from the SDN Controller. They store the SDN statistics as Java objects. Later, the main program gets specific status information from the SDN network, by calling these Java objects. In order to get flow and port statistics samples from the SDN controller, these modules requires calling the REST_API_Connection module

- **Network_Classification:** This module is an external function that is used by the main program. The module classifies APs according to the current traffic load (traffic rate) supported by the AP. This task is always required whenever an unbalanced condition is detected. In this case, the main program calls the Network_Classification function to mark each AP as OVER-LOADED, BALANCED and CANDIDATE by comparing the AP load with the total AP average load.

- **TCP_Connection_WUA:** This module is an external function that establishes TCP connections with the WUA application. The main program calls this external function whenever it requires wireless information from a given user

- **TCP_Connection_OVS:** This module is an external function that establishes a TCP connection with an application that is running on a particular SDN switch of the environment. The reason behind this is a technical issue related to the flows updates during user migrations. The issue will be explained in chapter 5 (Emulation Model).

Figure 7 presents the UML representation of the SDWNC software modules and their dependencies. In addition, the Figure 7 details the methods available on each module

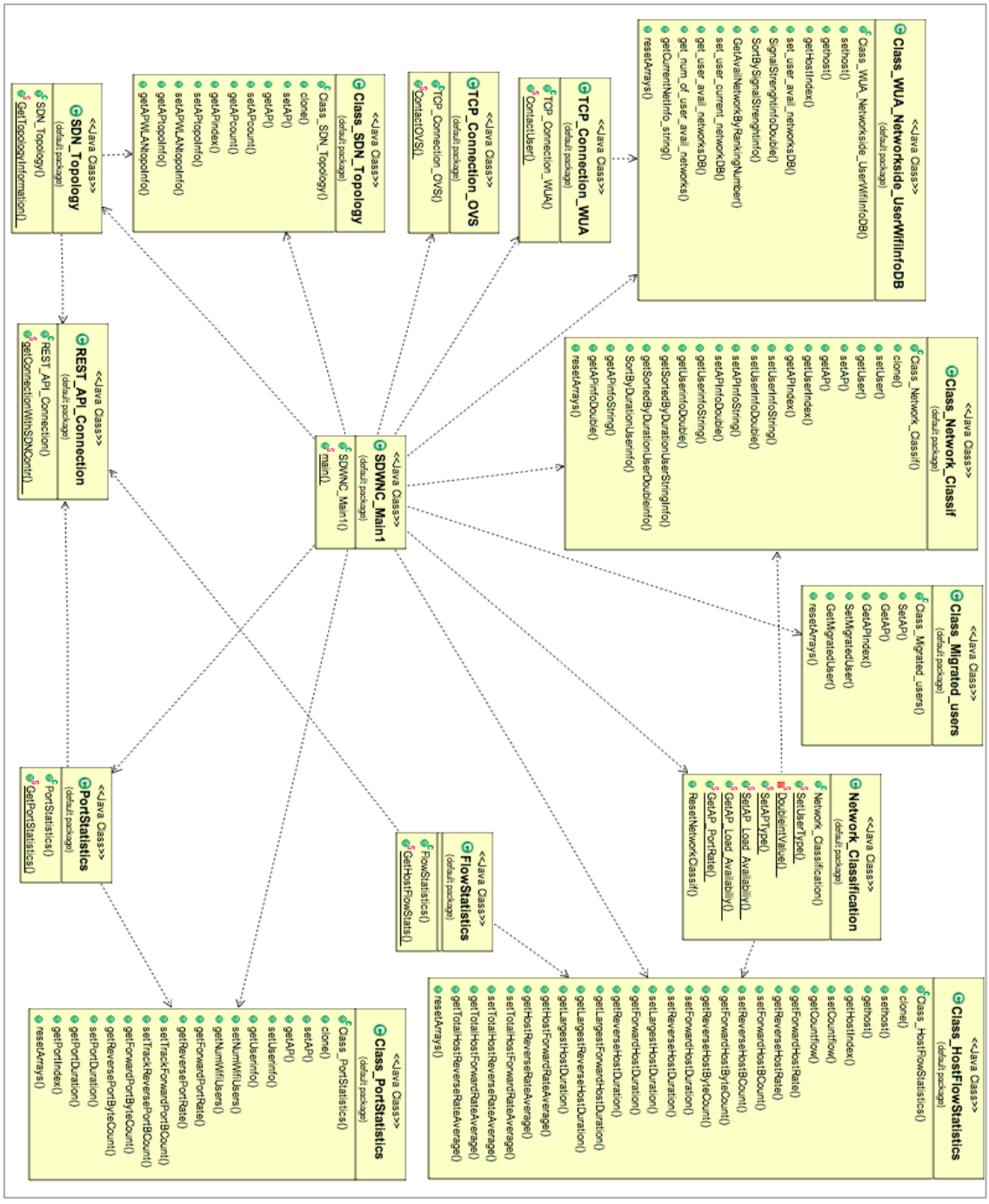

Figure 7. UML representation of the SDWNC software

3.4. Wireless User Application

The Wireless User Application (WUA) is part of the WUMS and runs on every wireless device. It makes the WUMS aware of the wireless environment and provides support to the SDWNC during user migrations. The WUA is a lightweight Java-based software. Like any Java application, it requires the Java Runtime Environment (JRE) engine to work. The WUA performs its actions upon request. It listens for incoming requests on a static TCP port 10007, which is hardcoded in the software. As shown previously in the message diagrams of Figure 5 and Figure 6, connections are always initialized by the SDWNC. Every time the WUA received a message request from the SDWNC, it executes commands from the wireless card utility of the user device.

In Figure 8, a wireless device is running the WUA application on a Windows OS. Figure 8 shows an example of the command syntax (at the top) and the output expected (at the top).

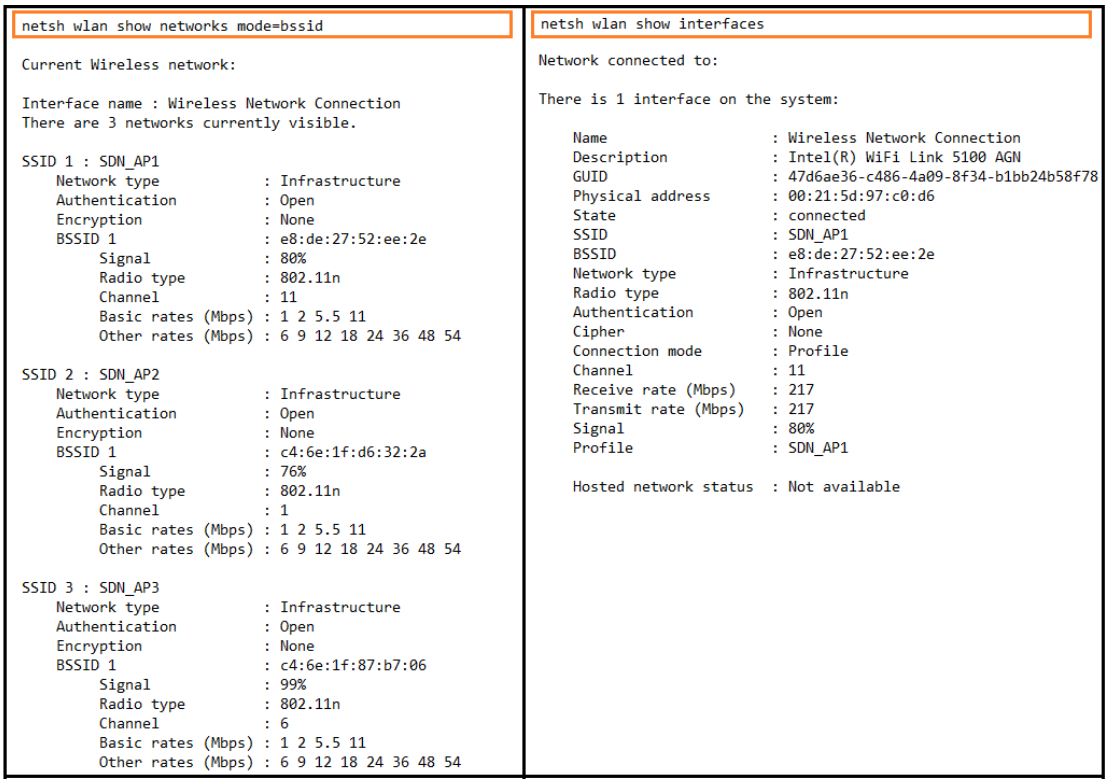

Figure 8. Available networks and current network information from WUA

When the SDWNC sends out the message containing the string "Request_User_Networks", the WUA returns the command and output shown on the left of Figure 8. However, when the SDWNC sends out a request containing the string "Request_User_Network_connection", the WUA returns the command and output shown on the right. By doing so, the WUA (of a particular wireless user) provides the SDWNC with the available networks information and the current association details respectively. Upon receiving these replies from the WUA, the SDWNC uses that information in order to select the target AP for a particular user.

Besides, the WUA plays an important role during user migrations. As explained in the previous section, once the WUMS determines the target AP for a given user, it provides assistance to that user during the migration. In particular, when the WUA receives the authentication information to re-associate to another AP, it performs the user migration. This authentication information may include the target SSID and password for authentication and encrypted connections. However, due to a technical implementation issue found in the APs configuration this study considers an open authentication method. The details of this issue will be explained in chapter 5 (Emulation Model). Next, the SDWNC sends out the SSID of the target AP as the only authentication information required for re-association. Once the WUA switches the wireless user to the new AP, the last task for the WUMS is to verify the success of the migration. For this reason, ten seconds after migration, the WUA receives a request regarding the current association details of the wireless user ("Request_User_Network_connection" message). Consequently, the WUA executes the corresponding command internally and sends the output to the SDWNC. On the other side of the communication, the SDWNC gets the new SSID from the message, and use it to verify that the user was successfully migrated to the target AP. The messages exchanged during this process are shown in Figure 6.

The assisted method provided by the WUMS does not rely on the traditional IEEE 802.11 approach, in which the user device is totally in charge of the re-association process. Although, the assisted method still introduces a minor connection disruption when the WUA moves a user to another AP, it is considerably smaller in comparison with the traditional approach. This is because the scanning process and re-association attempts, that are traditionally done when a user dissociate to a given AP, are avoided during the disassociation and re-association period.

4. User Migration Algorithm

The User Migration Algorithm (UMA) perform user migration in order to distribute the traffic load among APs, and thus improve the user QoS in an integrated wired/wireless SDN environment. The UMA resides in the SDWNC. It was designed as a centralized approach that works in a reactive manner against unbalanced load conditions in the network. The user migration decisions are based on the information gathered from both the wired and wireless SDN networks. In particular, the UMA identifies the load in the network by measuring the traffic rate and session duration of all flows that each AP is supporting at the moment. In this way, when the current load of a given AP is greater than the average load of all APs, the UMA is capable of detecting the asymmetry in the network. In these cases, the UMA makes decisions of which users to migrate, and where each particular user should be moved. Consequently, it re-distributes the load to other APs with vacant capacity.
Moreover, the UMA was developed under the following set of assumptions:

- In order for the UMA to perform user management operations, traffic rate values of WLAN ports must be above a certain pre-defined threshold. This is to avoid the activation of the UMA when the traffic load is negligible for the network
- The number of wireless users plus the number APs, as well as, the total load in the network are such that, a redistribution of the user load among APs, can bring the WUMS into to balance. For instance, a scenario with two wireless users connected to the same AP, where one user is generating heavy load and the other has light load, will never bring the WUMS into equilibrium. This is because there is not enough light users to switch to other APs, in order to compensate the load of the single heavy user, and thus balance the network
- The WUMS considers that the network is unbalanced when a formula (fairness formula) returns a value 10% below 1 (unbalanced < 0.9). The details of the formula and the considerations taken will be explained later in this section

- Whenever an unbalanced condition occurs, the UMA analyze one over-loaded AP at a time
- There must be sufficient capacity available on nearby APs, in order to absorb the exceeded traffic load. Otherwise, balancing the load will not result in a positive effect on the wireless users' QoS

The term U_h is defined as the average traffic rate of the user in the downlink direction. Although the WUMS gets information from both directions (uplink and downlink), the study is conducted to analyze the U_h in the downlink direction.

As explained in the previous chapter, the WUMS collects real-time information from the SDN controller. This information serves as the input metrics to select users for migration. These metrics are the U_h and the *user session duration.* The former provides a direct way of measuring the load contribution of a user in the wired/wireless network. In this way, the UMA can differentiate the load generated by each wireless user on the over-loaded AP. However, the latter was considered to provide a second criteria for migrating decision. This metric does not serve as an extra indicator of the traffic load issue. In fact, the *user session duration* provides information about wireless user behavior, such as it was defined and studied in the literature [48] and [49]. Thereby, the UMA considers wireless users with recent sessions (new users) as eligible for migration, avoiding the connection disruptions on wireless users with long time session (old users).

Lastly, the UMA utilizes wireless information from the selected users for migration. This is to select the target AP where the user should be move to. Basically, in order to perform the task, the UMA requires the SSID and signal strength information of nearby APs from the wireless user.

Figure 9 depicts all the processes performed by the algorithm

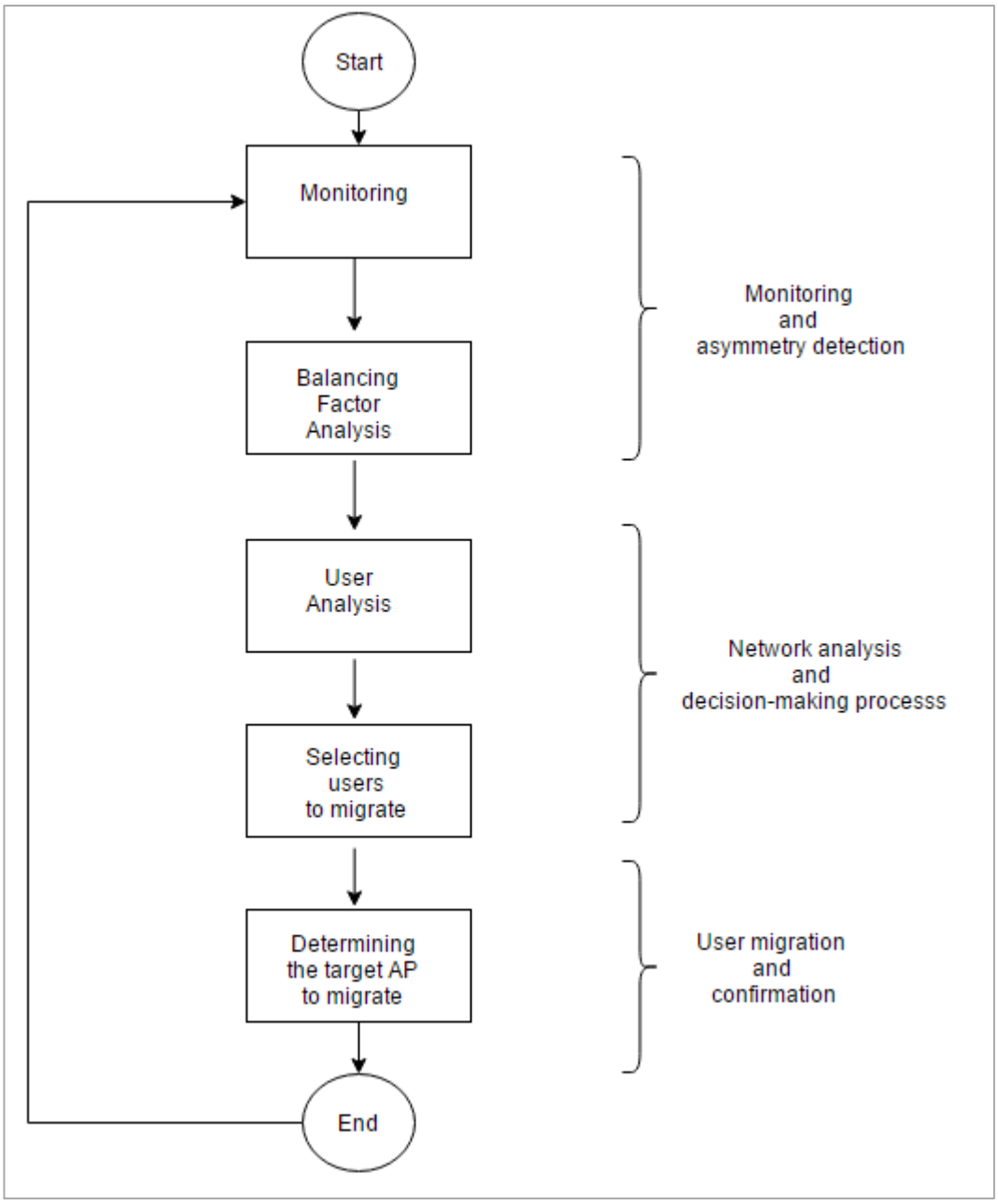

Figure 9. Process chart of the UMA

The remaining part of this chapter describes the each stage of the UMA:

Monitoring: With the topology information previously obtained from the SDN controller, the SDWNC build the AP list and the monitoring process starts. This process goes through the entire AP list in a cyclical manner, by querying the SDN controller for the Openflow port statistics. These statistics provide information regarding the number of associated users and their addresses (MAC and IP address), as well as, real-time information such as the inbound/outbound byte counters and

duration on the port. Following, we describe the logic of this stage. For each AP in the SDN topology (*from i=1 to i=I*), the UMA gets the port statistics. By reading the byte counters and duration, the UMA calculates the instant traffic rate on the WLAN interface ($P_{i,s=1}$) and stores it in its database. Due to the dynamic changes of traffic rates in the network, taking one sample of an instant value may lead to incorrect asymmetry detection. Therefore, the UMA goes through the AP list three times, calculating and storing the instant port rate sample values ($P_{i,s=1}$, $P_{i,s=2}$ *and* $P_{i,s=3}$). At the end of the third round, the UMA calculates the port rate of each AP, by taking the average of the three samples, which is denoted as P_i. The following equation summarize all the calculation performed to get the port rate of an AP (*i)*:

$$P_i = \frac{\left(\sum_{s=1}^{s=S} P_{i,s} \right)}{S} \qquad (1)$$

Where:

S: Total number of port statistics samples for a given AP. In this thesis, S=3

P_i: Average port rate of a given AP *i*

Figure 10 shows the flow chart of this process

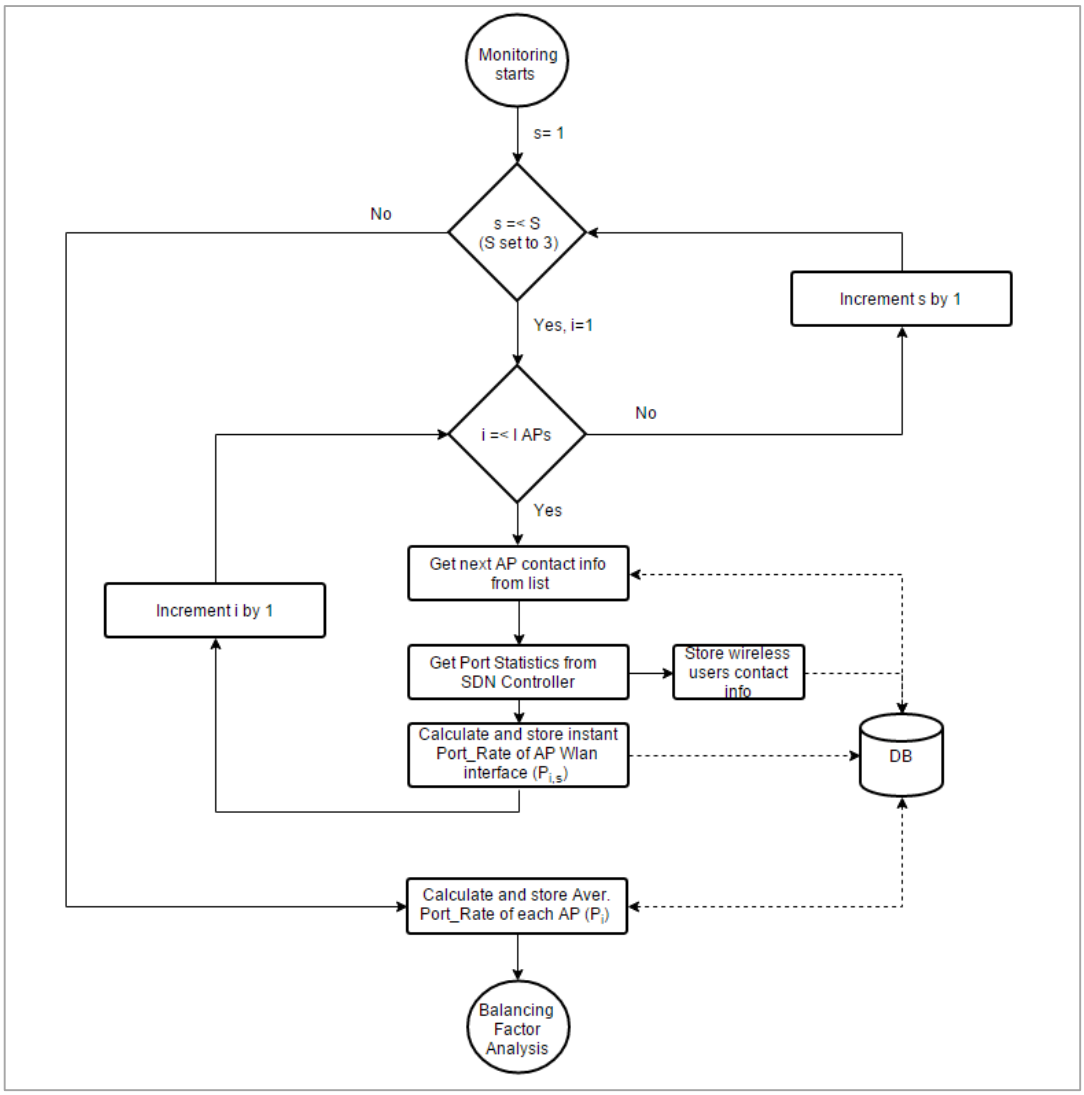

Figure 10. Flow chart of the Monitoring process

Finally, in order to study the behaviour of the monitoring process, a complexity analysis is provided. In Figure 10, we observe that the process requires to iterate with the number of samples (S) and number APs (I). This can be expressed using Big O notation as O(S*I). However, we can see that, regardless of the number of APs (I), the process always performs a constant number of iterations, given by the number of samples (S). Therefore, the entire monitoring process can be considered linear O(I).

Balancing factor analysis: Next, the UMA evaluates whether the network is balanced or unbalanced. In order to achieve this task, the UMA uses the previous port rate information of all APs as the input values for the fairness formula, first introduced in [38], and shown below:

$$\beta = \frac{(\sum_{i=1}^{i=I} P_i)^2}{(I * \sum_{i=1}^{i=I} P_i^2)} \qquad (2)$$

Where:

I: Total number of AP found in the SDN topology. In this thesis, I=3

P_i: Average port rate of each AP

Ideally, the formula (2) returns a value that indicates the degree of imbalance in the network. In particular, if the balancing factor value is closer to $\frac{1}{I}$ the more imbalance the network will become. On the other hand, balancing factor values closer to 1 indicates that the network is reaching a balanced state. However, the real implementation of this formula required to shift the decision threshold of imbalance to a value slightly below 1. In fact, it was observed during the tests that, in a real balanced network, the value returned by the fairness formula (2) is closer to 1, but it is never equal to 1. Hence, setting the decision threshold of imbalance to 1, made the UMA detect load asymmetry at all times, even when there was no real need for user management in the network. On the contrary, setting the decision threshold of imbalance to values below 0.9 allowed long periods of asymmetry, making the UMA less efficient. Finally, the decision threshold of imbalance was set to 0.9, which is 10% below 1 (the ideal condition).

Thereby, when the balancing factor value returned by the formula is below 0.9, we consider the network unbalanced. In that case, the next step is to determine which AP/s are causing the asymmetry. In order to do that, the UMA computes the average port rate of all the APs in the network ($\bar{L}$) as:

$$\bar{L} = \frac{\sum_{i=1}^{i=I} P_i}{I} \qquad (3)$$

Where:

I: Total number of AP found in the SDN topology. In this thesis, I=3

P_i: Average port rate of each AP

$\bar{L}$: Average port rate of all APs in the SDN topology

In this way, if P_i exceeds $\bar{L}$, the AP is marked as OVER-LOADED. On the other hand, if the P_i equals $\bar{L}$, the AP is marked as BALANCED. Otherwise, when P_i is below $\bar{L}$ the AP is marked as CANDIDATE. This last case means that the AP is capable to absorb the additional traffic load. The flow chart of this process is shown in Figure 11.

Finally, in order to study the behaviour of this process, a complexity analysis is provided. In Figure 11, we observe that in the case when the beta is below 0.9, the process requires a loop to repeat the tasks for every AP, until the process reaches the number of APs (I). Moreover, each operation inside the loop, is processed only once, depending whether the given AP is considered CANDIDATE, BALANCED or OVER-LOADED. Therefore, the entire process can be considered linear O(I).

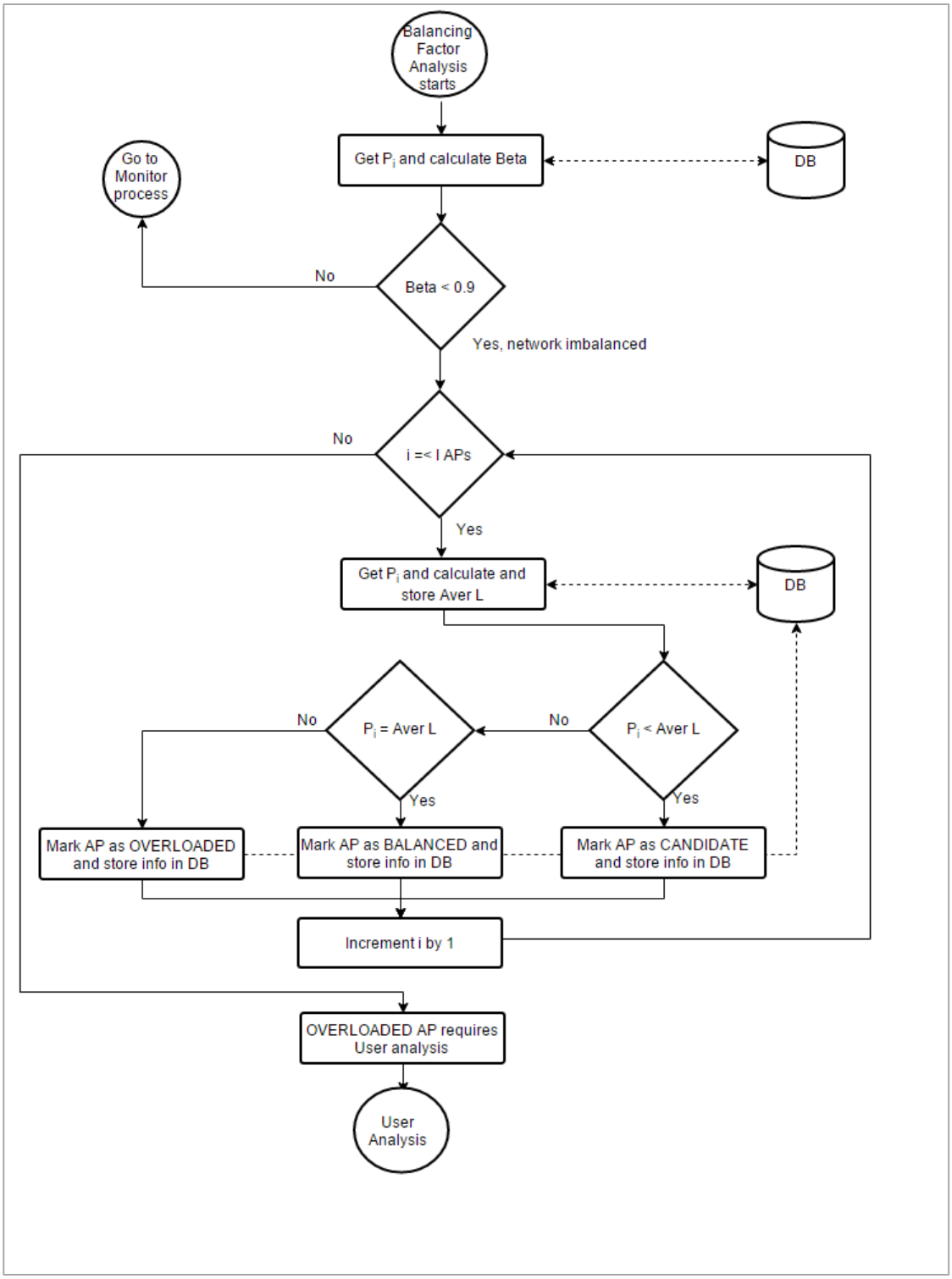

Figure 11. Flow chart of the Balancing Factor Analysis

User Analysis: Once the over-loaded AP is detected, the UMA performs an analysis of on every associated user. This does this until it reaches the total number of users in the over-loaded AP (*from h=1 to h=H*). As mentioned in the previous chapter, this process requires the SDWNC to collect the Openflow flow statistics of the over-loaded AP.

Following, we describe the logic of this analysis. For each user in the over-loaded AP (*from h=1 to h=H*), the UMA gets all flow statistics. Usually, Openflow statistics of a given flow contains the byte count and duration. With this information, the UMA calculates the traffic rate of one flow denoted as $U_{h,k}$. After calculating the traffic rate of each flow, the UMA adds all user traffic rates together (in the downlink direction). This is to get the total traffic rate for that particular user which is denoted as $U_{h,m}$. Occasionally, the collection of Openflow statistics returns inconsistent measurements. Therefore, the UMA requires at least three samples of flow statistics for each user. Next, the UMA calculates the average of the three samples in order to get the traffic rate of a given user (U_h). The following equations summarize all the calculation performed to get the traffic rate of a user (*h)*:

$$U_{h,m} = \sum_{k=1}^{k=K} U_{h,k} \qquad (4)$$

$$U_h = \frac{\left(\sum_{m=1}^{m=M} U_{h,m}\right)}{M} \qquad (5)$$

Where

H: Total number of users in the over-loaded AP

K: Total number of flows for a given user in the downlink direction

M: Total number of flow statistics samples for a given user. In this thesis M= 3

Lastly, in order to get the session duration for a particular user, the UMA determines the largest flow duration encountered among all the flows.

In Figure 12, a flow chart of the user analysis process is provided.

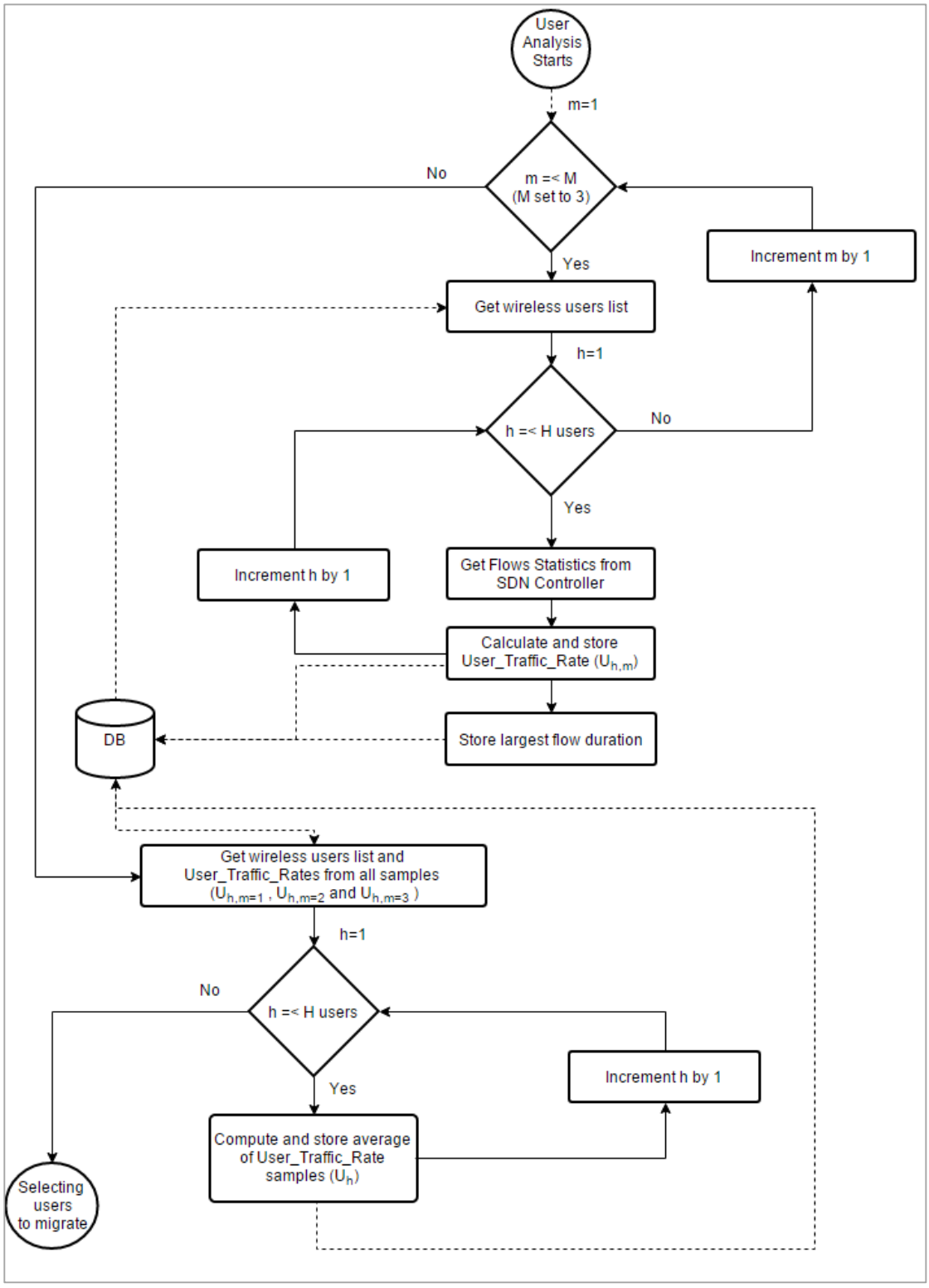

Figure 12. Flow chart of the User Analysis

Finally, in order to study the behaviour of this process, a complexity analysis is provided. In Figure 12, we observe that the process requires to iterate with the number

of samples (S), number of users (H), and finally with the number of users (H) again. This can be expressed using Big O notation as O(S*H + H). However, we can see that, regardless of the number of users (H), the process always performs a constant number of iterations, given by the number of samples (S). Therefore, the entire monitoring process can be considered O(H+H) which is the same as O(2H). Due to the fact that we are always interested in the performance of the process as H becomes large, the entire process can be considered linear O(H).

At this point, the analysis returns the user traffic information from the SDN network perspective, which serves as part of the input data required during the decision-making process of user migrations. The next step is to select users for migration.

Selecting users to migrate: After performing a user analysis, the UMA needs to determine which wireless users, associated to the over-loaded AP, will migrate (either heavy or light users). For this task, the UMA considers users that have recent sessions (new users) as eligible for migration. This is in favour of avoiding connection disruption on users with long time session (old users). According to this criterion, the UMA organized users on the list by the *user session duration*. In this way, the users list is built and sorted by the *user session duration* in ascending order. As shown in Table 1, the users list includes the user id (MacAddress), the traffic rate (U_h), and *user session duration*.

Table 1. Example of sorted users list

User ID	Session Duration [sec]	Traffic rate [Mbps]
MAC_ADDR_USER_1	User_Session_Dur_1	User_Traffic_Rate_1
MAC_ADDR_USER_2	User_Session_Dur_2	User_Traffic_Rate_2
:	:	:
:	:	:
MAC_ADDR_USER_N	User_Session_Dur_N	User_Traffic_Rate_N

Where

$$User\_Session\_Dur\_1 < User\_Session\_Dur\_2 < \ldots < User\_Session\_Dur\_N$$

Now, the UMA utilizes users list and selects users starting from the top. For every user selected, the user load (U_h) is added in a variable denoted as *UAL*, which serves as an accumulator of the users load. In fact, *UAL* is used to determine the amount of

load that the UMA withdraws from the over-loaded AP. The UMA mitigates the asymmetry by reducing the exceeded load to values closer to $\bar{L}$. In this way, only a certain amount of the traffic is moved to other APs, while the rest of it remains in the AP. The following condition controls the amount of user load that the UMA will consider to migrate:

$$\left(P_{i(overloadedAP)} - UAL\right) > \bar{L}$$

If the condition is true, the over-loaded AP is still above ($\bar{L}$). Hence, the UMA confirms the current user for migration and picks the next user to evaluate. However, when the condition returns false, adding the current user load drops the AP port rate (P_i) below $\bar{L}$. In this case, the UMA rejects the current user for migration, updates the *UAL* variable to the previous state (before adding the current user load U_h), and goes to the next process. Next, Figure 13 shows the flow chart of this process

Finally, in order to study the behaviour of this process, a complexity analysis is provided. Though it is not indicated in Figure 13, the wireless users list is an array, in which a linear search method is utilized for sorting. Thus, this sorting operation, that is performed H times, results in a linear complexity O(H). Moreover, in Figure 13, we observe that the process uses a loop that, in the worst case scenario, it would require to iterate a number of times closer to H. Moreover, each operation inside the loop, is processed only once, so they can be considered constant. Therefore, the complexity of the entire process is O(H + H), which is the same as O(2H). Due to the fact that we are always interested in the performance of the process as H becomes large, the entire process can be considered linear O(H).

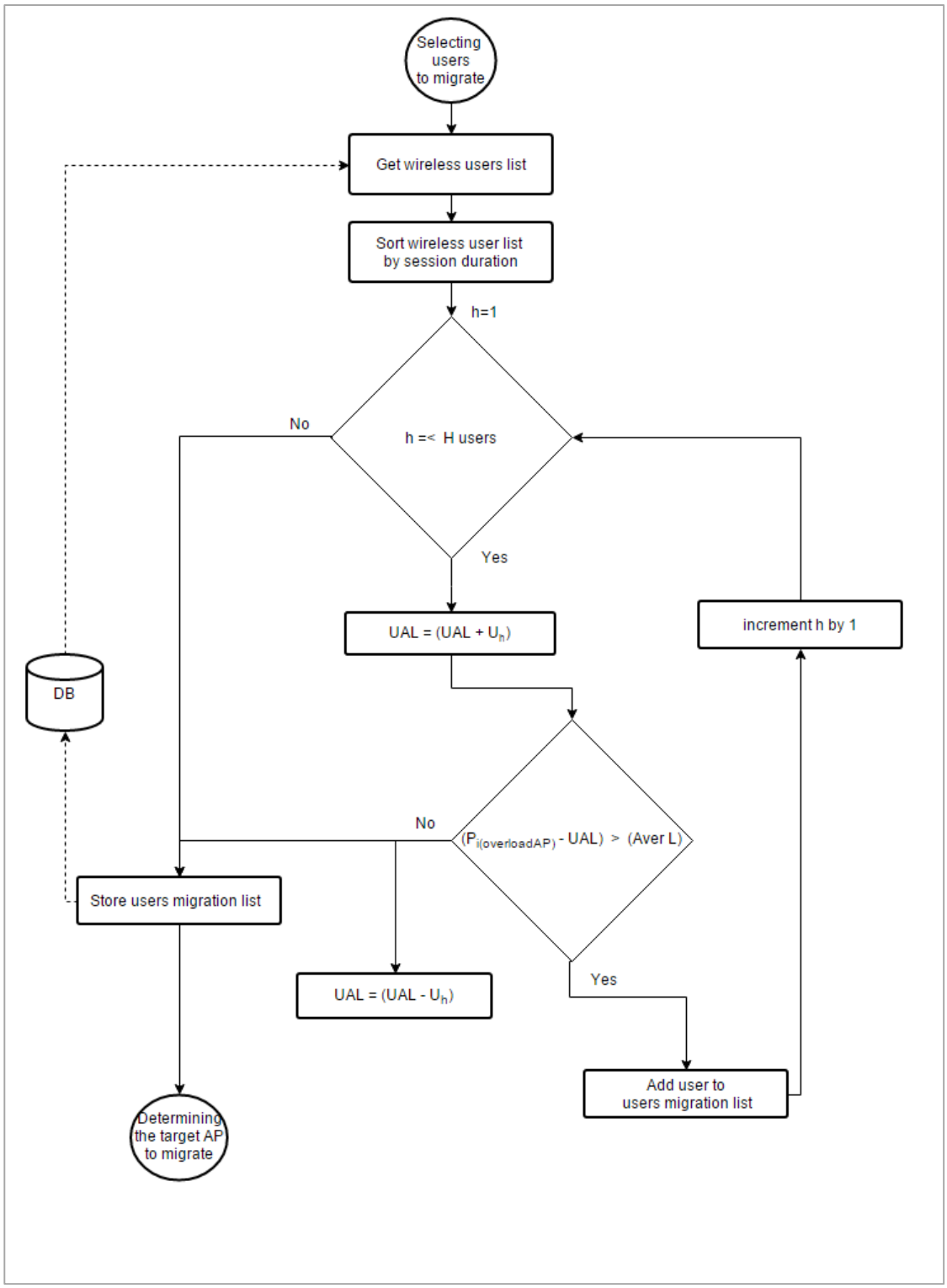

Figure 13. Flow chart of the Selecting users to migrate process

At this point, the UMA returns a list of the selected users for migration, whose accumulated load (*UAL*) will reduce the AP excess between $\bar{L}$ and the current AP port rate $\left(P_{i(overloadedAP)}\right)$.

Determining the target AP for each user: With the user migration list previously created, the UMA starts the process of moving users to under-loaded APs in the vicinity.

First, the UMA selects the target AP for each user. As explained in the previous chapter, the SDWNC contacts the user to acquire the current wireless association details and the available networks information. From the available networks information, the UMA filters APs by BSSID, which is the WLAN interface Mac address. Thereby, it gets the AP list of the ones that belong to the SDN network. The remaining APs in the list are filtered by the tag "CANDIDATE" (marked during the Balancing Factor Analysis), which identifies the under-loaded APs. Finally, the last filter selects the target AP based on the AP with the strongest signal strength. In this way, the SDWNC selects candidate APs based on the current AP traffic load and later, from the remaining ones, it chooses the target AP with best signal strength.

At this point, the UMA has selected the target AP for the user. The remaining part of the process provides assistance to users during migration. As explained in the previous chapter, the SDWNC sends the target AP authentication information to the WUA. After ten seconds, the UMA verifies that the migration was successful, by comparing the current SSID (sent by the WUA) string with SSID string that belongs to the target AP. The UMA returns to the monitoring process when all users, from the migration list, have been successfully moved.

Besides, it is important to highlight that the UMA always guarantee an even distribution of the network load. Particularly, the UMA considers the initial load on a "CANDIDATE" AP, as well as, the load of the users previously aggregated (if that were the case).

Finally, in order to study the behaviour of this process, a complexity analysis is provided. In **Error! Reference source not found.**, we observe that the process uses a oop that iterates H number of times. Moreover, the first operation inside the loop requires to get the list of available networks seen by the user. Though it is not indicated in **Error! Reference source not found.**, the number of available networks een by a given user can be expressed as A. In the worst case scenario, that the list of available networks cannot be reduced in size, the filtering operations that follows will be performed A number of times. Therefore, the complexity of the entire process ends up being O(H*A).

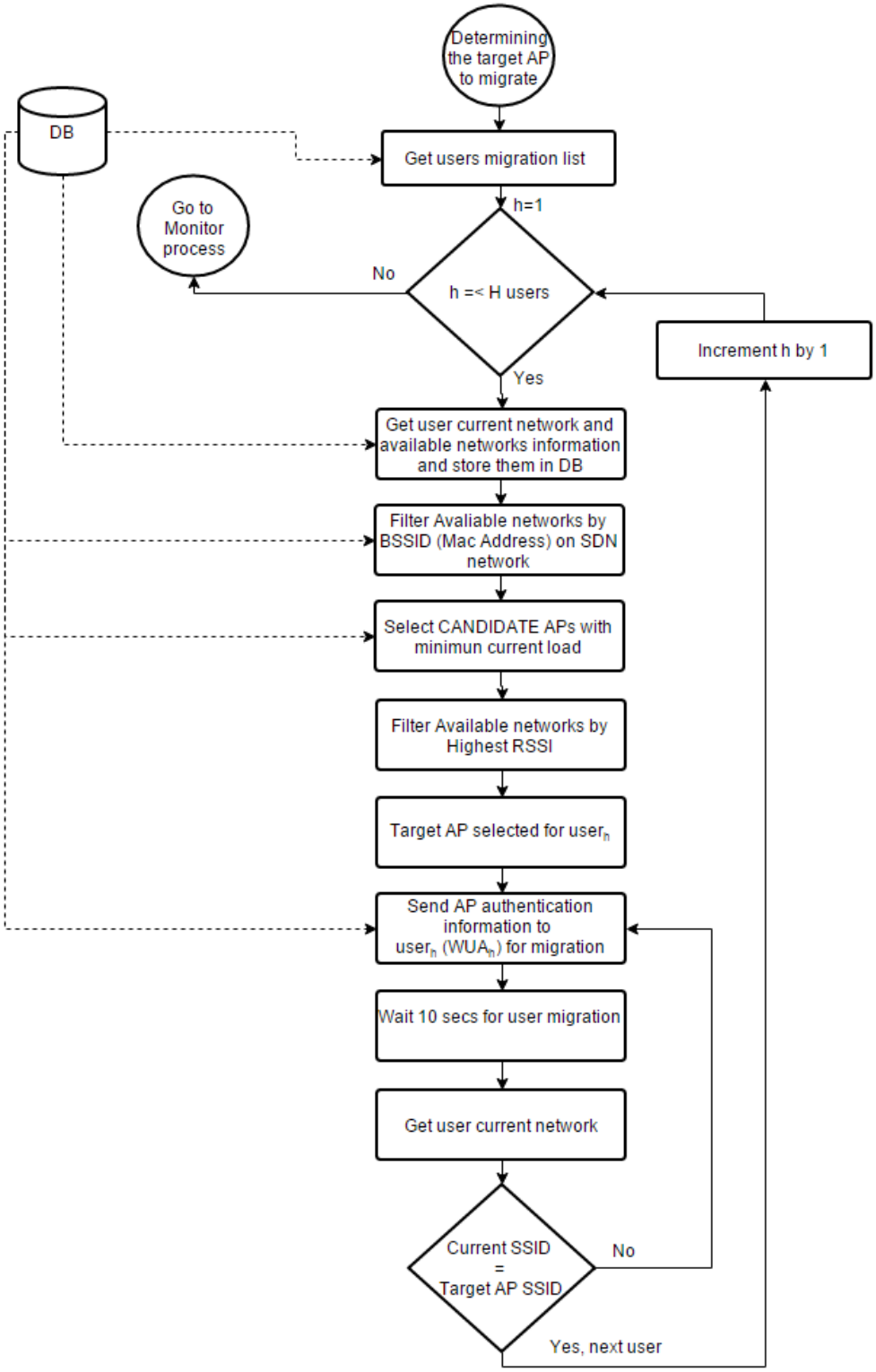

Figure 14. Flow chart for determining the target AP

5. Emulation Model

This chapter describes the technical aspects and challenges related with the environment to test the WUMS.

From the research conducted in this thesis, it was observed that there is no open and free out-of-the-box SDN testbed that can fulfill the requirements to develop and test the WUMS. For instance, the popular simulation tool ns-3 [44], provides accurate wireless models that includes the IEEE 802.11 standard. However, ns-3 lacks of an updated Openflow implementation. On the other hand, the current emulator Mininet [45], includes a wired SDN network model capable to run on a single computer, nevertheless, there is no wireless models available to use. In the case of real SDN implementations such as Openroads [42] and Ethanol [15], the two provide wired and wireless conditions. However, both approaches utilize a user-level software to run the Openflow protocol on each network device. Usually, user-level softwares are not well integrated with the operating system (OS), and thus they may cause poor scheduling decisions on the forwarding plane. Therefore, as part of the emulation model design, this thesis considers the OpenvSwitch, which is a standardized Openflow software that is part of the Linux kernel. This software supports various tunnelling methods and is commonly used on datacenter environments. Finally, the SDN controllers used on [15], [42] and [44] do not support REST APIs. This obstructs the interaction of upper services and applications with the SDN environment in a standardized manner. Following, the comparison of all the aspects between different testing scenarios is shown in Table 2.

As a result of the limitations found in the literature regarding testing environment, a real integrated wired/wireless SDN environment was built to test the WUMS.

Table 2. Comparison of SDN environments

Environment	Network type	Platform mode	Openflow version	Openflow software type	SDN Controller	REST API
ns-3	Wired/Wireless	Simulation	0.89	Kernel Level (Open vSwitch 2.0.2)	Internal Controller (C++ based)	No
Mininet	Wired	Emulator	1.0-1.3	Kernel Level (Open vSwitch 2.0.2)	Internal or external Controller	Depends on the external controller used
OpenRoads	Wired/Wireless	Real hardware devices (slicing network)	1.0	User level	NOX (C++ based)	No
Ethanol	Wired/Wireless	Emulation/Real hardware	1.0	User level	POX (Python based)	No
Thesis approach	Wired/Wireless	Emulation/Real hardware	1.0-1.3	Kernel level (Open vSwitch 2.3.0)	Opendaylight (Java based)	Yes

5.1.Model Description

The following implementation was conducted at the Advanced Network Technologies and Security (ANTS) Research Lab of UOIT. The environment consists of wired and wireless network devices connected together. However, instead of employing the traditional architecture of a distributed control and data plane, the integration of wired and wireless networks was done by employing SDN principles. As a result of utilizing this approach, operational simplicity and cost effectiveness was achieved by relying on open standards and virtualization tools. Hence, the testing environment was built from free and open source software, as well as, by using vendor-independent solutions of hardware and software.

The integrated SDN environment consists of wired and wireless network devices, a controller and stationary wireless users.

Wired devices are virtual machines (VMs) that run on top of the same physical server. This is possible by employing a virtualization software on the server. This software, often called hypervisor, manages resource and provides hardware abstraction and network connectivity to VMs. Moreover, on top of this virtualization platform, VMs are equipped with a Linux OS and the software to run the Openflow protocol, which is

the Open vSwitch [57] version 2.1.2. Thus, multiple VMs inside the server can act as SDN switches.

On the other hand, the wireless network devices are real commercial SOHO APs. Instead of using the manufacturer's software (firmware), these APs run the OpenWRT software. The OpenWRT [58] is a lightweight Linux operating system for embedded devices that provides CLI and GUI interfaces for management and troubleshooting. The operating system image was customized in order to add the Openflow protocol (Open vSwitch version 2.3.0) and networking tools such as Iperf. In this way, turning commercial APs into a SDN switches results in a straightforward integration of these devices with the virtual wired SDN network.

Further, wired and wireless devices requires the presence of an SDN controller. This is to manage traffic flows and network statistics on the entire environment. Thus, a dedicated VM (inside the server) is used as the SDN controller. Except for the additional software required to run the controller functionality, the VM is provisioned with the same characteristics as any other wired device in the server.

Finally, stationary wireless users are represented by laptops distributed in fixed locations, within the radio frequency range of the APs. A Wireless user has the WUA application activated and after associating to an AP, it generates certain traffic load in the environment. Though wireless users are part of the SDN environment, they are not conceived as SDN devices. In other words, all wireless users use the traditional networking stack to send and receive traffic in the network. Figure 15 presents a general view of the testing environment and the main components. Notice that more than a single wireless users is emulated on each laptop.

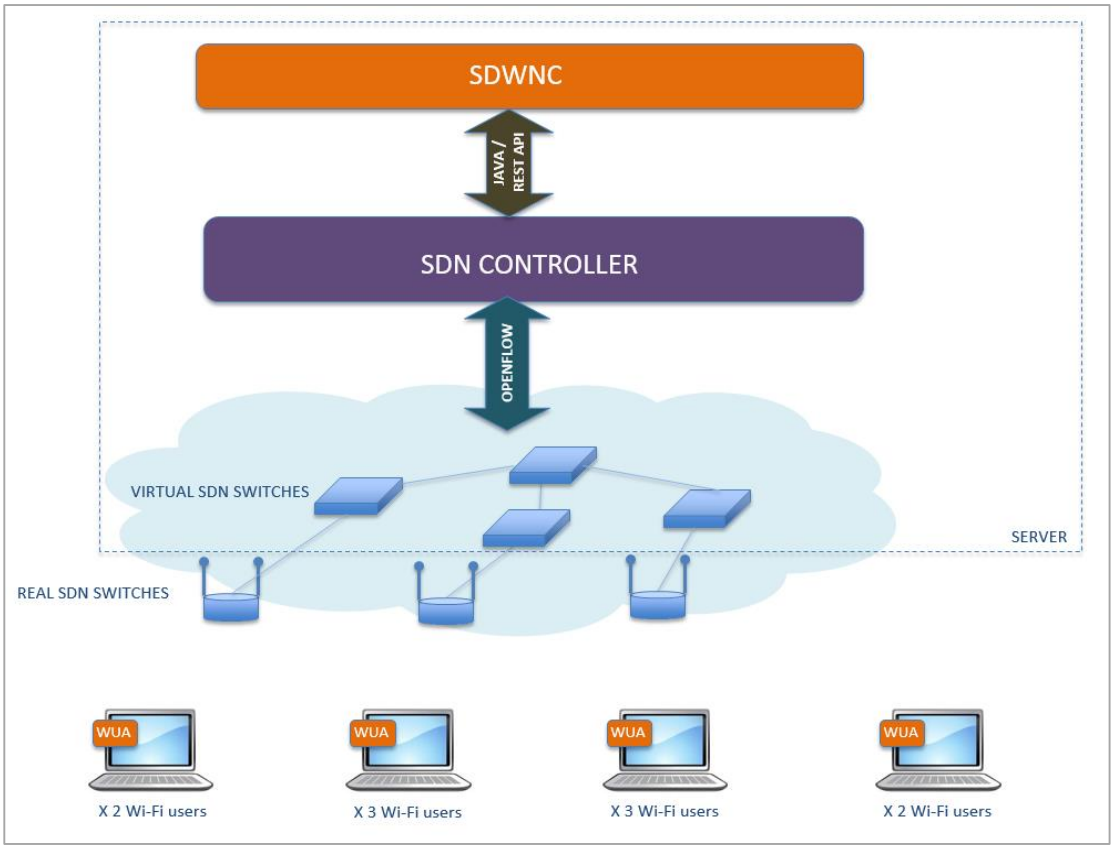

Figure 15. Emulated model

In the same way as with wireless and wired devices, the network connectivity in the SDN environment is both physical and virtual. In the case of wireless devices, the 100/1000 Ethernet WAN interface is used to physically connect the AP to a traditional lanswitch that provides connectivity to the server. However, the connectivity inside the server between VMs (controller and wired devices) is virtual. The virtualization platform provides a virtual switching layer, so that communication between VMs and external network devices is possible. Hence, connection are set by adding the VM network interfaces to the virtual switching layer.

Besides connectivity in layer 1, 2 and 3, the SDN architecture requires the establishment of virtual links between the SDN switches (wired and wireless devices). Virtual links are set on SDN switches by using the VxLAN tunnel encapsulation. Particularly, choosing a convenient encapsulation protocol was another design factor. In fact, the software to run the Openflow protocol (Open vSwitch), which is installed in all the SDN switches of the environment, supports the new encapsulation protocols NVGRE [59], and VxLAN [60]. Both NVGRE and VxLAN are capable of managing

more than 16 million VxLAN segments with a 24-bit identifier, allowing IEEE 802.11 users and services/applications to be in the same virtualized layer 2 network. Considering some of the reported problems with NVGRE, such as its Equal-CostMultiPath-based load balancing and IP fragmentation issues [55], we decided to use VxLAN for our purpose.

Once all virtual links are configured between SDN switches (wired and wireless), an overlay network is created. At this point, the SDN controller is capable to retrieve this information from each SDN switch (wired and wireless device). With this information, the SDN controller builds a topology map of the network. In particular, the SDN controller get knowledge of which port inside a SDN switch belongs to a certain virtual link in the topology. Given the fact that a SDN flow is a rule that specifies the input and output port of certain network traffic inside the SDN switch, the SDN controller can use topology information to install, remove and update flows on each SDN switch. In this way, when there is no flow that matches that traffic that enters the SDN switch, the SDN controller install the required flow. Later, the SDN switch can forward the traffic out to the port specified in the flow, which belongs to a virtual link that connects to a neighbouring switch.

As shown in Figure 16, wireless users and web services/applications communicate through the overlay network.

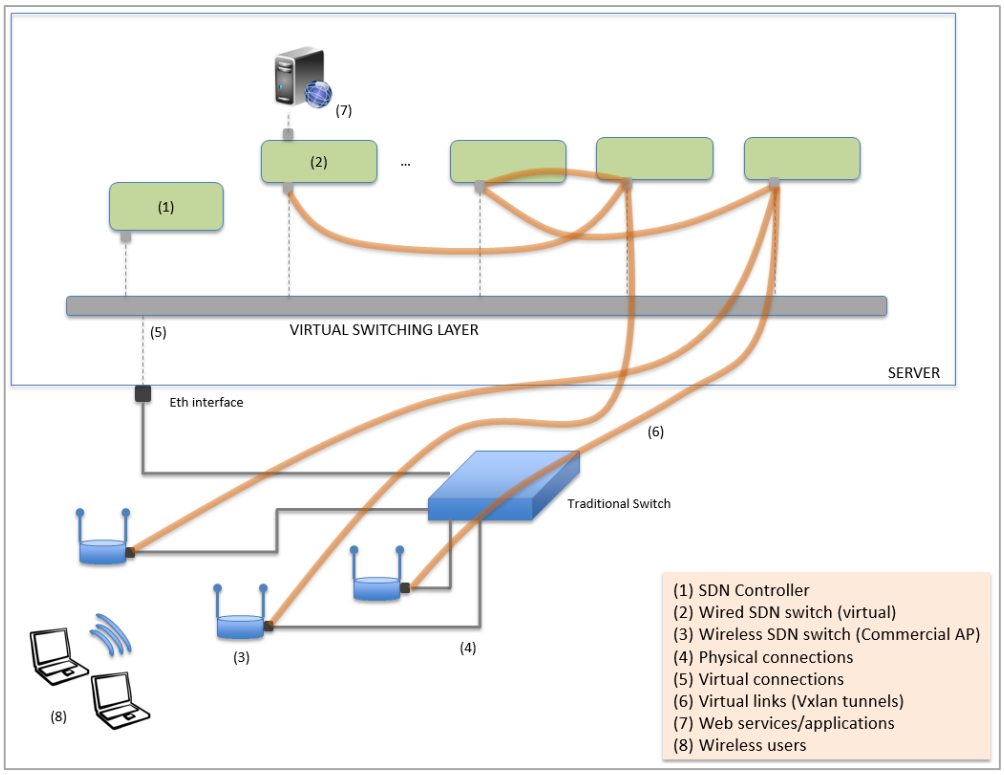

Figure 16. Network connectivity in the testing environment

Apart from establishing virtual links, further configuration was required in the SDN switches to allow connectivity between wireless users and web services/applications.

In general, SDN switches have an internal bridge to which virtual links and interfaces are added. In this way, the traffic from/to wireless users and web services/applications can be injected in the overlay network. On the wireless side, it is required that the wireless SDN switch has the WLAN interface set inside the bridge along with the virtual links (VxLAN tunnels). On the wired side, web/services applications running inside the server follows a similar approach. Basically, an interface of the virtual web service/application is added to the wired SDN switch bridge. However, it worth mentioning that, in this implementation, web/services and applications run on nested VMs. This secondary layer of virtualization is created on top of the wired SDN switch which is, actually, the first layer of virtualization inside the server. Nested VMs are connected to the overlay network by a virtual interface. At the same time, the virtual interface is attached to the bridge on the underlying wired SDN switch. Therefore, user traffic that arrives to that bridge destined to a web service/application will be forwarded to the nested VM.

A similar approach is followed to support the SDWNC functionality of the WUMS. However, this particular nested VM requires connectivity with the underlying network

in which the SDN controller resides, as well as, it requires access to the overlay network in order to contact the WUA component (on wireless users).

Figure 17 provides a conceptual scheme of the configuration explained above

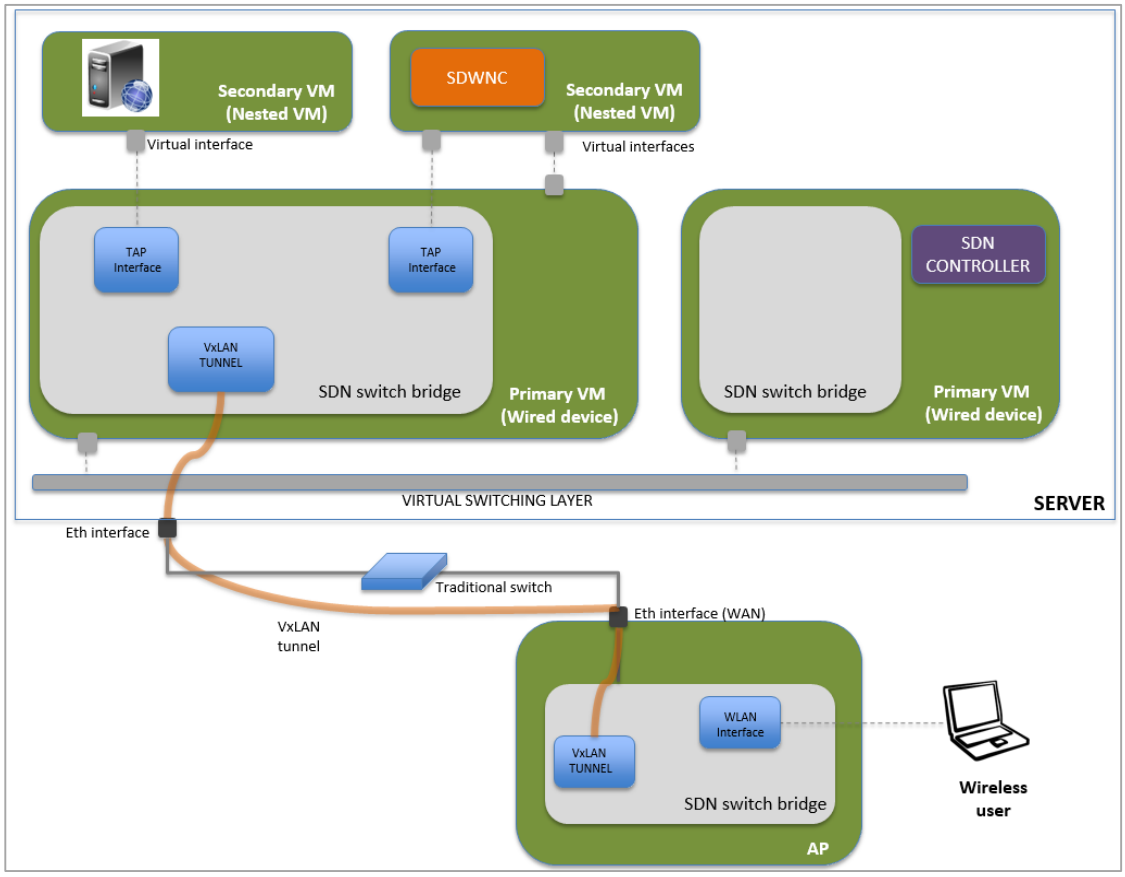

Figure 17. Description of internal connectivity for the SDWNC, web services/applications and wireless users

As mentioned before, in order to test the WUMS, wireless users inject traffic load in the environment. Although wireless users are represented by laptops, some of them share a portion of the hardware resources. Precisely, a single laptop holds a real instance of a wireless user plus multiple virtual instances to represent other wireless users. This is to fulfill the requirement of having a considerable numbers of users for the test, while considering the limitation of the number of laptops available in the lab. Starting with a real wireless user, a laptop can hold an additional user by creating a dedicate VM as well as by adding an extra wireless card (usb-wireless adapter) to the hardware. Notice that the maximum number of virtual instances that can be created on a single laptop depends on its hardware resources (RAM and/or USB ports available). Figure 18 shows the concept of having a single laptop with multiple wireless users

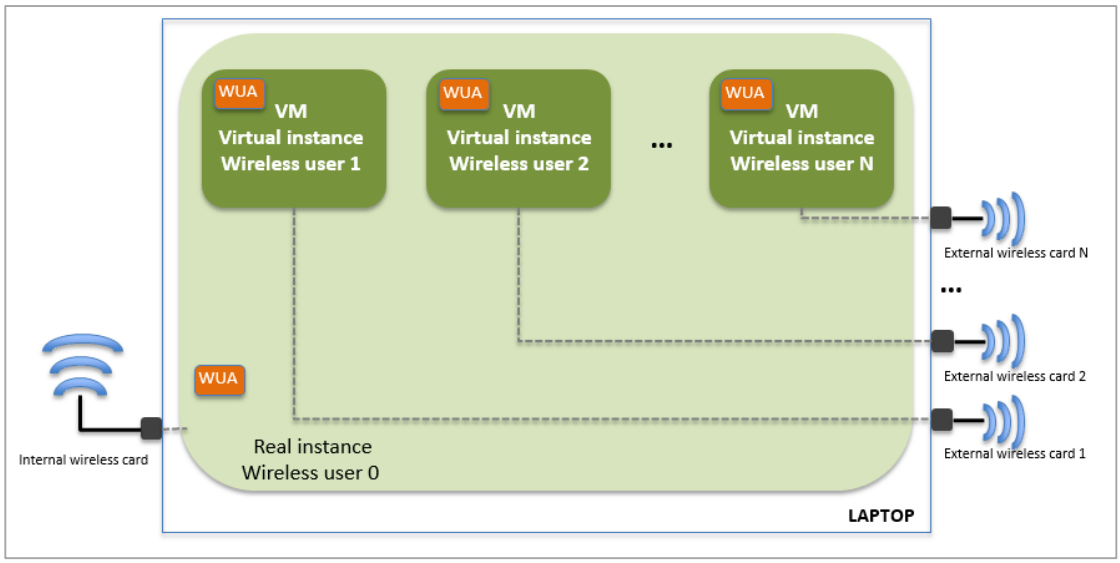

Figure 18. Emulation of wireless users in a single laptop

5.2. Implementation Considerations

During the set-up of the SDN environment a number of challenges were identified. It is important to mention that some of the issues found were not consider critical for the fulfillment of the thesis goals. However, they are relevant in the SDN field, and need to be addressed in future developments. Following, there is a list and description of the implementation issues:

- **IP Addressing**: In real scenarios, wireless networks provide a dynamic IP configuration service (DHCP service) to users during the association process. The DHCP service can either be distributed on each AP or centralized in the network. However, the integration of IEEE 802.11 networks in a SDN architecture challenges this traditional approach. Applying the SDN architecture leads to the utilization of a single broadcast domain. As shown in the previous section, the overlay network is basically a layer 2 network in which end-points (wireless users and services/applications) are able to communicate. In order to build this layer 2 network, the interfaces need to be configured as part of the local bridge of the SDN switch. As a result, it is expected that interfaces inside a bridge do not use IP address to operate [46]. However, this leads to implementation issues with existing software on network devices, such as what actually occurs with the OpenWRT in APs. In

particular, when the AP WLAN interface is added to the SDN switch bridge, the existing IP address settings, as well as, the services related with the interface (i.e.: DHCP service) are left without effect. A possible solution to this issue may be to include a dedicated DHCP server in the network and to configure the SDN controller to allow redirection of DHCP traffic (request and responds) between the users and the DHCP server. Authors of OpenRoads [42] experienced a similar issue and worked in favor of that solution. However, in this thesis, this is not a primary factor. Hence, in our WUMS wireless users utilized a static IP address to communicate. Therefore, when wireless users re-associate with an AP, the user's IP address remains the same. Nevertheless, the reader should be aware that in real scenarios, the extra delay introduced by the DHCP process must be considered during users association.

- **Open authentication:** In relation with the previous point, APs uses the open method to authenticate wireless users during the re-association process. Hence, the WUMS utilizes the target SSID as the only credential during migrations. However, in real deployment of wireless networks Shared key, EAP, Mac Address or pre-authentication methods are commonly used to provide a level of security. These methods require additional credential information to grant access to the AP and the message handshake during the authentication process introduces extra delay on user migrations [47].

- **Flow updates during user migrations:** The WUMS moves users from one AP to another in order to accommodate traffic load. When this happens, it is expected that user connectivity experiences a short disruption, but restoration should occurs immediately. However, during the first trials of the WUMS, users connections were never re-stablished after migrations. This was due to an issue found on the SDN controller functionality. In particular, it was noticed that when a wireless user migrates to another AP, the SDN controller does not track the user location properly. Actually, the SDN controller shows the wireless user connected on both previous and current AP. As a result of this malfunction, the SDN controller does not perform the flow update in the SDN network accordingly. Later, it was found that by modifying the user flows with the correct input and output ports at only one SDN switch of the

topology, the issue could be fixed. In fact, this SDN switch is the one closer to the web service/application VM, which happens to distribute the traffic (coming from the web service/application) to each AP in the topology.

Finally, in order to fix this problem, an automatic procedure of flow modification was added in the development of the SDWNC. Basically, the SDWNC updates the user flows in the SDN switch before and after performing the user migration. Precisely, once a user is selected for migration, the SDWNC updates the user flows so that network connectivity for the current and target AP is assured. After migration, the SDWNC modifies the user flows allowing connectivity with the target AP only. This approach requires to pieces of software. One is a module located in the SDWNC to send flow updates (TCP_Connection_OVS). Whereas, the other piece of software is an application that runs in the VM of the SDN switch, which receives the updates and apply the changes. Figure 19 shows the architecture design of the solution and Figure 20 depicts the final state of the flow table in the SDN switch after a user migration.

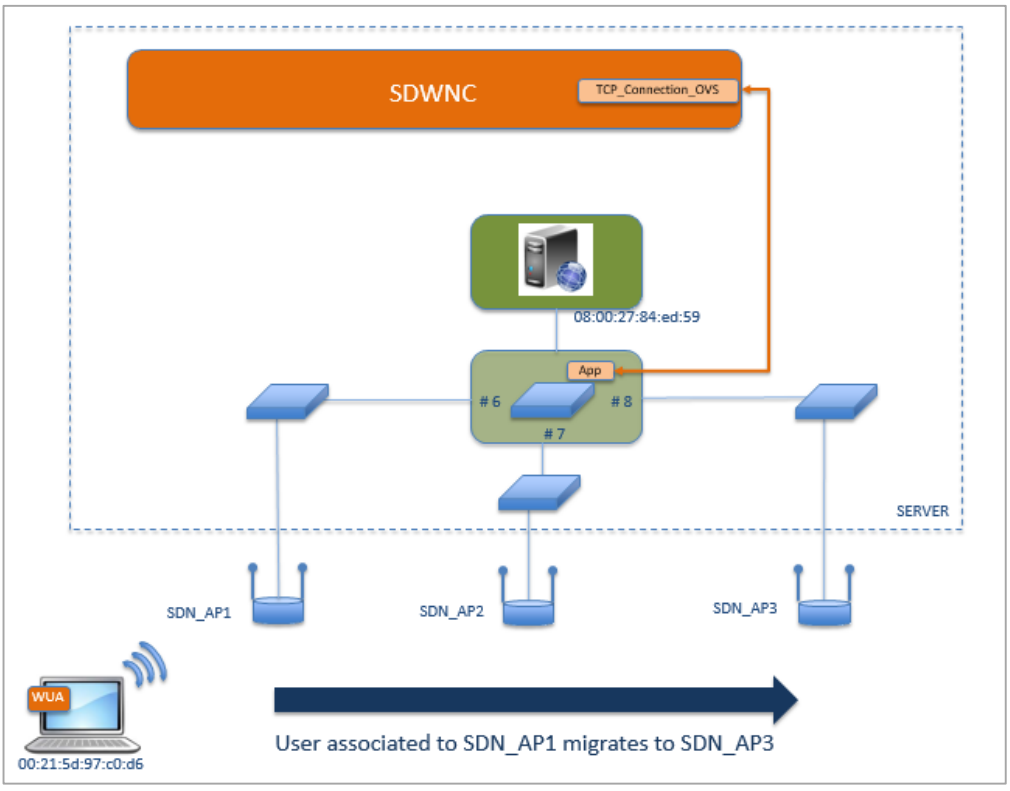

Figure 19. Architecture design of the flows update solution

```
[root@OVS20-Wireless_Test_Server ~]# ovs-ofctl -O Openflow13 dump-flows br10
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x2a00000000000000, duration=5.324s, table=0, n_packets=402, n_bytes=403716, idle_timeout=1800, hard_timeout=3600, pr
iority=10,dl_src=00:21:5d:97:c0:d6,dl_dst=08:00:27:84:ed:59 actions=output:4
 cookie=0x2a00000000000001, duration=5.323s, table=0, n_packets=264, n_bytes=399024, idle_timeout=1800, hard_timeout=3600, pr
iority=10,dl_src=08:00:27:84:ed:59,dl_dst=00:21:5d:97:c0:d6 actions=output:8
 cookie=0x2b000000000003f, duration=6.432s, table=0, n_packets=5, n_bytes=999, priority=2,in_port=8 actions=output:4,output:
5,output:6,output:7,output:3,CONTROLLER:65535
 cookie=0x2b00000000000040, duration=6.432s, table=0, n_packets=1, n_bytes=342, priority=2,in_port=3 actions=output:4,output:
5,output:6,output:7,output:8,CONTROLLER:65535
 cookie=0x2b000000000003e, duration=6.433s, table=0, n_packets=4, n_bytes=957, priority=2,in_port=7 actions=output:4,output:
5,output:6,output:8,output:3,CONTROLLER:65535
 cookie=0x2b000000000003d, duration=6.433s, table=0, n_packets=2, n_bytes=184, priority=2,in_port=6 actions=output:4,output:
5,output:7,output:8,output:3,CONTROLLER:65535
 cookie=0x2b000000000003c, duration=6.433s, table=0, n_packets=0, n_bytes=0, priority=2,in_port=5 actions=output:4,output:6,
output:7,output:8,output:3,CONTROLLER:65535
 cookie=0x2b000000000003b, duration=6.434s, table=0, n_packets=1, n_bytes=42, priority=2,in_port=4 actions=output:5,output:6
,output:7,output:8,output:3,CONTROLLER:65535
 cookie=0x2b00000000000009, duration=9.869s, table=0, n_packets=54, n_bytes=48476, priority=0 actions=drop
 cookie=0x2b00000000000009, duration=9.869s, table=0, n_packets=6, n_bytes=670, priority=100,dl_type=0x88cc actions=CONTROLLE
R:65535
```

Figure 20. Flow update after user migration in SDN switch

- **Bandwidth limitation policy and out-band signalling:** A bandwidth limitation policy was configured on every AP of the environment. This was required to test the WUMS and APs under congestion conditions, and to obtain network trace files manageable in size for later analysis. The policy affects user traffic (UDP and ICMP). However, it does not have any effect on TCP, which is used for the signaling traffic between SDWNC and WUA. Figure 21 shows the set of commands required to configure the policy on every AP. In particular, the policy was set to limit bandwidth over 15Mbps for UDP (protocol 17) and ICMP (protocol 1) traffic in the WLAN interface of each AP.

```
root@SDN_AP2:~# tc qdisc add dev wlan0 root handle 1: htb
root@SDN_AP2:~# tc class add dev wlan0 parent 1: classid 1:1 htb rate 15000kbit
root@SDN_AP2:~# tc class add dev wlan0 parent 1:1 classid 1:10 htb rate 15000kbit
root@SDN_AP2:~# tc class add dev wlan0 parent 1:1 classid 1:20 htb rate 15000kbit
root@SDN_AP2:~# tc qdisc add dev wlan0 parent 1:20 handle 20: sfq perturb 10
root@SDN_AP2:~# tc qdisc add dev wlan0 parent 1:10 handle 10: sfq perturb 10
root@SDN_AP2:~# tc filter add dev wlan0 protocol ip priority 1 u32 match ip protocol 17 0xff flowid 1:10
root@SDN_AP2:~# tc filter add dev wlan0 protocol ip priority 1 u32 match ip protocol 1 0xff flowid 1:10
root@SDN_AP2:~# tc filter add dev wlan0 protocol ip priority 1 u32 match ip protocol 17 0xff flowid 1:20
root@SDN_AP2:~# tc filter add dev wlan0 protocol ip priority 1 u32 match ip protocol 1 0xff flowid 1:20
```

Figure 21. Configuration of the bandwidth limitation policy on an AP

5.3.Proof-of-concept

During the implementation of the emulated model, a proof-of-concept was carried out. The goal of this test was to verify the correctness of the architecture design and the

implementation of the WUMS in the integrated wired/wireless SDN environment. The following elements conform the test:

- 11 x stationary wireless users. Wireless users emulated by real and virtual instances running on 4 laptops randomly distributed with extra IEEE 802.11n wireless cards (usb-wireless adapters)
- Each user generates 1Mbps UDP traffic in the downlink direction
- 3 x IEEE 802.11n APs with different SSID operating at different channel in 2.4Ghz (wireless SDN switches)

In the first part of the test, the WUMS is in passive mode. In this mode, the system only runs the Monitoring and Balancing Factor Analysis process. For the second part, it is fully activated, which means that all process of the UMA execute.
Initially, the wireless users are distributed in the same AP, except for one wireless users that is connected to another AP. In other words, one AP (SDN_AP1) supports 10 wireless users, the second AP (SDN_AP2) has one wireless users associated and the third AP (SDN_AP3) is idle. This situation generates load asymmetry in the network. Once activated, the WUMS performs a user analysis on the over-loaded AP (SDN_AP1), and selects number of wireless users to migrate. Finally, WUMS performs migrations from the over-loaded AP to other APs in the vicinity. In this way, the WUMS distributes the total load in the network by migrating wireless users among nearby APs. The results shown in Table 3 depicts the final distribution of the wireless users

Table 3. Wireless users association after migrations

Wireless users	Migrated	Associated to:
3.3.3.10	No	SDN_AP1
3.3.3.11	No	SDN_AP1
3.3.3.12	Yes	SDN_AP3
3.3.3.20	No	SDN_AP2
3.3.3.21	Yes	SDN_AP2
3.3.3.30	Yes	SDN_AP3
3.3.3.31	Yes	SDN_AP2
3.3.3.32	No	SDN_AP1
3.3.3.40	Yes	SDN_AP3
3.3.3.41	Yes	SDN_AP3
3.3.3.42	No	SDN_AP1

Another aspect to verify regarding the implementation, is the mechanism used by the WUMS to detect unbalanced conditions. As explained in chapter 4 (User Migration Algorithm), when the fairness formula (2) returns beta values below 0.9, asymmetry is detected in the network. Consequently, the WUMS performs the rest of the user management operations in order to mitigate the issue. Figure 22 shows balancing factor values measurements before and after the user management process. From 0 to 290 *secs*, the WUMS reports balancing factor values of 0.4 (closer to $\frac{1}{I}$) , which clearly reflects asymmetry in the network. Next, from 290 to 484 *secs* (194 *secs*), the WUMS performs all the user management tasks to accommodate the user load in the network. After migrations, the new balancing factor values computed by the WUMS are closer to 1. These results evidence that the WUMS performed the redistribution of the load successfully, and that the network is balanced.

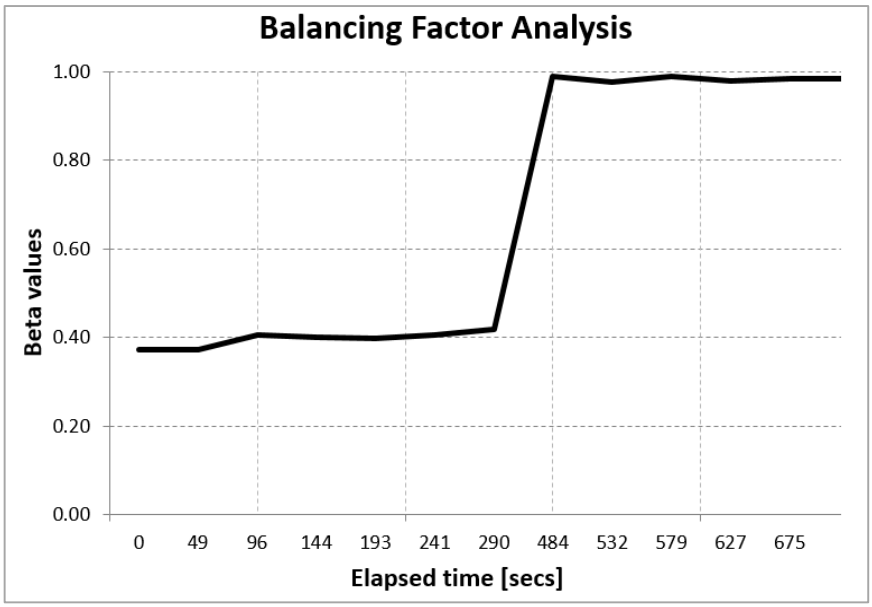

Figure 22. Balancing factor indicator before and after the WUMS migrate the wireless users

Finally, the final load redistribution exhibits a smooth allocation of the user traffic among nearby APs. This is because the WUMS checks the current load in the under-loaded APs, before migrating the user. Figure 23 provides the overall port rate (P_i) supported by each AP during the test, before and after user migrations. The results show that both SDN_AP2 and SDN_AP3 takes the exceeded load in SDN_AP1. Consequently, the P_i value on each AP is closer to $\overline{L}$. As expected, the WUMS redistributes 11 wireless users (each one with the same traffic rate) by assigning 4 users to two APs and 3 users to the remaining one.

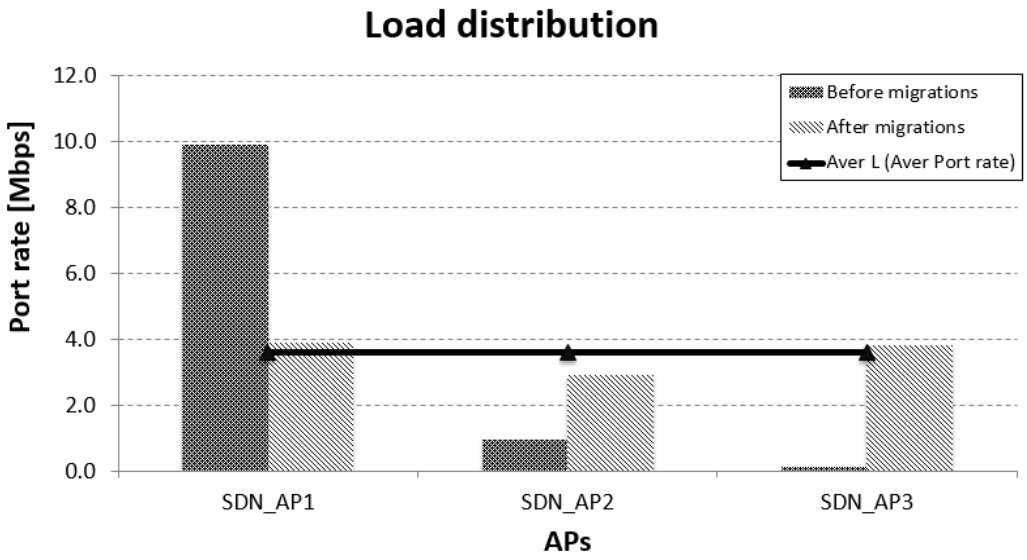

Figure 23. Load distribution on each AP during the test

5.4. Technical Details

This section describes all the hardware, software and tools used in the environment:

- **Server**: DELL CS24-T4 with 32GB RAM, 925 GB HDD and 12 INTEL XEON x 2.133 GHZ
- **Virtualization platform**: VMWare ESX-I hypervisor
- **Operating System on Server's VMs**: Linux CentOS 6.5 with Open vSwitch 2.1.2 software
- **Virtualization software for Server's Nested VMs:** VirtualBox version 5
- **Operating System on Server's Nested VMs**: Linux Ubuntu 15.4
- **SDN Controller**: Opendaylight Helium SR3. Among several SDN controllers currently available, the Opendaylight project (ODL) was selected due to its open source Java-based architecture, Openflow support, northbound REST API, extensive documentation available and continuous support from the community.
- **Virtual links between SDN switches:** VxLAN tunneling method
- **SDN southbound protocol:** Openflow version 1.3
- **Commercial APs:** TP-LINK 1043ND Qualcomm Atheros QCA9558 64MB RAM, 8MB FLASH and ath9k wireless driver.
- **Operating System on commercial APs**: Linux OpenWRT 14.07 with Open vSwitch 2.30. The OpenWRT is essentially a lightweight Linux OS developed for embedded devices that can be modified to include packages such as the Openflow protocol.
- **Wireless user devices:** Lenovo Intel i7, 8GB RAM laptop with Intel Centrino advanced N6205 wireless card
- **Additional wireless cards for emulated wireless users:**
    - TP-LINK TL-WN823N 300Mbps Wireless Mini USB Adapter
    - TP-LINK TL-WN725N Wireless N Nano USB Adapter 150Mbps
- **Virtualization software for emulated wireless users:** VirtualBox version 5
- **Operating System on wireless user devices**: Windows 7
- **Media server**: VLC media player
    - **Video file for streaming:** 1920 x 1040 MPEG-4 video file
    - **Audio file for streaming:** MP3 audio file

6. Performance Evaluation

In regards to the initial question of this thesis, the WUMS was developed to improve the user QoS on an integrated wired/wireless SDN network. Hence, analyzing the behaviour of the UMA from the wireless user perspective is vital for the fulfilment of the thesis goals. In particular, the results presented here aims at finding out the degree of QoS enhancement on wireless users when the WUMS takes the corresponding actions to mitigate network imbalance.

The WUMS is evaluated using the scientific methodology that relies on the empirical evidence collected from the environment. The network traffic of wireless users, captured by a network analyser tool, provide the required information to analyze the WUMS under this methodology.

6.1. Scenarios

All scenarios replicate the undesirable situation that occurs on public/crowded places. In those situations, a single AP supports all the traffic load from stationary wireless users, despite other APs in the vicinity remain idle. Scenarios are composed with the following elements:

- 3 x IEEE 802.11n APs (wireless SDN switches) with:
    - Different SSID (SDN_AP1, SDN_AP2 and SDN_AP3)
    - Operating at different channel in 2.4Ghz (channels 1,6 and 11)
- 10 x stationary wireless users. Wireless users emulated by real and virtual instances running on 4 laptops randomly distributed with extra IEEE 802.11n wireless cards (usb-wireless adapters)SDN controller (running in dynamic forwarding mode)
- WUMS (SDWNC, and WUA on each wireless user)
- 15Mbps bandwidth limitation policy on APs for UDP & ICMP traffic
- Web services/applications: A media server for video and audio streaming
- Wireless user types. Heavy user: 2Mbps UDP video streaming and 500kbps ICMP. Light user: 200kbps UDP audio streaming

- All wireless users capturing traffic (network traces)

Moreover, all scenarios have wireless users with new and old session durations. The reason for that is that the UMA start migrating wireless users with recent connections, instead of moving those who have been connected for a long time. In addition, the experiments take into account the traffic load in the over-loaded AP (port rate P_i). The latter is the additional input for the WUMS to mitigate the asymmetry. Therefore, all tests include both heavy and light types of users depending on the current traffic load.

In the first scenario, shown in Figure 24, a number of both heavy and light users have been connected to the over-loaded AP for a long time. In addition, the newcomers light users contribute on the total traffic load handle by the AP. At this point of the experiment, the WUMS should be activated in order to redistribute the exceeded load of the over-loaded AP to other APs in the vicinity. After mitigating this asymmetry, the tests should evidence an improvement on users' performance.

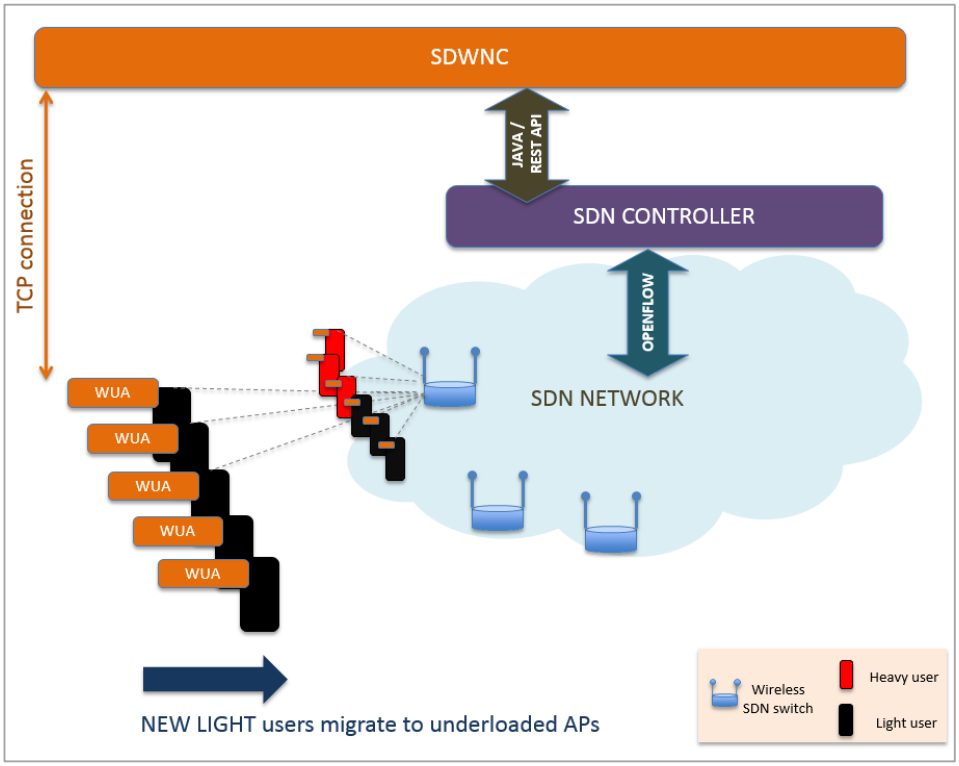

Figure 24. Scenario of new light and old heavy users in over-loaded AP

On the other hand, the second scenario, shown in Figure 25, depicts a set of heavy and light users that has been connected for a long time. Meanwhile, new users generates high amount of traffic load in the AP. At that instant, the WUMS should redistribute the exceeded load of the over-loaded AP to other APs in the vicinity. After this, the tests should evidence an improvement on users' performance for this scenario as well.

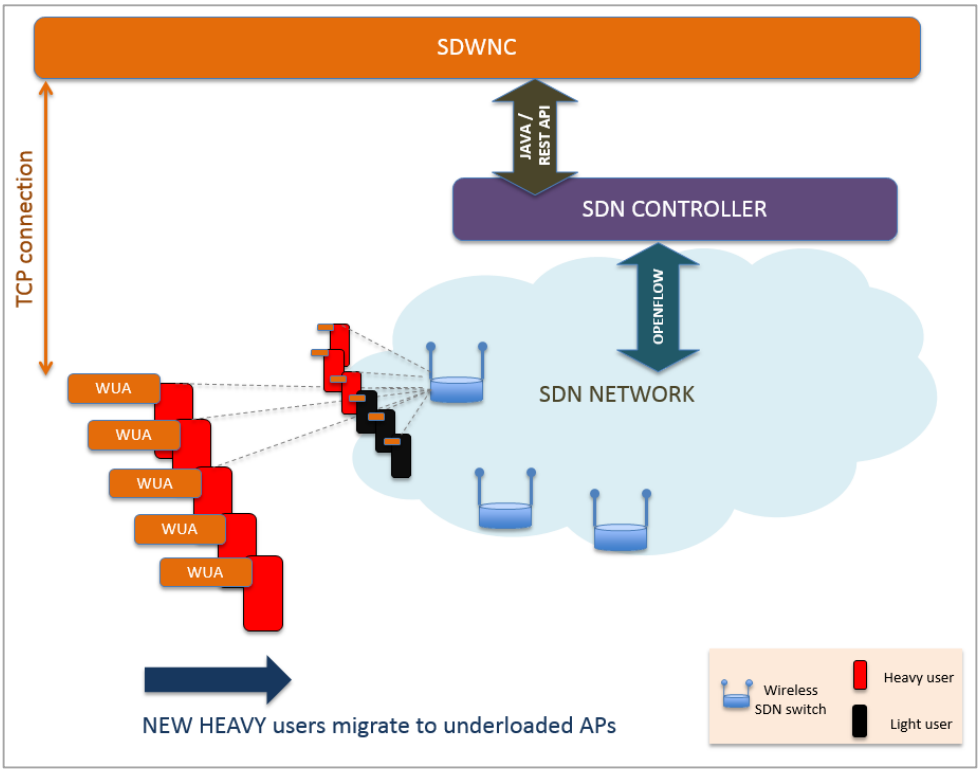

Figure 25. Scenario of new heavy and old light users in over-loaded AP

In addition, within each scenario, the number of new and heavy users varies as follows:

- 80 % of heavy users and 20 % light users
- 50 % of heavy users and 50 % light users
- 20 % of heavy users and 80 % light users

Thereby, applying this combination of user types to the scenarios presented above, will evaluate theUMA under several networking conditions. More precisely, the UMA will return different migration decisions when the number of heavy and light user changes from one scenario to another. This is because the UMA selects a user to

migrate based on its session duration and the accumulated load of previous selected users. Lastly, all the scenarios described before will be run with and without background traffic (BT). In those cases, the BT represents 50% of the total user load, which is appended to the total user load generated on the over-loaded AP. It is important to highlight that BT is assumed as fixed load in the AP, in a sense that the WUMS cannot re-distribute it. Accordingly, BT is considered an external source of disturbance in the AP that the WUMS cannot control. Table 4 summaries all these scenarios.

Table 4. Different testing scenarios for the WUMS

TESTING SCENARIOS	80% HEAVY 20% LIGHT		50% HEAVY 50% LIGHT		20% HEAVY 80% LIGHT	
	NO BT	BT 50%	NO BT	BT 50%	NO BT	BT 50%
NEW LIGHT & OLD HEAVY	Test_10_1	Test_10_3	Test_20_1	Test_20_3	Test_30_1	Test_30_3
OLD LIGHT & NEW HEAVY	Test_10_2	Test_10_4	Test_20_2	Test_20_4	Test_30_2	Test_30_4

6.2.Results

This section discusses the results obtained from the scenarios of Table 4. All scenarios start with wireless users associated to the same AP (SDN_AP1). After the first 300 *secs*, the WUMS is activated, and the asymmetry is detected. At this point, the WUMS analyzes the load in SDN_AP1 and determine the set of wireless users to migrate. By the end of the test (approximately 800 *secs*), the total user load in the network is shared among other nearby APs (SDN_AP2 and SDN_AP3).

All experiments have a duration of approximately 800 *secs*. In order to measure the improvement obtained before and after applying user migration, each experiment considers:

- Data before migration: Between 100 and 300 *secs* of the test time

- Data after migration: Between 500 and 700 *secs* of the test time

According to Table 4, the 80H/20L are presented first. Next, the 50H/50L scenarios are exhibited, followed by the 20H/80L scenarios.

80H/20L with and without BT (Test_10_1, Test_10_2, Test_10_3 and Test_10_4): In the 80H/20L scenarios, the number of heavy users is greater than light users. The total traffic load for 80H/20L (NL_&_OH and OL_&_NH) is 20.4 Mbps without BT, and 30.6 Mpbs with BT. In fact, due to the bandwidth limitation policy of 15 Mbps presented on each AP, there is an initial congestion in SDN_AP1. The results in Figure 26 show a considerable improvement in the UDP traffic rate after the execution of the WUMS, especially in the OL_&_NH scenarios:

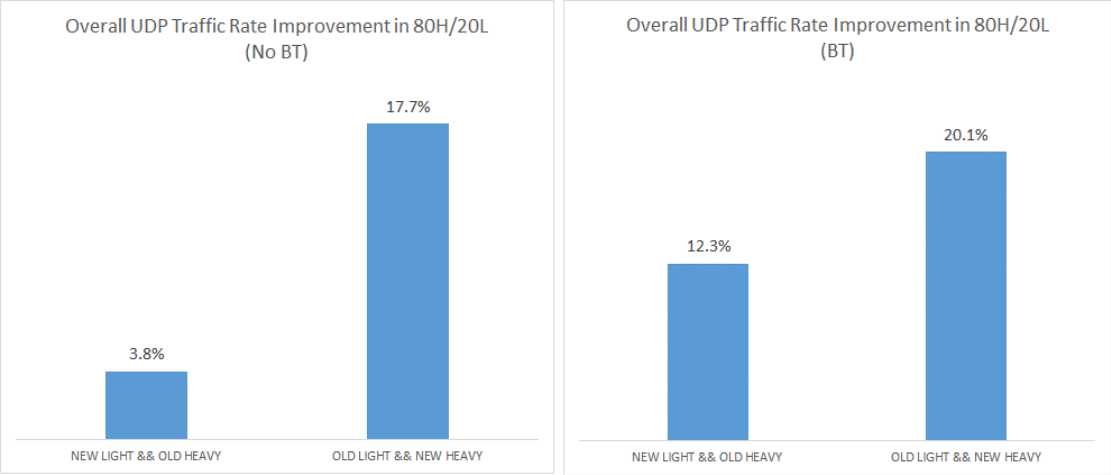

Figure 26. Overall UDP traffic rate improvement in 80H/20L with and without BT

Next, Figure 27 presents noticeable improvements in the overall end-to-end delay and jitter for both scenarios. These network parameters are related with the UDP traffic (video and audio streaming) of all wireless users:

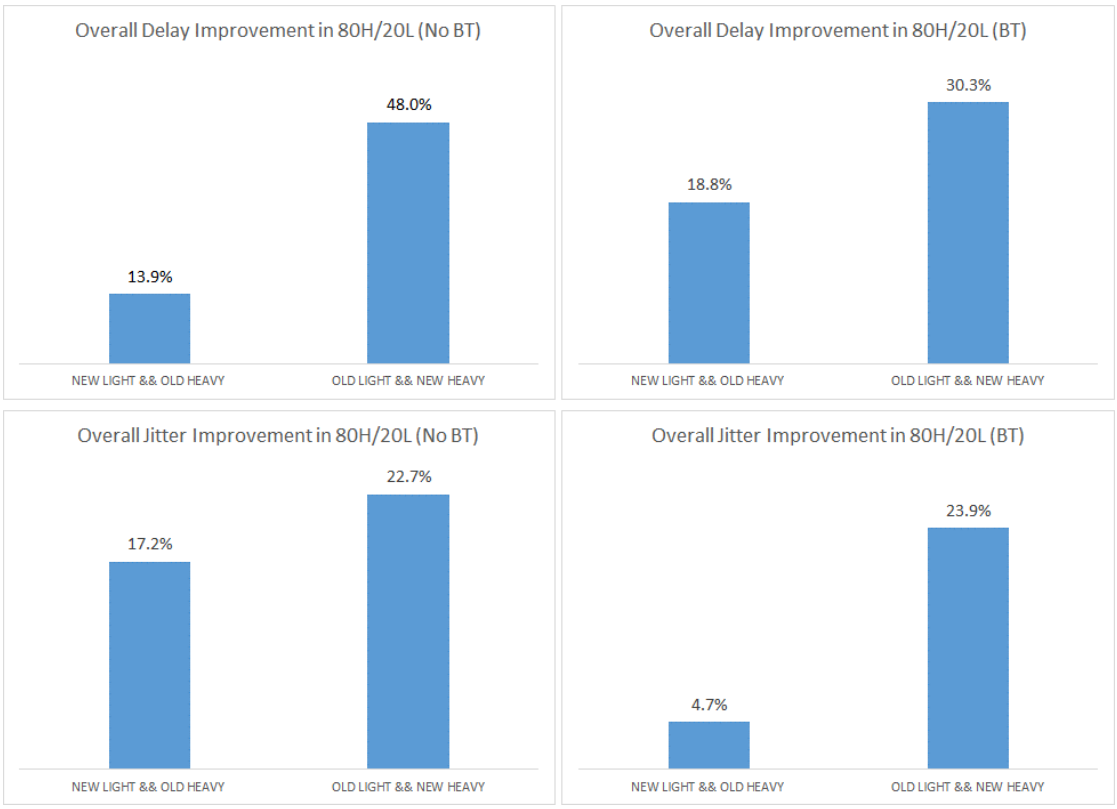

Figure 27. Overall delay improvement in 80H/20L

Regarding the ICMP traffic generated by heavy users, the average number of responses received by wireless users is low during congestion. However, after migrations ICMP responses increase substantially for both scenarios. Figure 28 shows the results of the 80H/20L without BT (left) and with BT (right).

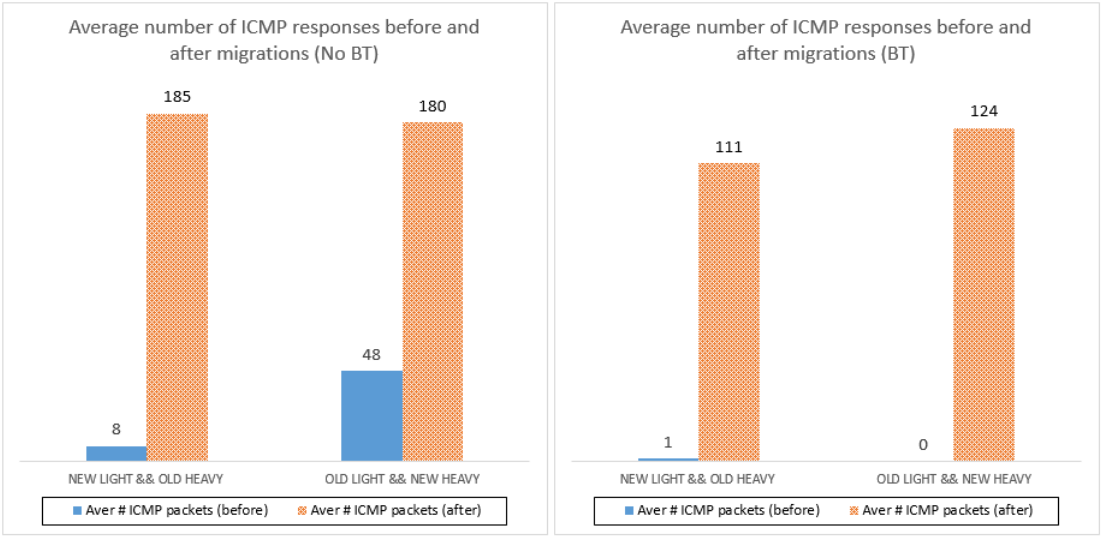

Figure 28. Average number of ICMP responses received by wireless users in 80H/20L with and without BT

Lastly, the results, in Figure 29, show the disconnection gap (in seconds) that each wireless user experienced during the migration process. Notice in Figure 29 that non-migrated users have no disconnection gap (zero).

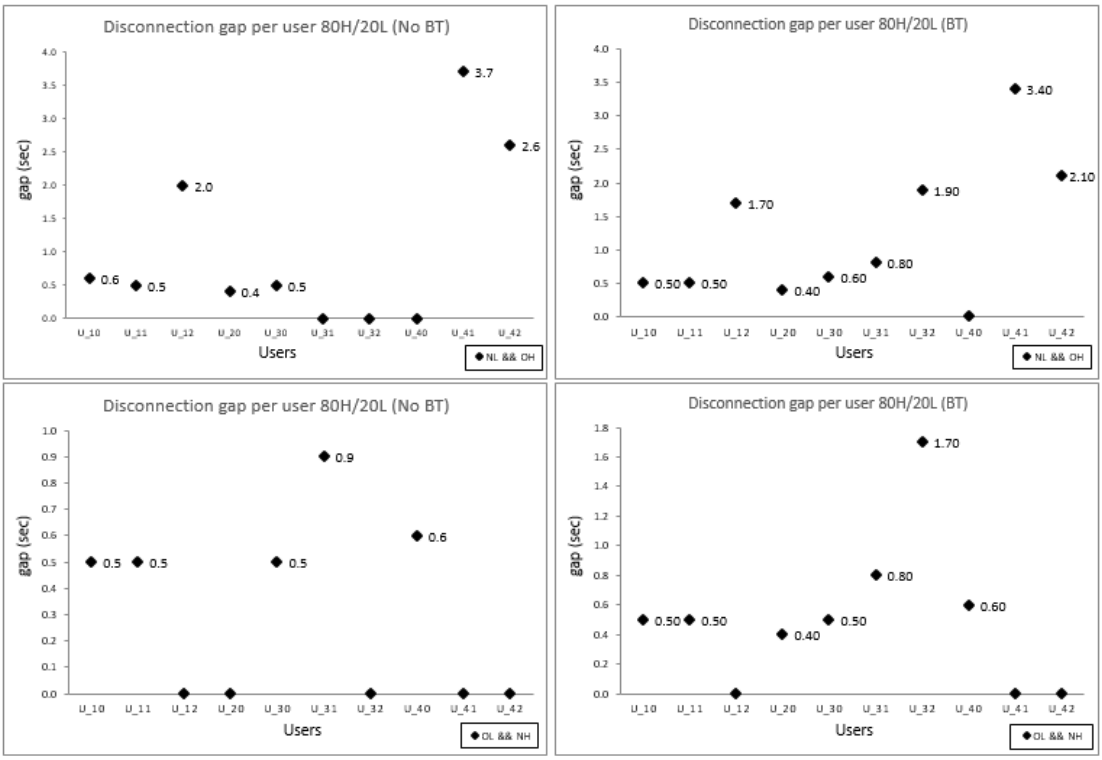

Figure 29. Disconnection gap per wireless user for 80H/20L with and without BT

50H/50L with and without BT (Test_20_1, Test_20_2, Test_20_3 and Test_20_4):

In the 50H/50L scenarios, an equal number of heavy and light users is considered. The total traffic load for 50H/50L (NL_&_OH and OL_&_NH) is 13.5 Mbps when BT is not applied, and 20.25 Mpbs with BT.

In the case of no BT, there is no initial congestion in SDN_AP1, so the UDP traffic rate does not experience any improvement after migrations.

On the other hand, when BT is present, the SDN_AP1 is affected by an initial congestion. However, there is no clear enhancement of UDP traffic rate, after performing user migration.

Figure 30 show these results without BT (left), and 50H/50L with BT (right):

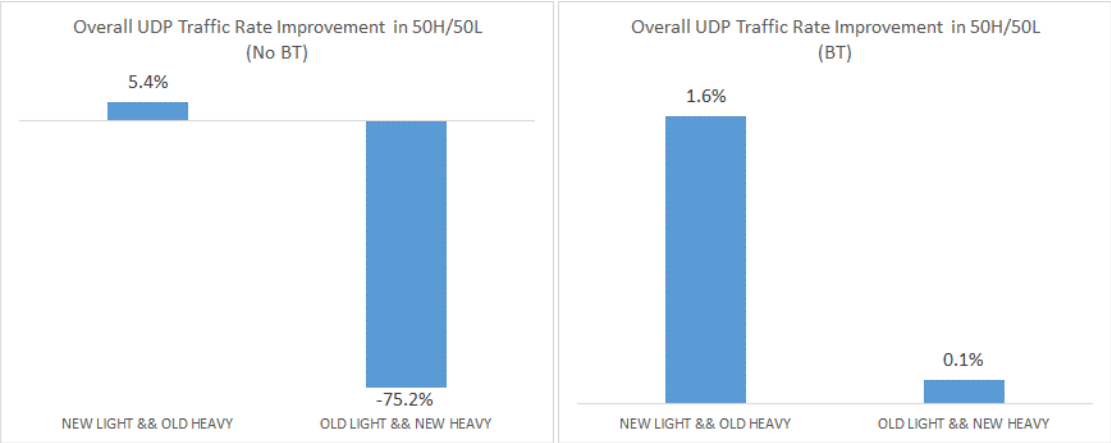

Figure 30. Overall UDP traffic rate improvement in 50H/50L with and without BT

Next, the scenario without BT does not exhibit any improvement in terms of end-to-end UDP delay and jitter. On the contrary, when BT is applied, the jitter is the only network parameter that is enhanced, especially when the scenarios is NL_&_OH. Figure 31 shows the end-to-end UDP delay and jitter without BT (left) and with BT (right). Notice in the Figure 31 that in both cases there is no significant enhancement of these parameters.

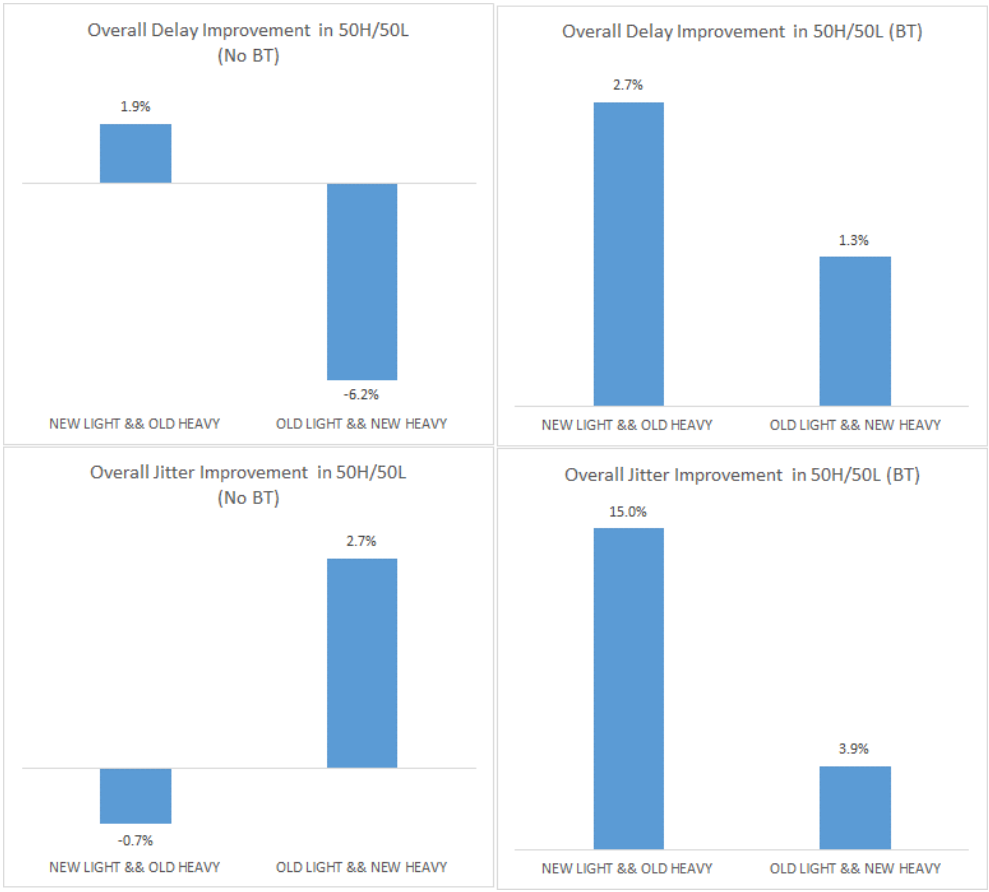

Figure 31. Overall delay and jitter improvement in 50H/50L with and without BT

Regarding the ICMP traffic generated by heavy users, the average number of responses received by wireless users is low, before migration. After migrations, ICMP responses increase substantially for both scenarios. This is shown in Figure 32 for the scenario without BT (left) and with BT (right). Notice in the Figure 32 that before migrations, there are more average packets received in the scenario without BT than the one with BT. The reason is that the scenario without BT never exceeds the bandwidth limitation policy in SDN_AP1.

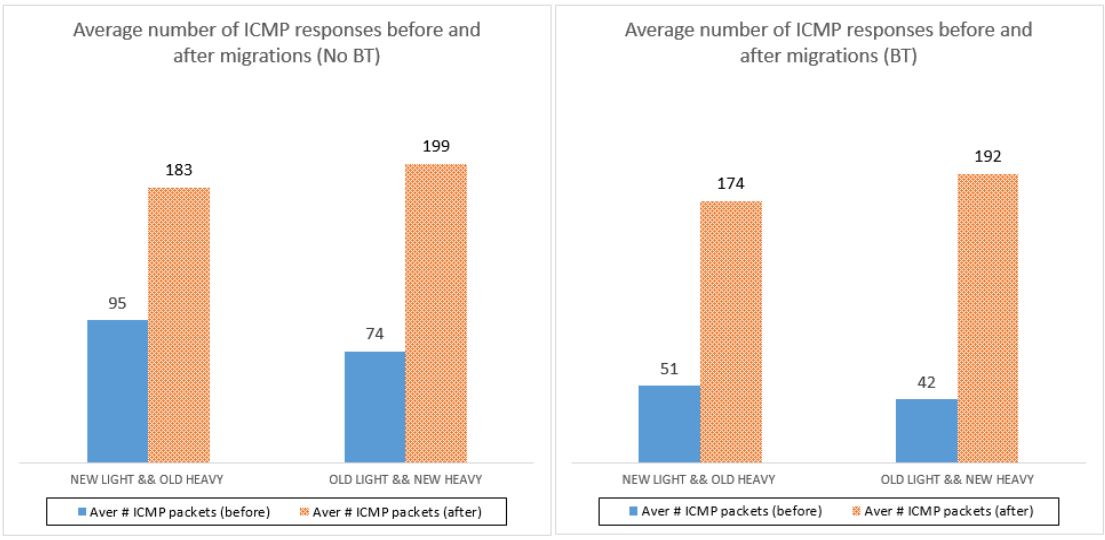

Figure 32. Average number of ICMP responses received by wireless users in 50H/50L with and without BT

Finally, the set of dispersion graphs of Figure 33 display the disconnection time of every user in all 50H/50L scenarios. User with zero disconnection time are users that were not migrated.

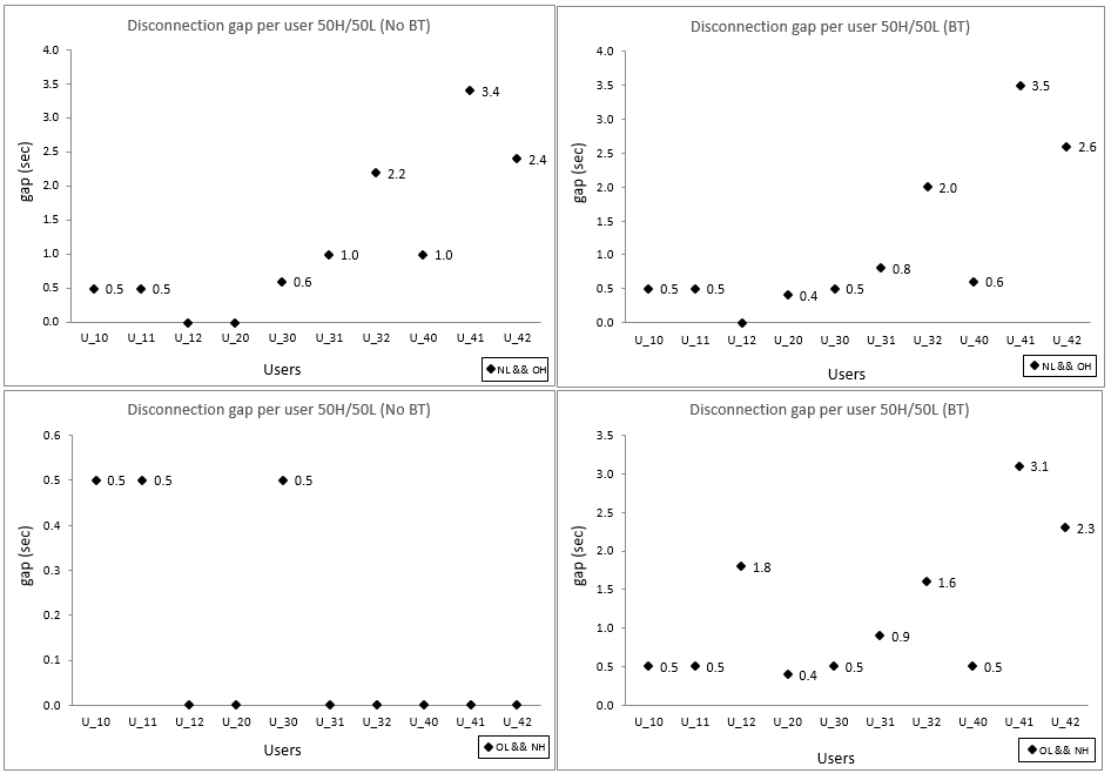

Figure 33. Disconnection gap per wireless user for 50H/50L with and without BT

20H/80L with and without BT (Test_30_1, Test_30_2, Test_30_3 and Test_30_4): In the 20H/80L scenarios, the number of light users is greater than heavy users. The total traffic load for 20H/80L (NL_&_OH and OL_&_NH) is 6.6 Mbps without BT, and 9.9 Mpbs with BT. In both cases, there is no initial congestion in SDN_AP1, so the UDP traffic rate does not get a substantial improvement after migrations. Figure 34 show these results without BT (left), and 20H/80L with BT (right):

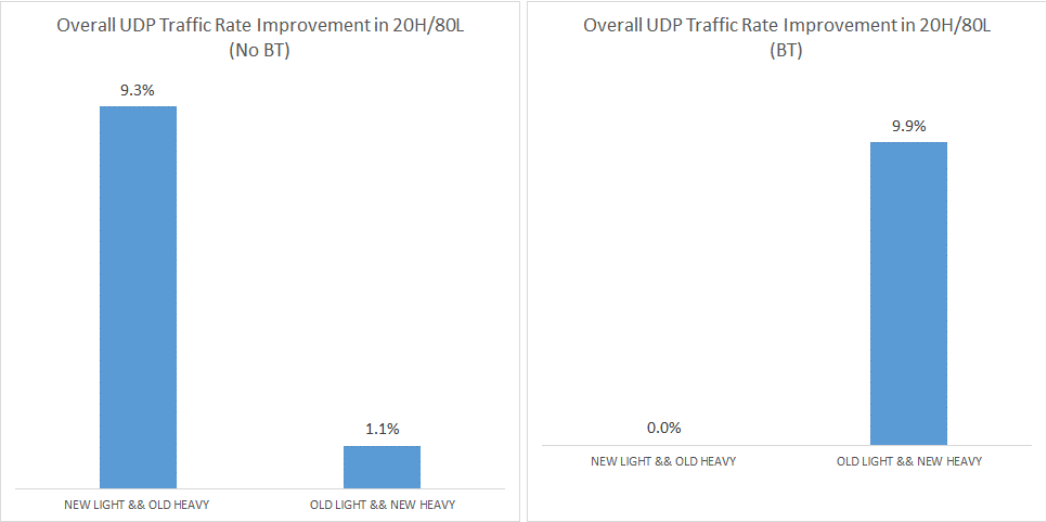

Figure 34. Overall UDP traffic rate improvement in 20H/80L with and without BT

Next, the scenarios does not exhibit any substantial improvement in terms of end-to-end UDP delay and jitter. Particularly, the results obtained in the OL_&_NH scenario with BT show minor improvements in these parameters. In this case, the WUMS migrates all the wireless users to other APs (SDN_A2P2 and SDN_AP3) leaving only the BT in SDN_AP1. However, the results are not significant and consistent in comparison with the ones obtained in congested scenarios. Figure 35 shows the end-to-end UDP delay and jitter without BT (left) and with BT (right).

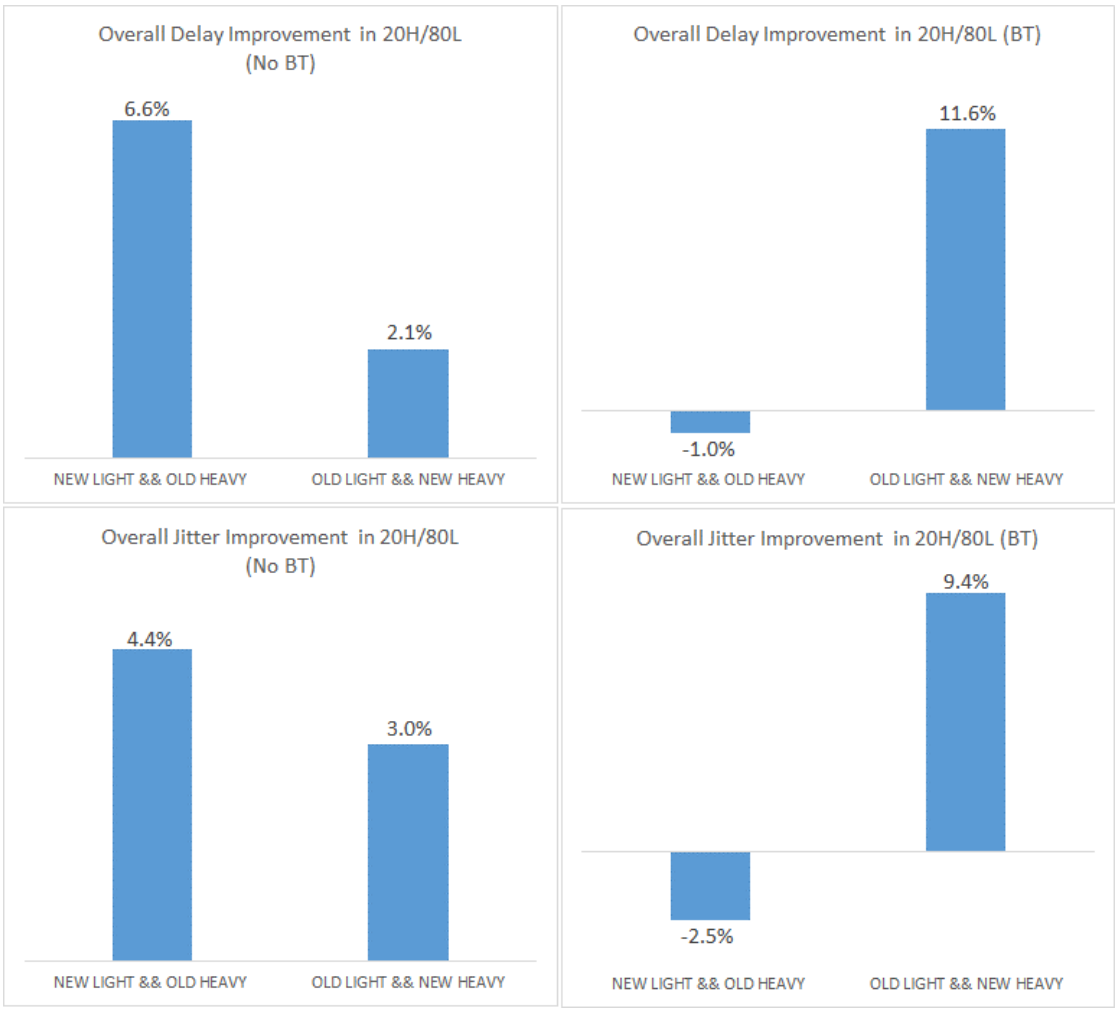

Figure 35. Overall delay and jitter improvement in 20H/80L with and without BT

In regards to the ICMP traffic generated by heavy users, the average number of responses received by wireless users experience considerable changes. This is shown in Figure 36 for the scenario without BT (left) and with BT (right).

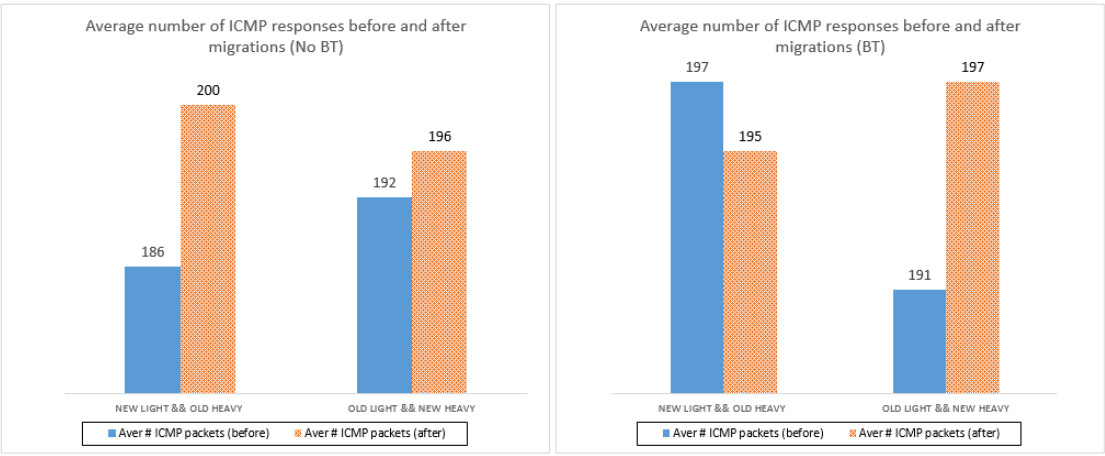

Figure 36. Average number of ICMP responses received by wireless users in 20H/80L with and without BT

Finally, the set of dispersion graphs of Figure 37 display the disconnection time of every user in all 20H/80L scenarios. User with zero disconnection time are users that were not migrated.

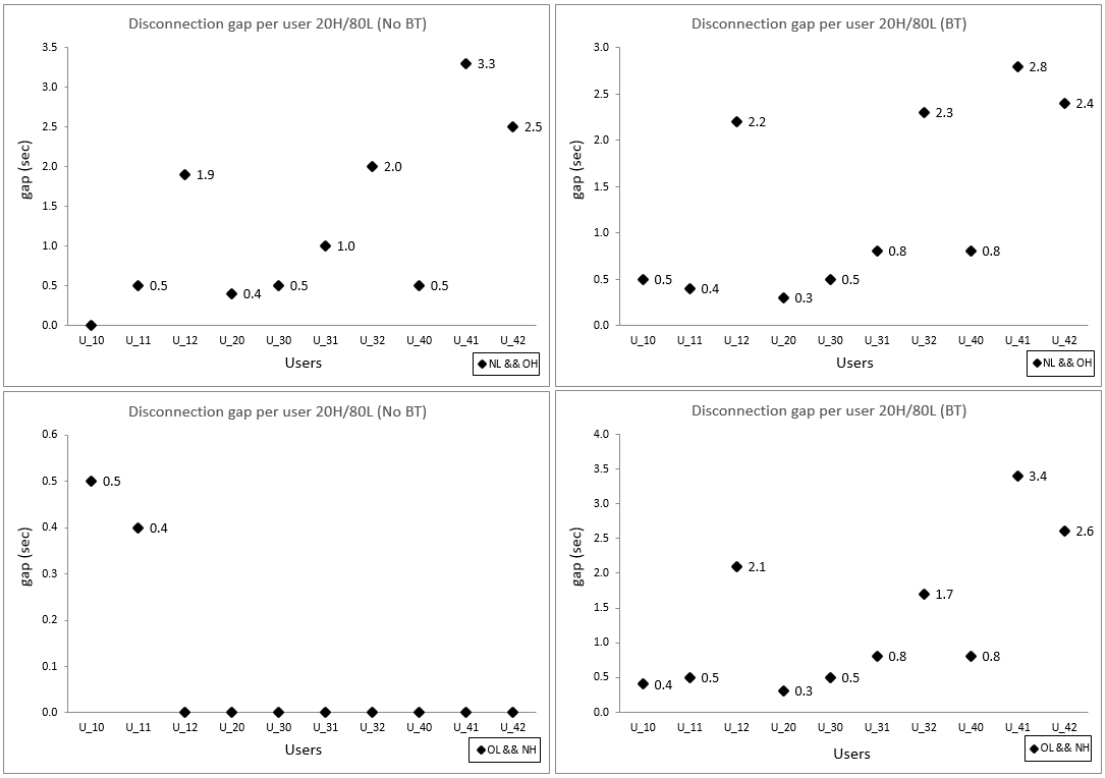

Figure 37. Disconnection gap per wireless user for 20H/80L with and without BT

The 80H/20L scenarios provided a noticeable improvement in end-to-end UDP delay and jitter. In these scenarios, the total traffic load exceeds the bandwidth limitation policy configured in SDN_AP1, so congestion occurs frequently. In particular, the 80H/20L scenarios returns major improvements in end-to-end jitter and delay when the UMA deals with OL_&_NH. This is because 80% of the users are heavy users. Hence, moving these users first and leaving light users in the AP helps alleviating congestion while it contributes in the network balancing tasks in a more efficient manner. On the other hand, in the case of NL_&_OH, the UMA starts migrating the light users first, which does not provide significant improvement until heavy users are migrated. Consequently, in order to solve the imbalance condition in the network, the WUMS performs additional user migrations. In fact, the OL_&_NH scenario resulted in less migrations than the NL_&_OH scenario.

Moreover, the 80H/20L scenarios show that the number of ICMP responses received by heavy users substantially increases, after the WUMS redistributes users among the APs.

On the other hand, the results from the 50H/50L and 20H/80L scenarios does not show a clear pattern of enhancement in end-to-end UDP delay and jitter. In fact, the results from these experiments indicated minor improvements and/or degradation of the user performance. In the same way, these scenarios do not show a noticeable improvement in the number of ICMP responses received by heavy users. Except for the 50H/50L scenario with BT in which congestion occurs, the traffic load of these scenarios is not initially affected by a bottleneck. Hence, in non-congestion scenarios, wireless users will not benefit from the WUMS.

Moreover, the disconnection gaps observed in all scenarios prove that connection disruptions are minimal for wireless users. This is due to the assisted approach provided by the WUMS during user migrations. Without this approach, wireless users would experience extra delay due to network scanning and several attempts of re-association with other AP in the vicinity. Although the WUMS wait for 10 *secs* before requesting the user for migration confirmation, the disconnection gap is usually much less than that value. In fact, the overall disconnection time of users in all scenarios is between 0.5 and 1.4 *secs*.

Another aspects to highlight, is the dissimilar end-to-end UDP delay and jitter improvement noticed on heavy and light users. **Error! Reference source not found.** epicts the values of heavy and light users in the 80H/20L scenario. Notice in the **Error! Reference source not found.** the values of end-to-end delay and jitter for the DP traffic of heavy users (left) versus the ones for light users (right).

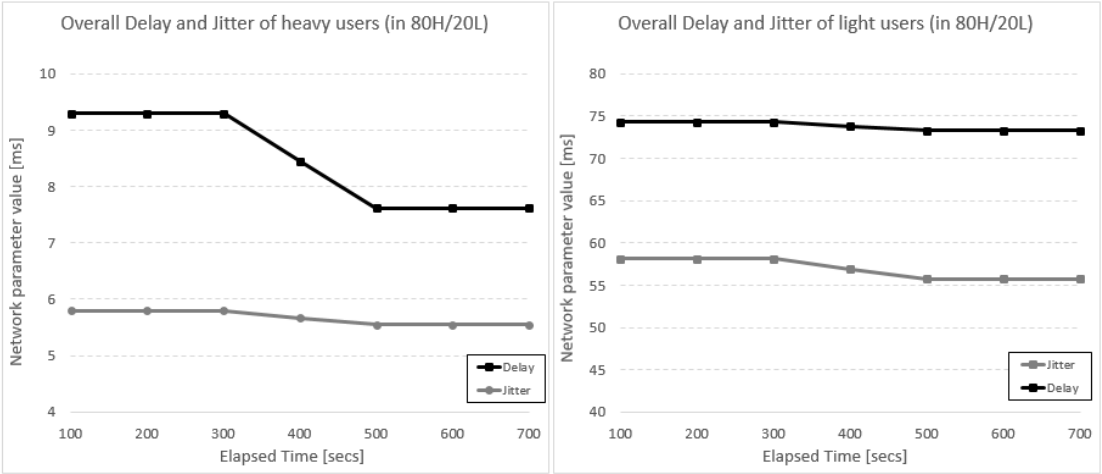

Figure 38. Example of overall delay and jitter of heavy and light users obtained in 80H/20L scenario

Heavy users receives video streaming (MPEG video) while light users receives audio streaming (MP3 audio). The video streaming traffic has more strict values of end-to-end delay and jitter than the audio streaming traffic. This makes heavy users more sensible to network changes than light users. As shown in **Error! Reference source ot found.**, the values of end-to-end delay and jitter obtained from the experiments are commonly:

- 7-12 *msec* (delay) and 5-10 *msec* (jitter) for heavy users
- 70-80 *msec* (delay) and 50-70 *msec* (jitter) for light users

In addition, the amount of traffic generated by a heavy user is 10 times greater than the one generated by a light user. When there is congestion in the AP, the traffic that belongs to a heavy user cannot be accommodated at the rate that the application (ie: video) requires (aprox 2Mbps). However, the traffic rate from a light user (200kbps) can be transmitted in short burst making it more robust to congestion.
Therefore, the improvements achieved in end-to-end delay and jitter were more evident on heavy users than in light users.

7. Conclusion and Future Works

The thesis provided a detailed insight on the development of a wireless user management system (WUMS) in an integrated wired/wireless SDN environment. The development of the WUMS was successfully achieved in a straightforward, standardized and cost-effective manner. This was possible thanks to the programmability and network abstraction features of SDN, which allow the creation of services/applications to manage certain aspects of the network from a centralized point. Also, the achievement was possible due to the flexibility of open standards and protocols that facilitated the deployment of non-vendor specific and affordable hardware to build the environment.

Both the implementation and the results evidences that the performance of wireless users can be enhanced by utilizing the WUMS under a SDN architecture. The proactive user management approach got prominent results in congested scenarios, especially, when migrating the heavy users in the first place. However, in non-congested scenarios the results were not favorable. Although, the WUMS alleviates the over-loaded AP by redistributing user load among other APs, there was no noticeable improvements on wireless users. This is due to the simplistic scheme adopted regarding the network metrics in the migration process. In fact, rather than just considering the traffic rate, session duration, available APs and signal strength, the UMA should make decisions based on additional information from the IEEE 802.11 physical and data layers, such as the channel collisions, interference, utilization time, as well as, information related with performance anomaly. However, this goal cannot be achieved, in a standardized manner if the SDN architecture is not wireless aware in the first place. Both the Openflow protocol and SDN controller requires wireless capabilities in order to collect, transport and present this information to upper services/applications. Although, there are plans to extend the Openflow protocol for wireless environments, the Open Networking Foundation (ONF), which is the organization in charge of Openflow protocol support, has not released any wireless version of the protocol to these days.

Certain aspects were not considered as part of this thesis; however they are worth mentioning for future works. One of these aspects is the size of the wired/wireless environment, which is not comparable to real deployments. In those cases, service

providers may offer IEEE 802.11 network access to multiple transient users with several APs, so the WUMS need to be analyzed in terms of scalability, management complexity and user mobility. Another aspect to explore in the future is the applicability of the WUMS in cooperative scenarios. For instance, a wireless users that belongs to one service provider may get better service if it is moved to a nearby AP of a different service provider. Sharing network information between providers can be considered as a win-win situation for both sides, due to the fact that network coverage and capacity is improved without major investments in the network. Thus, WUMS should collects information from several SDN controllers and migrates wireless users to more convenient APs (that may or not belong to the same service provider). Although the thesis does not tackled this particular aspect, the implementation is feasible and requires minor changes in the design of the WUMS.
Besides the prospective works related to user management, other IEEE 802.11 issues might be interesting to explore in an integrated SDN environment. Considering that SDN brings in the concept of network abstraction, upper services/applications can be developed in a straightforward manner by collecting network information from the SDN controller. For instance, a SDN service/application capable to perform fined-grained QoS management, might be worth studying in deep. A service/application of this characteristics can allow control of resources per network application and/or wireless user by means of the manipulation of Openflow meters in SDN switches. Another example of SDN services/applications that can be considered are the ones related with wireless security. For example, a system capable to detect untrusted users and propagate a user blacklist to other SDN networks might enforce network security in cooperative scenarios of multiple service providers. The challenges in the definition and implementation of security policies to prevent, detect and mitigate anomalies in a wired/wireless SDN environment can provide a great contribution in this direction.

Finally, as a result of the research conducted to elaborate this thesis, it was revealed that there is still a lack of literature regarding the performance evaluation of the Openflow protocol on different commercial APs. This is an important topic given the fact that the Openflow protocol runs on limited hardware resource devices. Hence, stress tests that considers the amount of traffic load, number of users, flows and policies that are supported with Openflow are a valuable source of information for researchers in the field.

8. References

[1] IEEE Standard for Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements: Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE, 2001.

[2] Oz, Effy. Management information systems. Cengage Learning, 2008.

[3] IEEE 802.11n-2009—Amendment 5: Enhancements for Higher Throughput". IEEE-SA. 29 October 2009. doi:10.1109/IEEESTD.2009.5307322.

[4] Wi-Fi Alliance, "Easy-to-Use Wi-Fi: Opportunities for Service Providers A Consumer Polling Brief from the Wi-Fi Alliance", https://www.wi-fi.org/download.php?file=/sites/default/files/private/20120620_Mobile_Subscribers_Want_Passpoint.pdf retrieved on Dec. 3, 2015.

[5] Picco-Schwendener, Anna, and Lorenzo Cantoni. "Tourists and Municipal Wi-Fi Networks (MWN): The Case of Lugano (Switzerland)." Information and Communication Technologies in Tourism 2015. Springer International Publishing, 2015. 565-578.

[6] Xfinity wifi, http://wifi.xfinity.com/ retrieved on Dec. 3, 2015.

[7] Fon, https://corp.fon.com/es retrieved on Dec. 3, 2015.

[8] Aerohive Networks, "High-Density Wi-Fi Design Principles", http://docs.aerohive.com/pdfs/Aerohive-Whitepaper-Hi-Density%20Principles.pdf retrieved on Dec. 3, 2015.

[9] Open Networking Foundation, https://www.opennetworking.org/sdn-resources/sdn-definition retrieved on Dec. 3, 2015.

[10] Unknown, Author. "OpenFlow Switch Specification, Version 1.5.1 (Protocol 0x06)." Mar 26 (2015)

[11] Suresh, Lalith, et al. "Towards programmable enterprise WLANS with Odin." Proceedings of the first workshop on Hot topics in software defined networks. ACM, 2012.

[12] Dely, Peter, et al. "Cloudmac—an Openflow based architecture for 802.11 MAC layer processing in the cloud." Globecom Workshops (GC Wkshps), 2012 IEEE. IEEE, 2012.

[13] Patro, Ashish, and Suman Banerjee. "COAP: a software-defined approach for home WLAN management through an open API." ACM SIGMOBILE Mobile Computing and Communications Review 18.3 (2015): 32-40.

[14] Kim, Won-Suk, et al. "Seamless Handoff and Performance Anomaly Reduction Schemes Based on OpenFlow Access Points." Advanced Information Networking and Applications Workshops (WAINA), 2014 28th International Conference on. IEEE, 2014.

[15] Moura, Henrique, et al. "Ethanol: Software Defined Networking for 802.11 Wireless Networks." IFIP/IEEE International Symposium on Integrated Network Management (IM). 2015.

[16] Kreutz, Diego, et al. "Software-defined networking: A comprehensive survey." proceedings of the IEEE 103.1 (2015): 14-76.

[17] Chaudet, Claude, and Yoram Haddad. "Wireless software defined networks: Challenges and opportunities." Microwaves, Communications, Antennas and Electronics Systems (COMCAS), 2013 IEEE International Conference on. IEEE, 2013.

[18] Costanzo, Salvatore, et al. "Software Defined Wireless Networks (SDWN): Unbridling SDNs." (2012).

[19] CISCO, “Aggressive Load Balancing on Wireless LAN Controllers (WLCs) Release 6.0.188.0 and Later Configuration Example”, http://www.cisco.com/c/en/us/support/docs/wireless/4400-series-wireless-lan-controllers/113160-aggressive-load-balancing-clients-00.html retrieved on Dec. 3, 2015.

[20] Juniper, “Understanding Load Balancing for Wireless Radios”, http://www.juniper.net/documentation/en_US/network-director1.5/topics/concept/wireless-loadbalancing.html retrieved on Dec. 3, 2015.

[21] Heusse, Martin, et al. "Performance anomaly of 802.11 b." INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies. Vol. 2. IEEE, 2003.

[22] Pilosof, Saar, et al. "Understanding TCP fairness over wireless LAN." INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies. Vol. 2. IEEE, 2003.

[23] Murty, Rohan, et al. "Dyson: An Architecture for Extensible Wireless LANs." USENIX Annual Technical Conference. 2010.

[24] Yen, Li-Hsing, Tse-Tsung Yeh, and Kuang-Hui Chi. "Load balancing in IEEE 802.11 networks." Internet Computing, IEEE 13.1 (2009): 56-64.

[25] Nicholson, Anthony J., et al. "Improved access point selection." Proceedings of the 4th international conference on Mobile systems, applications and services. ACM, 2006.

[26] Papanikos, Ioannis, and Michael Logothetis. "A study on dynamic load balance for IEEE 802.11 b wireless LAN." Proc. COMCON. Vol. 2001. 2001.

[27] Yukuda, Y., and Yuji Oie. "Decentralized access point selection architecture for wireless LANs e ployability and robustness." Vehicular Technology Conference, 2004. VTC2004-Fall. 2004 IEEE 60th. Vol. 2. IEEE, 2004.

[28] Vasudevan, Sudarshan, et al. "Facilitating access point selection in IEEE 802.11 wireless networks." Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement. Usenix Association, 2005.

[29] Stanley, Dorothy, Pat Calhoun, and Michael Montemurro. "Control and provisioning of wireless access points (CAPWAP) protocol specification." (2009).

[30] Cisco, “Chapter 8 - Controlling Lightweight Access Points”, http://www.cisco.com/c/en/us/td/docs/wireless/controller/7-0/configuration/guide/c70/c70lwap.html#wp1459271 retrieved on Dec. 3, 2015.

[31] Extreme Networks, “Products: Wireless Appliances”, http://www.extremenetworks.com/product/identifi-wireless-controllers# retrieved on Dec. 3, 2015.

[32] Huawei, Huawei AC6605 is a box wireless access controller, http://huaweienterpriseusa.com/ac-access-control-0/ac6605 retrieved on Dec. 3, 2015.

[33] IEEE 802.11k-2008 — Amendment 1: Radio Resource Measurement of Wireless LANs. doi:10.1109/IEEESTD.2008.4544755. ISBN 978-0-7381-5421-3.

[34] Villegas, E. Garcia, R. Vidal Ferre, and J. Paradells Aspas. "Load balancing in WLANs through IEEE 802.11 k mechanisms." Computers and

Communications, 2006. ISCC'06. Proceedings. 11th IEEE Symposium on. IEEE, 2006.

[35] Ryou, Jong Bum. Adaptive load balancing metric for wlans. Oregon State University, 2011.

[36] Puthalath, Lalith Suresh. Programming the enterprise WLAN: an SDN approach. Diss. Instituto Superior Técnico, 2012.

[37] Yen, Li-Hsing, and Tse-Tsung Yeh. "SNMP-based approach to load distribution in IEEE 802.11 networks." Vehicular Technology Conference, 2006. VTC 2006-Spring. IEEE 63rd. Vol. 3. IEEE, 2006.

[38] Velayos, Hector, Victor Aleo, and Gunnar Karlsson. "Load balancing in overlapping wireless LAN cells." Communications, 2004 IEEE International Conference on. Vol. 7. IEEE, 2004.

[39] Le, Yuan, et al. "Load balancing access point association schemes for ieee 802.11 wireless networks." Wireless algorithms, systems, and applications. Springer Berlin Heidelberg, 2011. 271-279.

[40] Sawma, Gilbert, et al. "ALBA: An autonomic load balancing algorithm for IEEE 802.11 wireless networks." Network Operations and Management Symposium, 2008. NOMS 2008. IEEE. IEEE, 2008.

[41] Collotta, Mario. "FLBA: A fuzzy algorithm for load balancing in IEEE 802.11 networks." Journal of Network and Computer Applications 53 (2015): 183-192.

[42] Yap, Kok-Kiong, et al. "The stanford openroads deployment." Proceedings of the 4th ACM international workshop on Experimental evaluation and characterization. ACM, 2009.

[43] Aruba Networks, “Aruba Networks Position Statement on CAPWAP”, http://community.arubanetworks.com/aruba/attachments/aruba/115/422/1/CAPWAP+Position.pdf retrieved on Dec. 3, 2015.

[44] ns-3 project (Network Simulator 3), “OpenFlow switch support”, https://www.nsnam.org/docs/release/3.13/models/html/openflow-switch.html, retrieved on Dec. 3, 2015.

[45] Antonenko, Vitaly, and Ruslan Smelyanskiy. "Global network modelling based on mininet approach." Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking. ACM, 2013.

[46] Open vSwitch FAQ, “Configuration problems”, http://git.openvswitch.org/cgi-bin/gitweb.cgi?p=openvswitch;a=blob_plain;f=FAQ;hb=HEAD retrieved on Dec. 3, 2015.

[47] Pack, Sangheon, et al. "Fast-handoff support in IEEE 802.11 wireless networks." Communications Surveys & Tutorials, IEEE 9.1 (2007): 2-12.

[48] Balachandran, Anand, et al. "Characterizing user behavior and network performance in a public wireless LAN." ACM SIGMETRICS Performance Evaluation Review. Vol. 30. No. 1. ACM, 2002.

[49] Lee, Jaehwan, Sunghyun Choi, and Hanwook Jung. "Analysis of user behavior and traffic pattern in a large-scale 802.11 a/b network." Proceedings of the First Workshop on Wireless Network Measurements (WiNMee 2005), Trentino, Italy. 2005.

[50] Schulz-Zander, Julius, et al. "Programmatic orchestration of wifi networks." 2014 USENIX Annual Technical Conference (USENIX ATC 14). USENIX Association, 2014.

[51] Malik, Aqsa, et al. "QoS in IEEE 802.11-based Wireless Networks: A Contemporary Survey." arXiv preprint arXiv:1411.2852 (2014)

[52] Jabri, Issam, et al. "IEEE 802.11 Load balancing: an approach for QoS Enhancement." International Journal of Wireless Information Networks 15.1 (2008): 16-30.

[53] Brickley, Olivia, Susan Rea, and Dirk Pesch. "Load balancing for QoS enhancement in IEEE802. 11e WLANs using cell breathing techniques." 7th IFIP International Conference on Mobile and Wireless Communications Networks, Maroc. 2005.

[54] Van der Schaar, Mihaela, Yiannis Andreopoulos, and Zhiping Hu. "Optimized scalable video streaming over IEEE 802.11 a/e HCCA wireless networks under delay constraints." Mobile Computing, IEEE Transactions on 5.6 (2006): 755-768.

[55] Lewin-Eytan, Liane, et al. "Designing modular overlay solutions for network virtualization." IBM Technical Paper (2012).

[56] OpenDaylight platform, MD-SAL RESTCONF API, https://wiki.opendaylight.org/view/OpenDaylight_Controller:MD-SAL:Restconf retrieved on Dec. 3, 2015.

[57] Open vSwitch, http://openvswitch.org/ retrieved on Dec. 3, 2015.

[58] OpenWrt, https://openwrt.org/ retrieved on Dec. 3, 2015.

[59] Garg, Pankaj, and Yu-Shun Wang. "NVGRE: Network Virtualization using Generic Routing Encapsulation." (2014).

[60] Mahalingam, Mallik, et al. Virtual extensible local area network (VXLAN): A framework for overlaying virtualized layer 2 networks over layer 3 networks. No. RFC 7348. 2014.