AICT 2020

The Sixteenth Advanced International Conference on Telecommunications

ISBN: 978-1-61208-802-0

September 27th – October 1st, 2020

AICT 2020 Editors

Mayank Maheshwari, Hughes Systique Corporation, India

Horia Stefanescu, Orange Romania SA, Romania

AICT 2020

Foreword

The Sixteenth Advanced International Conference on Telecommunications (AICT 2020), held between September 27 – October 1st, 2020 covered a variety of challenging telecommunication topics ranging from background fields like signals, traffic, coding, communication basics up to large communication systems and networks, fixed, mobile and integrated, etc. Applications, services, system and network management issues also received significant attention.

The spectrum of 21st Century telecommunications is marked by the arrival of new business models, new platforms, new architectures and new customer profiles. Next generation networks, IP multimedia systems, IPTV, and converging network and services are new telecommunications paradigms. Technology achievements in terms of co-existence of IPv4 and IPv6, multiple access technologies, IP-MPLS network design driven methods, multicast and high speed require innovative approaches to design and develop large scale telecommunications networks.

Mobile and wireless communications add profit to large spectrum of technologies and services. We witness the evolution 2G, 2.5G, 3G and beyond, personal communications, cellular and ad hoc networks, as well as multimedia communications.

Web Services add a new dimension to telecommunications, where aspects of speed, security, trust, performance, resilience, and robustness are particularly salient. This requires new service delivery platforms, intelligent network theory, new telecommunications software tools, new communications protocols and standards.

We are witnessing many technological paradigm shifts imposed by the complexity induced by the notions of fully shared resources, cooperative work, and resource availability. P2P, GRID, Clusters, Web Services, Delay Tolerant Networks, Service/Resource identification and localization illustrate aspects where some components and/or services expose features that are neither stable nor fully guaranteed. Examples of technologies exposing similar behavior are WiFi, WiMax, WideBand, UWB, ZigBee, MBWA and others.

Management aspects related to autonomic and adaptive management includes the entire arsenal of self-ilities. Autonomic Computing, On-Demand Networks and Utility Computing together with Adaptive Management and Self-Management Applications collocating with classical networks management represent other categories of behavior dealing with the paradigm of partial and intermittent resources.

We take here the opportunity to warmly thank all the members of the AICT 2020 Technical Program Committee, as well as the numerous reviewers. The creation of such a broad and high quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to AICT 2020. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the AICT 2020 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that AICT 2020 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in the field of telecommunications.

AICT 2020 Chairs:

AICT Publicity Chair
Mar Parra, Universitat Politecnica de Valencia, Spain

AICT 2020

COMMITTEE

AICT 2020 Publicity Chair

Mar Parra, Universitat Politecnica de Valencia, Spain

AICT 2020 Technical Program Committee

Ghulam Abbas, GIK Institute, Pakistan
Iwan Adhicandra, University of Sydney, Australia
Abdulazaz Albalawi (Aziz), University of California, Santa Cruz, USA
Michele Albano, Aalborg University, Denmark
Nicolae Dumitru Alexandru, Academy of Technical Sciences of Romania / "Gheorghe Asachi" Technical University of Iasi, Romania
Marco Aurélio Spohn, Federal University of Fronteira Sul, Brazil
Ilija Basicevic, University of Novi Sad, Serbia
Oussama Bazzi, Lebanese University, Lebanon
Carlos Becker Westphall, Federal University of Santa Catarina, Brazil
Sara Behjatjamal, Altinbas University, Turkey
Stefano Berretti, University of Firenze, Italy
Robert Bestak, Czech Technical University in Prague, Czech Republic
Antonella Bogoni, Scuola Superiore Sant'Anna-TeCIP Institute, Italy
Eugen Borcoci, University "Politehnica"of Bucharest (UPB), Romania
Christos Bouras, University of Patras, Greece
An Braeken, Vrije Universiteit Brussel, Belgium
Lubomir Brancík, Brno University of Technology, Czech Republic
Peter Brida, University of Zilina, Slovakia
Fernando Cerdan, Polytechnic University of Cartagena, Spain
Chen Chen, Chongqing University, China
Enrique Chirivella Pérez, Universitat de Valencia, Spain
Estefanía Coronado Calero, Fondazione Bruno Kessler, Trento, Italy
Kevin Daimi, University of Detroit Mercy, USA
Thomas Dreibholz, SimulaMet - Simula Metropolitan Centre for Digital Engineering, Norway
Roman Dunaytsev, Saint-Petersburg State University of Telecommunications, Russia
Ersin Elbasi, American University of the Middle East, Kuwait
Mario Ezequiel Augusto, Santa Catarina State University, Brazil
Mário Ferreira, University of Aveiro, Portugal
Gianluca Fontanesi, University College Dublin, Ireland
Wolfgang Frohberg, AKAD University Stuttgart, Germany
Ivan Ganchev, University of Limerick, Ireland / University of Plovdiv "Paisii Hilendarski", Bulgaria
Seema Garg, Nokia, India
Gelayol Golcarenarenji, University of the West of Scotland, UK
Luís Gonçalo Cancela, ISCTE-IUL & Instituto de Telecomunicações, Portugal
Iván González, University of Málaga, Spain
Norton Gonzalez, Luciano Feijão Faculty, Sobral, Ceará, Brazil

Christian Grasso, University of Catania, Italy
Jan Haase, University of Lübeck, Germany
Takeshi Ikenaga, Kyushu Institute of Technology, Japan
Ilias Iliadis, IBM Research - Zurich Laboratory, Switzerland
Vahid Joroughi, GomSpace, Denmark
Branislav Jovic, Defence Technology Agency (DTA) | New Zealand Defence Force (NZDF), Auckland, New Zealand
Georgios Kambourakis, University of the Aegean, Greece
Dimitris Kanellopoulos, University of Patras, Greece
Hamzeh Khalili, Fundació i2CAT, Barcelona, Spain
David Khoury, AUST (American University of Science &Technology), Beirut, Lebanon
Uliana Kochetkova, Saint Petersburg State University, Russia
Lida Kouhalvandi, Istanbul Technical University, Turkey
Višnja Križanović, Josip Juraj Strossmayer Unversity of Osijek, Croatia
Dragana Krstic, University of Niš, Serbia
Hoang Le, Google, USA
Gyu Myoung Lee, Liverpool John Moores University, UK
Juraj Machaj, University of Zilina, Slovakia
Zoubir Mammeri, IRIT - Toulouse, France
Kajetana Marta Snopek, Warsaw University of Technology, Poland
Alexandru Martian, University Politehnica of Bucharest, Romania
Erik Massarczyk, RheinMain University of Applied Sciences Wiesbaden Rüsselsheim, Germany
Antonio Matencio-Escolar, University of the West of Scotland, UK
Natarajan Meghanathan, Jackson State University, USA
Amalia Miliou, Aristotle University of Thessaloniki, Greece
Andrea Morichetta, University of Camerino, Italy
Ioannis Moscholios, University of Peloponnese, Greece
Marco Mugnaini, University of Siena, Italy
Karim M. Nasr, University of Greenwich, UK
Antonio Navarro, Universidad Complutense de Madrid, Spain
Claudia Cristina Oprea, University Politehnica of Bucharest, Romania
Constantin Paleologu, University Politehnica of Bucharest, Romania
Nicola Pasquino, Università di Napoli Federico II, Italy
Cathryn Peoples, Ulster University, UK
Edwige Pissaloux, Université de Rouen Normandie | LITIS & CNRS/FR 3638, France
Ladislav Polak, Brno University of Technology | SIX Research Center, Brno, Czech Republic
Emanuel Puschita, Technical University of Cluj-Napoca, Romania
Adib Rastegarnia, Open Networking Foundation, USA
Ustijana Rechkoska-Shikoska, University for Information Science and Technology "St. Paul the Apostle" - Ohrid, Republic of Macedonia
Ruben Ricart-Sanchez, University of the West of Scotland, UK
Juha Röning, University of Oulu, Finland
Zsolt Saffer, Institute of Statistics and Mathematical Methods in Economics | Vienna University of Technology, Austria
Abheek Saha, Hughes Systique Corp., India
Demetrios Sampson, University of Piraeus, Greece
Vladica Sark, IHP - Leibniz-Institut für innovative Mikroelektronik, Frankfurt (Oder), Germany
Vincent Savaux, b<>com, Rennes, France

Motoyoshi Sekiya, Fujitsu Laboratories, Japan
Sergei Semenov, HiSilicon, Sweden
Ana Serrano Tellería, University of Castilla La Mancha, Spain
Shuaib Siddiqui, Fundacio i2CAT, Barcelona, Spain
Alex Sim, Lawrence Berkeley National Laboratory, USA
Zdenek Smekal, Brno University of Technology, Czech Republic
Antonio Sorrentino, Università Parthenope, Napoli, Italy
Gautam Srivastava, Brandon University, Canada
Kostas Stamos, University of Patras, Greece
Angelo Trotta, University of Bologna, Italy
Thrasyvoulos Tsiatsos, Aristotle University of Thessaloniki, Greece
Sezer Ulukaya, Trakya University, Turkey
Rob van der Mei, Centre for Mathematics and Computer Science (CWI), Netherlands
Bernd E. Wolfinger, University of Hamburg, Germany
Ramin Yahyapour, Georg-August-Universitaet Goettingen/GWDG, Germany
Mehmet Akif Yazici, Informatics Institute - Istanbul Technical University, Turkey
Mariusz Zal, Poznan University of Technology, Poland
Stefania Zinno, University of Naples Federico II, Italy
Piotr Zwierzykowski, Poznan University of Technology, Poland

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

Wi-Fi Device Localization in an Indoor Environment Using Graph Mapping

Abheek Saha*, Mayank Maheshwari†
Hughes Systique Corporation
email:{*abheek.saha**,*mayank.maheshwari*†*@hsc.com*}

***Abstract*—Indoor location tracking is an emerging technology that enables consumer oriented businesses such as retailers and hoteliers to better track movement patterns of visitors and generate key metrics like in/out count (footfall), dwell time and understand the popular route(s) inside the facility. These are in turn used for maximizing customer safety, scheduling of optimal workforce and optimal product placements. In this paper, a method is demonstrated for user position mapping by tracking the Wi-Fi signals sent by their smartphones as they walk through an indoor environment, using only the anonymous network probing signals emitted intermittently by each device. Our approach is scalable, unobtrusive and does not require active participation or installation of any special software on tracked devices, with minimal infrastructure costs. Since the devices are not connected to any access point, the signal is anonymized, which aids in protection of user privacy. It is shown how the raw metrics are used to generate accurate position data and ensemble dynamics over a known indoor topology, and how individual models of human behaviour can be used to predict mass movement of crowds in an indoor setting.**

Keywords*—*Wi-Fi tracking; Wi-Fi Probe requests; RSSI fingerprint; group dynamics; MAC address randomization

I. Introduction

Indoor location positioning and non-intrusive tracking of users, by signals from their Wi-Fi device, is being used in various industry verticals to gain more insight about their customer behavior [1]. The insight thus obtained can be utilized to provide better customer experience. Liu et al in [2] describe various techniques that are being used for indoor location positioning and tracking. Some of the techniques make use of cellular signals for indoor location tracking (see also [3]). Other methods utilize Bluetooth(BT) or Universal Wideband (UWB) signals [4]. Accuracy of methods based on cellular technology is low (50-200 m) [2], and BT based techniques require a BT tag to be attached to tracked item. Further, BT and similar techniques have much smaller range as compared to WiFi signals and hence require a very different approach. In the rest of the paper, we will limit ourselves specifically to WiFi as a wireless access technology for our location determination purpose.

The application that we have in mind is crowd-modelling in an indoor arena. The key outcome of crowd-modelling is to know how many people are located in which part of the indoor arena, i.e., the size of the crowd and further, how the crowd is moving, within the arena. There are many different sensors available for this purpose, starting from optical processing of fixed cameras, footfall sensors, heat sensors, etc. All of them are similar in that they provide random samples of location information, which have to be converted to ensemble data. For our solution, we use Wi-Fi fingerprinting as an equivalent sensor. As we shall see, our method produces similar sampling output and our modelling approach may be used for any sensor based method. In our case the Wi-Fi signals of interest are the probes sent out by Wi-Fi end-points (typically, mobile phones carried by individuals) to detect Wi-Fi networks in the vicinity. We measure the Received Signal Strength Indicator (RSSI) for each probe received and then use a trained Machine Learning (ML) model to convert this into an indication of the zone from which the signal came. This kind of RSSI fingerprinting methods require training/calibration on each new place since no two indoor environments have the same signal propagation characteristics. While there has been a fair amount of research in this area (see Section II), to the best of our knowledge, earlier work have not considered all the other factors which impact the accuracy of location prediction, i.e., different types/make of devices, orientation of the device, etc. We shall show that all of these factors have significant impact on the accuracy of location estimation.

Typically, most systems which track Wi-Fi devices across various locations use the Wi-Fi MAC address as a device identifier. Clearly, there are privacy issues involved in tracking a device using a permanent identifier. In iOS 8, Apple introduced MAC randomization [5] to maintain user privacy during active scan for Wi-Fi networks. Since then, most of the Android phone OEMs have started doing MAC randomization. In the latest versions of Android, Google introduced support for MAC randomization in *Android open source project* (AOSP), hence covering nearly all modern devices. Even though it is possible to predict the location of the Wi-Fi device (identified by its real MAC or randomized MAC), the randomization of MAC makes tracking of a device across locations difficult, given that devices change to a new randomized MAC after transmitting few messages. Various *device fingerprinting* techniques have been identified, which utilize the information available in Wi-Fi messages (other than MAC address) [5][6]. However these techniques are better suited for identifying the type or brand of the device rather than a unique instance. Our method, on the other hand, does not require identification of individual devices; we use probe-measurements as a random sampling technique to generate ensemble location data, which is then fitted to our model.

In the rest of this paper, we use the term *Wi-Fi device* to refer to any consumer device with an active Wi-Fi interface (computers, mobile phones, tablets, etc.). We use the term *scanner* to refer to special access-points operating in monitor mode and placed at known locations, running a special application to capture Wi-Fi signals. One of the scanners is the *anchor scanner*, that is used as a reference to generate differential RSSI signals as mentioned in Section IV. Devices may either

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0

be *training devices*, used to train the location prediction engine, or *tracked devices* whose readings are captured and fed to the location prediction engine for location estimation. The input is in the form of an *RSSI fingerprint*, comprising of differential readings from multiple scanners.

The rest of the paper is organized as follows. In Section II, we cover the current state of the art in Wi-Fi based indoor location positioning . In Section III, we discuss our experiments in indoor location calibration and identify the different non-environmental factors which impact the accuracy of our method. In Section IV, we give a brief description of the Machine Learning technique used for the backend location prediction. In Section V, we show how we convert the individual location samples to a model of the indoor location as a whole. Finally in Section VI, we discuss our final results and potential for future work.

II. Literature Review

Various methods based on Wi-Fi signals are being used for the localization of devices in the indoor environment. Some of the techniques perform lateration by measuring the distance of the Wi-Fi transmitter from Wi-Fi receivers placed at known locations. The distance is measured by calculating either the *Time of Arrival (ToA)* or *Time Difference of Arrival (TDoA)* [7]. In order to give accurate results lateration techniques require time synchronization either between the Wi-Fi transmitter and receiver or between multiple receivers. These techniques also need a very accurate measurement of arrival time since a minute error in measurement leads to an error of a few 100 meters in location calculation [7]. Other techniques measure the angle of incidence (AoA) of Wi-Fi signal at two or more Wi-Fi receivers [8]. The angle of incidence is then used to determine the location of the Wi-Fi transmitter. This method requires a clear line of sight between transmitter and receiver, which is not achievable in the indoor environment for most of the use cases.

The use of RSSI measured value for location determination has been widely discussed [7][9][10][11]. The most common RSSI based techniques employ propagation loss models to measure the distance between the Wi-Fi transmitter and receiver [12]. Distance between transmitter and three or more receiver is then used to find the location of the transmitter. These techniques do not work very well because of multipath in the indoor environment, given the frequency at which Wi-Fi operates, as we shall discuss in Section III. However, there are many other factors which impact the performance. For example, in [13], the authors have described the difference in Wi-Fi transmission characteristics in different Wi-Fi devices. It should be noted that RSSI fingerprinting is being investigated for millimeter wave radio, including the new 802.11ad Wi-Fi standards [14]. The problem of converting movement of ensembles of individuals on a graph have been studied in multiple contexts. These include the movement of a fluid within tubes [15][16], the transport of particles in a network [17], diffusions on graphs with random jumps [18][19][20], etc. The key constituents of these models are the evolution function for each particle on each edge and the transition functions at the vertices, to ensure that there is no build-up between edges. A good summary is provided in [21].

III. Factors Impacting RSSI Readings In an Indoor Environment

In our solution, we have utilized RSSI based techniques for location determination; this is known as location patterning [7]. There are obvious advantages to this technique; it does not require any specialized hardware and can be implemented using standard off the shelf Wi-Fi receivers. Further, it does not require any modification of behaviour, either by the user or by the user device, since measurements are taken passively based on the normal scanning behaviour of the device; this makes the technique less obstrusive and more scalable. The core of this technique is to learn RSSI patterns (RSSI fingerprint) for each location of interest and learn how to map these patterns to transmit locations. This technique consists of two phases. In the first phase, we use labelled data to calibrate our algorithm, which also takes into account the peculiarities of the environment. In the second, we can use real-time measurements for prediction. Obviously, the placement of the scanners cannot be changed between the two phases, neither the environment.

During the calibration phase, the RSSI values of messages received by scanners are tagged by the known location of the transmitter and collated to create the RSSI fingerprint corresponding to a location. The output of this phase is an RSSI fingerprint database corresponding to each location of interest. This database is used to match received RSSI tuples during the prediction phase, as we shall see below.

A. Measurement of RSSI in a closed area

As our work is based on location patterning, using RSSI readings as the fingerprint, we wish to find all the exogenous factors which can cause variations in the RSSI readings. To this end, extensive testing of RSSI readings was done for different categories of devices in an instrumented environment. For our experimentation, we used Raspberry Pi 3 $B+$ boards with Alfa AWUS036HEH Wi-Fi USB dongles to act as Wi-Fi scanners. The Alfa AWUS036HEH Wi-Fi USB dongle was put in monitor mode to sniff Wi-Fi packets while the inbuilt Wi-Fi of Raspberry Pi provided connectivity to LAN. We used one of the floors of our office building to install Wi-Fi scanners. Wi-Fi scanners were hung from the ceiling for better signal reception. Figure 1 shows the placement of Wi-Fi scanners. We divided our office floor into zones of similar size and placed one scanner in each zone. Placement of scanners was such that it avoided concrete/metal pillars and other wireless devices. The working of the system is shown in Figure 2. An application called *find3-cli-scanner* from the opensource package [22] was installed on the Raspberry PIs to sniff Wi-Fi probe packets and forward them to the central server. For each Wi-Fi device (identified by MAC address) the central server collates RSSI values received from scanners, forming a tuple of RSSI values received from all the scanners. For example, if the RSSI value of a probe request was R_j at scanner $S_j, j \in [1..7]$ a typical RSSI fingerprint tuple with

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0

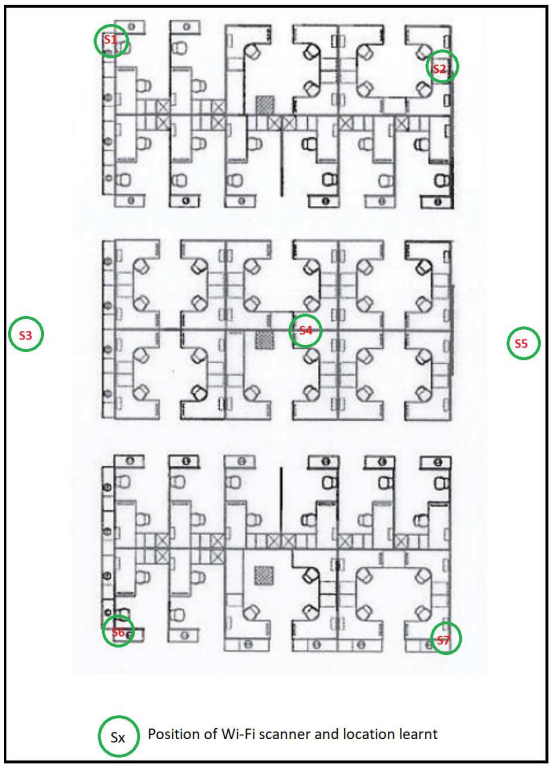


Figure 1. WiFi Scanner Placement.

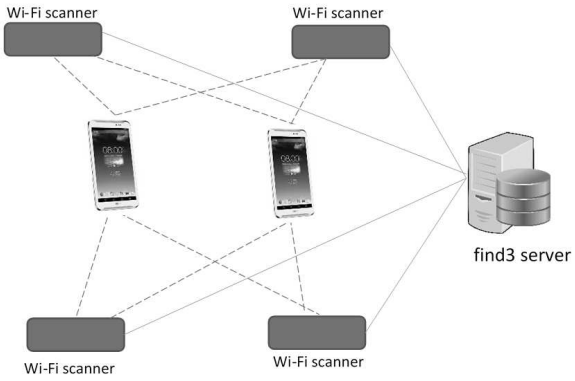


Figure 2. Test and calibration setup.

[S1:R1, S2:R2, S3:R3, S4:R4, S5:R5, S6:R6, S7:R7]

(a) Standard tuple.

[S1:R1-R4, S2:R2-R4, S3:R3-R4, S5:R5-R4, S6:R6-R4, S7:R7-R4]

(b) Differential tuple.

Figure 3. RSSI fingerprint formats.

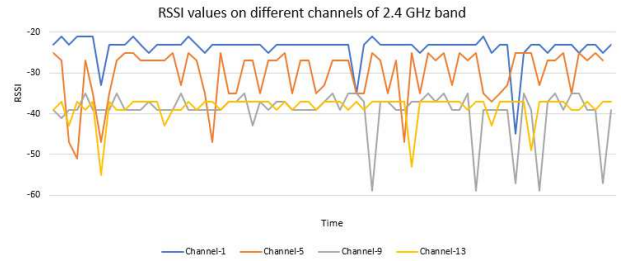
RSSI values from all 7 scanners is shown in Figure 3. For our test, we collected about 2000 RSSI fingerprints over a duration of 8 hours for the calibration phase for each learned location. Collection of RSSI fingerprints over a longer time period helps in capturing variation caused by movement of users in measurment environment. RSSI fingerprints collected during the calibration phase were used to train a ML classification algorithm. For location prediction, the RSSI fingerprint of a device is fed to the same ML classification algorithm, which predicts the probable location of the Wi-Fi device. During our initial tests with this arrangement, location predictions were not very accurate, with prediction accuracy ranging from 70% in the best case to 20% in the worst case. Where we defined accuracy as the percentage of times when system predicted the zone correctly. The reasons for accuracy variation will be explained in the Subsection III-B.

B. Factors impacting location prediction accuracy

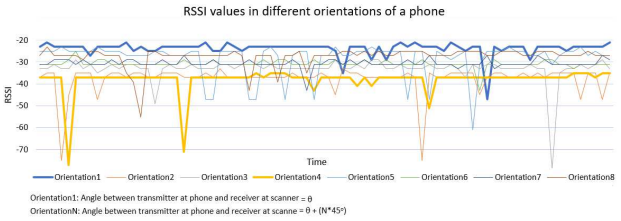
Significant variation in RSSI measurements are caused by non-environmental factors such as the channel, the orientation of the phone, device type and movement of users near the transmitting or receiving device. In this section, we shall report the outcome of experiments that we conducted on the effect of some of these factors. We start with the frequency configuration. Wi-Fi access points typically use frequency hopping, so as to reduce channel specific impairments. In our case, this means that the RSSI probe from the same location may be measured by multiple scanners on different channels, which will then be combined into the same tuple of measurements as a fingerprint. As it turns out this does not work, because different channels *even within the same 2.4Gz band* show wide variation in RSSI readings for the same scanner-transmitter pair. It is hard to say definitely whether this is because of noise in those bands or simply propagation related; however, sporadic interference can be ruled out, because the effects were sustained over a fairly large period of time. In our test, a Wi-Fi device (Moto G5 phone) was placed at a known location, and we measured the RSSI values of packets sent by this device at one of the scanners. The system was configured initially at Channel 1 (2.412 GHz), and then on channels 5(2.432 GHz), 9(2.452 GHz) and 13(2.472 GHz). Figure 4a shows the differences of RSSI values of probe requests received on different channels. There was a difference of about 20dB between the average of the RSSI values of probe requests received on channels 1 and 13.

Our next experiment considers the effect of phone orientation (angle between transmitter of phone and receiver of scanner). We conducted a test wherein one Wi-Fi device (Moto G5 phone) was placed at one of the locations, and RSSI values of its probes were captured at one of the scanners; both devices were set to channel 9. RSSI values of probe requests from Moto G5 were collected at the scanner at 8 different orientations 45° apart, for 15 minutes each. Figure 4b shows the average of the RSSI values for each orientation. The angle between Orientations 1 and 4 was 180°, and RSSI values on these two orientations differ by about 18dB. It is

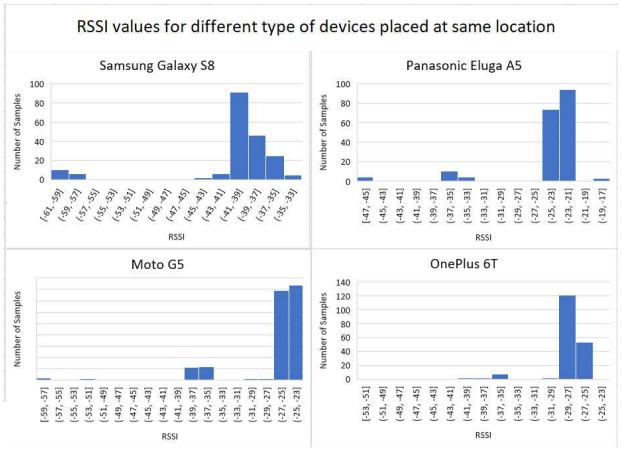
Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0



(a) Channel based.



(b) Different orientations.



(c) Different devices.

Figure 4. RSSI variations due to various factors.

clear that the RSSI values change considerably with the change in orientation. The indoor localization method should consider the possibility that the *tracked device* could be placed/carried in any possible orientation.

Finally, we consider the device itself. It turns out that if the *tracked devices* used during the location prediction phase are different from those with which the training was done, the accuracy of location prediction reduces significantly, as low as 20% at some locations. Figure 4c shows the recorded RSSI values for different Wi-Fi devices placed at the exact same location and orientation. We can see that the average RSSI value from OnePlus 6T (-27 dB) and average RSSI value from Motorola G5 (-39 dB) differ by about 12 dB. Our observations have been reproduced by other authors [13]; the difference is due to the combination of chipset, Power-Amp and antennae used by different manufactures.

IV. A Machine Learning Solution to Location Identification

Based on the factors identified in Section III-B, we have identified methods by which we can improve the quality of the RSSI measurements. For example, to take care of the channel variation, we locked all AP scanners to the same channel. If the channel is changed, all the APs must switch to the new channel and a new calibration phase has to take place. For the other factors, we introduced the concept of differential readings using one of the Wi-Fi scanners (typically the one at the center of the coverage area) as an *anchor scanner*. For creating an RSSI fingerprint, instead of using absolute RSSI value, we subtracted the RSSI value at *anchor scanner* from RSSI value at every other scanner (Figure 3b). This takes care of most of variations caused by both the chipset specific and environmental sources. Differential reading removes the differences caused by Power-Amp and antennae used in different types of devices. It is possible to designate any scanner as the anchor, and even introduce multiple anchors for additional robustness. This will increase the complexity of the training but add even more robustness to the data. We shall study this in future work. We used the Machine Learning package *FIND3* [22] for location prediction. The *FIND3* package runs multiple machine learning algorithms in parallel and then chooses the best among them using the Youden's J-statistic diagnostic metric [23] as given in equation (1). These include the K-nearest neighbour, linear SVM, Decision tree, Random Forest, and Extend Naive Bayes algorithms. Using the labeled data provided, each algorithm is trained with a subset of the data and then tested using the remaining part of the data. The prediction is in the form of a probability factor P_L for each location L. Based on the predictions by ML algorithms *Youden's J statistic* is calculated for each location and each ML algorithm.

$$
\begin{aligned}
J &= \frac{T_p}{T_p + F_n} + \frac{T_n}{T_n + F_p} - 1 \\
T_p &= \sum \mathcal{I}_{P_{L_e} > \sigma} && \textit{–True Positive} \\
F_p &= \sum \mathcal{I}_{P_L > \sigma}, L \neq L_e && \textit{–False Positive} \\
T_n &= \sum \mathcal{I}_{P_L < \sigma} \forall L \in \mathcal{L}, L_e \notin \mathcal{L} && \textit{–True Negative} \\
F_n &= \sum \mathcal{I}_{P_L < \sigma} \forall L \in \mathcal{L}, L_e \in \mathcal{L} && \textit{–False Negative}
\end{aligned}
\tag{1}
$$

In the above, σ is an externally supplied goodness-of-fit metric. A value of 1 for *J* indicates that the prediction by algorithm is perfect and the value of *0* indicates the prediction by algorithm is useless. Based on this metric and a given set of training data, the *FIND3* package will find the best fit model for location prediction. Using this algorithm, the entire calibration was done using a single device and then applied the prediction to multiple different types of devices. We collected more than 2000 samples for each location. After calibration, we performed location prediction for three different types of devices; Samsung Galaxy S8, Panasonic Eluga A5, and Motorola G5. We obtained more than 80% prediction accuracy on all of these devices, within a 3 meter radius of the calibration positions.

V. Modelling the Group Dynamics of Wi-Fi Users

Up to now, we have captured the individual probes from individual transmitters and resolved these into a location with

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0

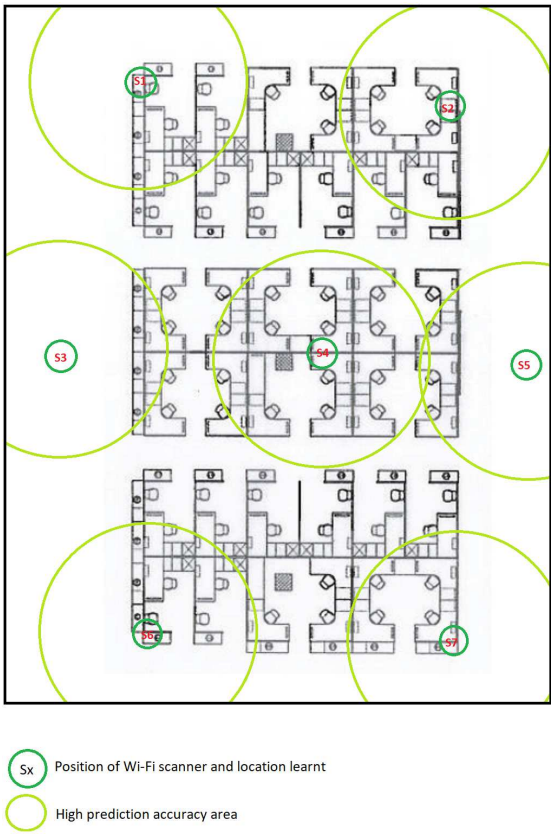


Figure 5. High Accuracy Zones.

a certain probability. In this section, we will see how we can convert the individual data-points into an aggregate model. To do this, we first use a spreading function to convert the impulse data from the samples to a continuous estimate of occupancy and then try to model this as a stochastic process. The model will allow us to predict ensemble behaviour, so that we can convert the movement of individuals into that of a crowd.

The problem of reconstructing occupancy data from sampling is made more complex by the fact that we have no control over the sampling points and further, that each brand of mobile transmits probes at different intervals as shown in Table I. The spreading function must be sufficiently broad so as to capture this variation, but not so broad that we over-value samples.

As discussed in Section IV, the machine learning algorithm gives results in terms of specific zones where the access points are centered with an effective radius of about $3m$. In order to use this, we convert the coverage region into a graph, where the zones represent edges and the vertices represent the transitions from one zone to the other (Figure 6). Each received probe, hence, has to be mapped by the location mapping algorithm into one or more zones with the associated probability of fit.

To despread the probing data and estimate the occupancy function for each zone, we use a root raised cosine spreading function $rrc(T,t,i)$ with a cutoff of 0.85 and a spreading interval of 50 seconds. The spreading function $rrc(T,t,k)$ estimates the likelihood of the transmitter being in the same zone in the time interval $[T-t, T+t]$ from which we have received a probe at time T. The computation of the occupancy function estimate $\hat{c}(l,t) : \mathcal{L} \times [0..T] \longrightarrow \mathbb{R}$ is the weighted sum of all the despread samples in that zone, as is shown in (2). Here $\mathcal{L}$ is the set of all locations within the coverage area and $s(l,\tau)$ refers to a probe request received at time τ which is resolved to be in location l with the probability $s(l,t)$.

$$\hat{c}(l,t) : \sum_{\tau} s(l,\tau) rrc(t-\tau) \tag{2}$$

Once we implement this over all the resolved probes $s \in \mathcal{S}$, we have continuous occupancy estimates $c(l,t)$ for all locations l and for all $0 \leq t \leq T$.

A. Modeling individual user behaviour

Once we have converted the empirical probe data into continuous occupancy estimates $\hat{c}(l,t)$ and the coverage area into a graph $\Gamma = \{\mathcal{N}, \mathcal{V}\}$, we now have to choose a model to fit the empirical data. The baseline assumption is that all the users are homogeneous and the basic individual model is only affected by the edge (location) of the transmitter and the position within that edge. Within the evolution function, we have to identify the parameter which captures the effect of the environment on the particle. If we model the network as a series of pipes and the individuals within it as a frictionless fluid, then the cross-section of each pipe (edge) determines the transport rate within that edge. The entire system of equations must then be solved simultaneously, taking into account the topology of the graph, i.e., the number of edges coming together at each vertex.

In our case, the movement of individual users is modelled by an Ito Diffusion $dx_t = b(x)dt + \sigma(x)dW_t, x(0) = a$ with the $b(), \sigma()$ obeying the usual Lipschitz conditions and W_t being a standard Brownian motion. The diffusion captures both the variability of the data and the drift of the user within a given edge. A key statistic is the *exit* process, which is defined as follows: given that there are N transmitters in a given edge, moving as per a given process, what is the likelihood that the transmitters will exit the edge at a given vertex within the next T_e time period. If we can compute the probability function $P(T_e)$ for the exit time on the eth edge, we can predict the flow of movement from within an edge to the neighbouring edges. A second interesting statistic is the transition process, which determines the user behaviour when she reaches a vortex and has to choose among the edges meeting at the vortex. In this paper, we focus on the exit time. For a diffusion starting from any point a within the known domain, the exit time is given by Dynkin's formula (3).

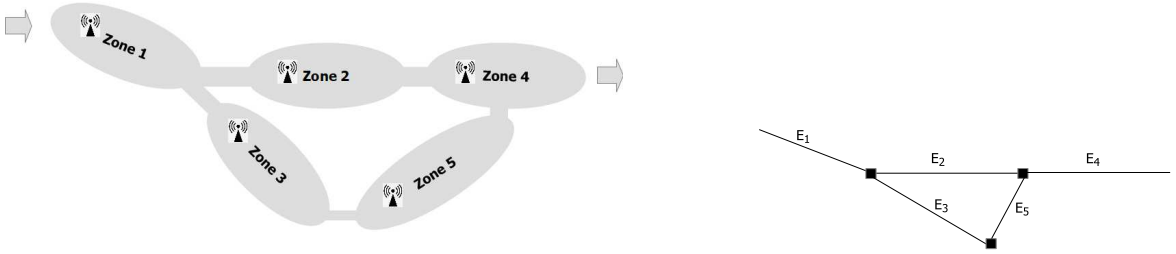
$$\mathbb{E}^a(f(X_t)) = f(a) + \int_0^t \mathcal{A}f(x)ds \tag{3}$$

$$\mathcal{A}f = b(x)\partial_x f + \frac{1}{2}\sigma^(x)\partial_x^2 f \tag{4}$$

By setting boundary conditions $f(a) = a$, $f(1) = 0$, we can convert the above to an ordinary partial differential equation $b(x)\partial_x u(x) + 1/2\sigma^2(x)\partial_x^2 u(x) + a = 0$, which is solvable using standard techniques. Once we know the function $a(x)$, we can predict the movement of users from one edge to the other.

TABLE I: PROBE REQUEST TRANSMISSION INTERVAL IN SECONDS

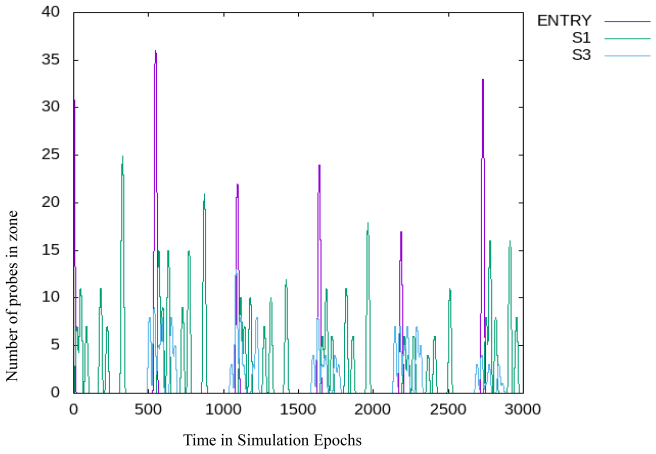
Phone	Phone Screen On			Phone Screen Off			Wi-Fi settings screen open		
	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min
Samsung Galaxy S8	383	702	96	420	787	120	10	10	10
Motorola G5	124	210	49	1020	1920	286	8	3	11
Samsung Galaxy Tab 4	130	130	130	380	600	120	10	10	10
Panasonic Eluga A5	133	268	15	Does not transmit when screen is off			9	11	8



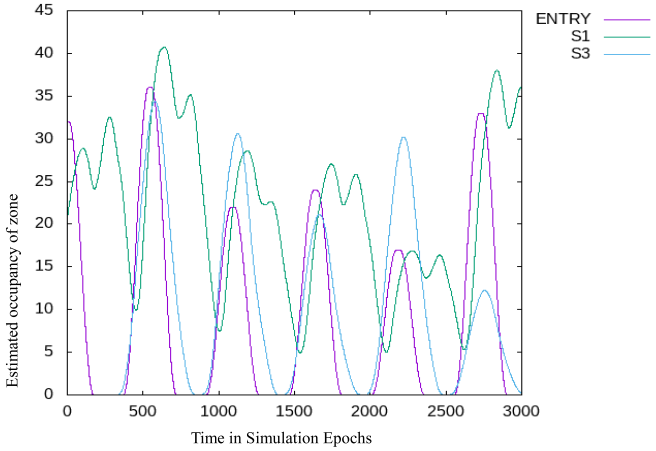
(a) Mapping of topology.

(b) Conversion to graph.

Figure 6. Topological mapping of user locations to a graph.



(a) Probe Data - raw.



(b) Probe data - Fully despread.

Figure 7. Effect of de-spreading on raw probe data.

In our simulation, we use a Brownian Bridge as a model for the movement of individual users within each edge, as shown in (5), where each edge is mapped onto $[0..1]$. The edge specific drift rate γ^i. The advantage of the Brownian bridge is that each user is guaranteed to exit the edge at time γ^i, since $\lim_{t\to\gamma^i} x_t = 1$. Note that x_t^i is normalized with respect to hypothetical length of the edge L^i. The second term is the transition condition at each vertex ($t = 1$), where $\mathcal{I}_j$ is the set of vertices which meet at the jth vertex. The entire set of equations has to be solved for all the edges simultaneously, with the transition conditions providing the boundary value functions.

$$dx_t^i = \frac{1 - x_t}{\gamma^i - t} dt + dW_t, c_0^i = 0 \quad (5)$$

$$\sum_{i \in \mathcal{I}_j} c_1^i = 0 \quad (6)$$

B. Simulation and mass dynamics

While diffusions on graphs can be solved numerically [24], or by using vanishing viscosity techniques [18], we opt to use a simulation method. We seed the prediction by taking a snapshot of the occupancy data at a time $t = 0$ and then use our model to predict the expected $c_t^i \forall i$ edges. The corresponding estimated distribution $p(c(T))$ is compared with the actual empirical distribution $\hat{c}(T)$ to get an idea of how close our model is to reality.

A key metric is the correlation of the modelled occupancy for edges m and n adjacent to each other in the graph 6b. If $c^i(t)$ captures the estimated occupancy of the ith edge,$0 \leq t \leq T$ and the edges m, n are adjoining with a drift rate of $1/\gamma^i$ we should see a cross correlation peak for $c^m(t)c^n(t - 1/\gamma^i)$. In figure 8, we have plotted the correlation between adjacent edges. The first curve shows the correlation of the empirical data and the second shows the correlation peaks for the modelled data. We note that the two curves are in relative match with each other, with the gap between successive peaks capturing the drift from one branch to the other.

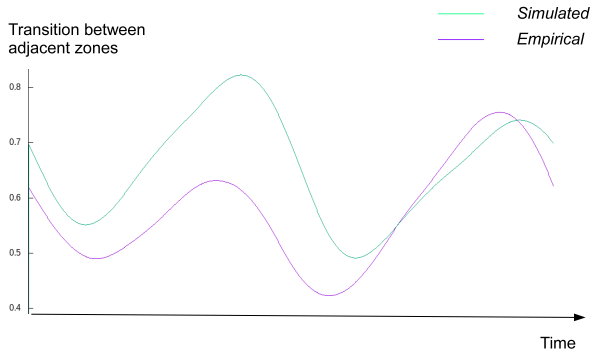


Figure 8. Transition of crowd across adjacent edges measured using estimated occupancy.

VI. Conclusion and future work

In this paper, we have considered the problem of localization of users in an indoor environment of known topology, using only the probes transmitted by their Wi-Fi enabled devices. By using differential measurements, we have shown that a machine learning solution can accurately pinpoint the location within given zones with 80% accuracy, without requiring any kind of user tracking. Further, we have shown the use of empirical measurements to reconstruct the group-dynamics of the ensemble user population by modelling the behaviour of the individuals as diffusions on a graph. The results show that is possible to use the observed mass dynamics of users to derive the individual models of user movements within zones. Our basic approach of using Machine Learning to map between RSSI fingerprints and transmitter locations is a domain of active research. In this paper, we have used the ML algorithm from *FIND3*, without any significant re-architecting. In future work, we would refine the algorithm to take into account our particular use case. One of the challenges for FIND3 (as with many other algorithms) is that it cannot handle incomplete input. In our situation, this means that probe messages which are not picked up by all scanners cannot be processed at all and have to be discarded. In a large indoor arena, it is impractical to expect all probe messages to be picked up by all scanners. Further, like all ML based algorithms, the questions of stability, accuracy and computational requirement require further work. Retraining of the ML for each change in interior topology is CPU intensive and slow; hence, we would like to find ways to augment existing algorithms for minor changes, rather than retrain the entire ML. This is under active consideration. For the group dynamics part, we intend to focus on better models of user behaviour and better metrics for capturing mass dynamics as measured through empirical data. This will help us to create more accurate models of user behaviour which can be validated empirically.

References

[1] D. Goodin., "No, this isn't a scene from minority report. this trash can is stalking you." ars Technica, Tech. Rep., Sep. 2013. [Online]. Available: https://arstechnica.com/information-technology/2013/08/no-this-isnt-a-scene-from-minority-report-this-trash-can-is-stalking-you/ [accessed:2020-08-01].

[2] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 37, no. 6, Nov. 2007, pp. 1067–1080.

[3] V. Otsason, A. Varshavsky, A. LaMarca, and E. de Lara, "Accurate gsm indoor localization," in UbiComp'05 Proceedings of the 7th international conference on Ubiquitous Computing, Tokyo, Japan, Sep. 2005, pp. 141–158.

[4] S. Gezici and H. V. Poor, "Position estimation via ultra-wide-band signals," Proceedings of the IEEE, vol. 97, no. 2, 2009, pp. 386–403.

[5] J. Martin and et al., "A study of mac address randomization in mobile devices and when it fails," in Proceedings on Privacy Enhancing Technologies, 2017, pp. 365–383.

[6] M. Vanhoef, C. Matte, M. Cunche, L. Cardoso, and F. Piessens., "Why mac address randomization is not enough: An analysis of wi-fi network discovery mechanisms." in In ACM Asia CCS,, May 2016, pp. 413–424.

[7] Cisco, "Wi-fi location-based services, 4.1 design guide." Cisco, Tech. Rep., May 2008. [Online]. Available: https://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Mobility/WiFiLBS-DG.html/ [accessed:2020-08-01].

[8] J. Xiong and K. Jamieson, "Arraytrack: a fine-grained indoor location system," in in Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13), 2013, pp. 71–84.

[9] M. A. Youssef, A. Agrawal, and A. U. Shankar, "Wlan location determination via clustering and probability distributions," in Proceedings of the First IEEE International Conference on Pervasive Computing and Communications,, March 2003, pp. 143–150.

[10] M. Youssef and A. Agrawal, "The horus wlan location determination system," in ACM Proceedings of the 3rd international conference on Mobile systems, applications, and services,, June 2005, pp. 205–218.

[11] P. Bahl and V. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in Proceedings IEEE INFOCOM, 2000, pp. 143–150.

[12] P. Kumar, L. Reddy, and S. Varma, "Distance measurement and error estimation scheme for rssi based localization in wireless sensor networks," in IEEE Conference on Wireless Communication and Sensor Networks,, 2009, pp. 1–4.

[13] G. Lui, T. Gallagher, B. Li, A. G. Dempster, and C. Rizos, "Differences in rssi readings made by different wi-fi chipsets: A limitation of wlan localization," in IEEE International Conference on Localization and GNSS (ICL-GNSS),, 2011, pp. 53–57.

[14] T. Nitsche and et al., "Ieee 802.11ad: directional 60 ghz communication for multi-gigabit-per-second wi-fi [invited paper],," IEEE Communications Magazine, vol. 52, no. 12, December 2014, pp. 132–141.

[15] G. Coclite and M. Garavello, "Vanishing viscosity for traffic on networks," Siam Journal of Mathematical Analysis, vol. 42, no. 4, 2010, pp. 1761–1783.

[16] Y. Achdou, F. Camilli, A. Cutri, and N. Tchou, "Hamilton-jacobi equations on networks," in Proceedings of the 18th World Congess, The International Federation of Automatic Control,, 2011.

[17] S. Albeverio and S. Kusuoka, "Diffusion processes in thin tubes and their limits on graphs," The Annals of Probability, vol. 40, no. 5, 2012, pp. 2131–2167.

[18] F. Camilli, C. Marchi, and D. Schieborn, "The vanishing viscosity lime for hamilton-jacobi equations on networks," Journal of Differential Equations, no. 254, 2013, pp. 4122–4143.

[19] M. I. Freidlin and S.-J. Sheu, "Diffusion processes on graphs: stochastic differential equations, large deviation principle," Probability Theory and Related Fields, 2000, pp. 181–220.

[20] M. I. Freidlin and A. D. Wentzell, "Diffusion processes on graphs and the averaging principle," Annals of Probability, vol. 21, no. 4, 1993, pp. 2215–2245.

[21] Y. V. Pokornyi and A. Borovskikh, "Differential equations on networks (geometric graphs)," Journal of Mathematical Sciences, vol. 119, no. 6, 2004, pp. 691–718.

[22] find3, "The framework for internal navigation and discovery," 2018. [Online]. Available: https://github.com/schollz/find3-cli-scanner/ [accessed:2020-08-01].

[23] W. J. Youden, "Index for rating diagnostic tests," Cancer, vol. 3, no. 1, 1950, pp. 32–35.

[24] M. K. Fijavz, D. Mugnolo, and E. Sikolya, "Variational and semigroup methods for waves and diffusion in networks," Applied Mathematics and Optimization, vol. 55, no. 2, 2007, pp. 219–240.

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0

Charms and Virtual Network Functions Primitives Experiments using Open Source MANO Framework

Andra Ciobanu ,Cosmin Conțu, Eugen Borcoci
Telecommunications Department
University POLITEHNICA of Bucharest – UPB
Bucharest, Romania
Emails: andraciobanu90@yahoo.com, cosmin.contu@elcom.pub.ro, eugen.borcoci@elcom.pub.ro

***Abstract*—Network Function Virtualization (NFV) is a strong technology to support the development of flexible and customizable virtual networks in multi-tenant and multi-domain environment. Open issues still exist for architectural, interoperability, design and implementation aspects. The contributions of this paper consist in introducing Open Source MANO OSM framework, present deeper knowledge about Virtual Network Function (VNF) charms and primitives. The objective of this deeper knowledge is accomplished with a proxy charm experiment. Also, a bug fix for proxy charm is presented and other capabilities and possible limitations in different contexts are going to be examined and sorted out for the future works. Towards this aim, this paper continues the authors work and develops new functionalities along with the previous ones and integrates them in a complex network topology using OSM. This work can help the developers implementing NFV systems based on OSM**

Keywords—VNF;OSM;Charm;Primitives;Bug

I. INTRODUCTION

Telecommunication infrastructures consist of a myriad of technologies from specialized domains such as radio, access, transport, and core and (virtualized) data center networks. Designing, deploying and operating end-to-end services are commonly manual and long processes performed via traditional Operation Support Systems (OSS) resulting in long lead times (weeks or months) until effective service delivery [2]. Moreover, the involved workflows are commonly hampered by built-in hazards of infrastructures strongly coupled to physical topologies and hardware-specific constraints.

Technological advances under the ages of *Software Defined Networking (SDN*) [3] and *Network Function Virtualization (NFV)* [3] bring new ways in which network operators can create, deploy, and manage their services. SDN, NFV, and cloud computing technologies are powerful tools enabling services, and systems to meet certain objectives (e.g., a customer requesting a specific network service). Altogether, the process shall be timely, consistent, secure, and lead to cost reduction due to automation and virtualization. We refer to *Network Service Orchestration (NSO)* as the automated management and control processes involved in services deployment and operations performed mainly by telecommunication operators and service providers [4].

However, to realize this paradigm, there is a need to model the end-to-end (E2E) service and have the ability to abstract and automate the control of physical and virtual resources delivering the service. The coordinated set of activities behind such process is commonly referred to as orchestration.

In this paper, another orchestration framework has been studied and used, i.e., Open *Source MANO (OSM).* The reason for this is that SONATA framework and project itself it is a part of entire OSM and is not going to be treated alone anymore in the future, but as an integrated part of OSM.

OSM is an ETSI-hosted open source community delivering a production-quality Management and Orchestration (MANO) stack for NFV, capable of consuming openly published information models, available to everyone, suitable for all Virtual Network Functions (VNFs), operationally significant and Virtual Infrastructure Management (VIM)-independent. OSM is aligned to NFV ISG information models while providing first-hand feedback based on its implementation experience [5].

The main purpose of this paper is to continue with development of previous experiments [1][9], but which are now based on OSM framework in order to understand the capabilities of the framework, and to develop and test some custom VNFs and service chains. Previous experiments were presented in coauthor's papers [1], and they consist in VNF and network services development with SONATA and OSM frameworks.

The paper is organized as follows. Section II is an overview of related work and a short high-level overview of the OSM. Section III presents a selective view in explanation of "day 0, day 1, day 2 VNF configurations and concepts, VNF primitives and charms, all of them integrated with OSM framework. Section IV contains the results of the charm experiments done with OSM framework and all the steps taken. Section V presents conclusions and future work.

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0

II. Related Work-OSM Architecture and Functions

This section shortly presents a selective view on an open-source solution and some related work dedicated to service development and orchestration in virtualized networks and its relation to OSM architecture, when applicable.

The goal of ETSI European Telecommunications Standards Institute (ETSI) OSM is the development of a community-driven production-quality E2E Network Service Orchestrator (E2E NSO) for telco services, capable of modelling and automating real telco-grade services, with all the intrinsic complexity of production environments. OSM provides a way to accelerate maturation of NFV technologies and standards, enable a broad ecosystem of VNF vendors, and test and validate the joint interaction of the orchestrator with the other components it has to interact with: commercial NFV infrastructures (NFVI+VIM) and Network Functions (either VNFs, Physical Network Functions- PNFs or Hybrid ones).

OSM's approach aims to minimize integration efforts thanks to four key aspects:

A well-known *Information Model (IM),* aligned with ETSI NFV, that is capable of modelling and automating the full lifecycle of Network Functions (NF) (virtual, physical or hybrid), Network Services (NS), and Network Slices (NSI), from their initial deployment (instantiation, Day-0, and Day-1) to their daily operation and monitoring (Day-2). OSM's IM is completely infrastructure-agnostic, so that the same model can be used to instantiate a given element (e.g., VNF) in a large variety of VIM types and transport technologies, enabling an ecosystem of VNF models ready for their deployment everywhere.

OSM provides a *unified northbound interface (NBI),* based on NFV SOL005, which enables the full operation of system and the Network Services and Network Slices under its control. In fact, OSM's NBI offers the service of managing the lifecycle of Network Services (NS) and Network Slices Instances (NSI), providing as a service all the necessary abstractions to allow the complete control, operation and supervision of the NS/NSI lifecycle by client systems, avoiding the exposure of unnecessary details of its constituent elements.

The OSM extended the concept of "Network Service" , so that an NS can span across the different domains identified like virtual, physical ones or technological like Radio access network (RAN) core, and transport networks. Therefore, it is possible to control the full lifecycle of an NS interacting with VNFs, PNFs and Hybrid Network Functions (HNFs) in an undistinguishable manner along with on demand transport connections among different sites. In addition, OSM can also manage the lifecycle of Network Slices, assuming when required the role of Slice Manager, extending it also to support an integrated operation [6].

III. VNF Configurations and Concepts, VNF Primitives and Charms

The OSM's approach aims to minimize integration efforts, so a well-known Information Model (IM), aligned with ETSI NFV is capable of modelling and automating the full lifecycle of Network Functions.

Virtual Network Function Descriptor [VNFD] defines the resources required for realizing the VNF. The descriptor includes various components that are part of the VNF. In addition, it also defines the VNF level configuration information.

The VNFD connects *Virtual Deployment Units (VDUs)* using the internal *Virtual Links (VLs)*. Each VDU represents a Virtual Machine (VM)/Container. The following diagram from Figure 1 illustrates the internal structure of VNFD.

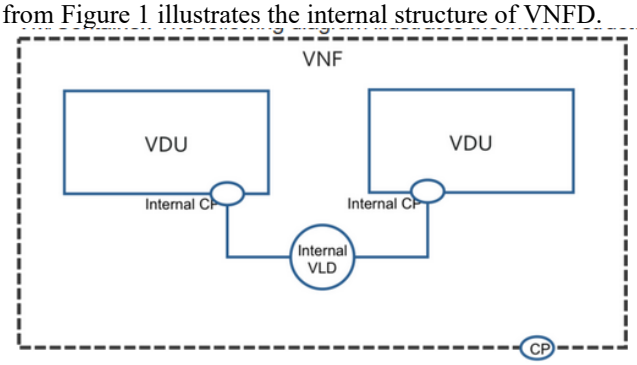


Figure 1 Virtual Network Function Descriptor [7]

The VDUs attach to the internal VLs using the internal *Connection Points (CPs).* So, the VNFD captures the list of VDUs and the internal VLs that connect the VDUs [7]. NFV promises to go from traditional network management to native NFV management, with highly efficient automation and operation.

In order to fulfill the complete onboarding process, a VNF Package will be produced and it will be part of the OSM catalogue for its inclusion in a Network Service. The onboarded VNF should aim to fulfil the lifecycle stages. It requires to function properly, for the NFV MANO layer to be able to automate. To accomplish the lifecycle stages, the resulting package includes all the requirements, instructions and elements which are: basic instantiation (a.k.a. "Day0"), service initialization (a.k.a. "Day1") and runtime operations (a.k.a. "Day2") – see Figure 2.

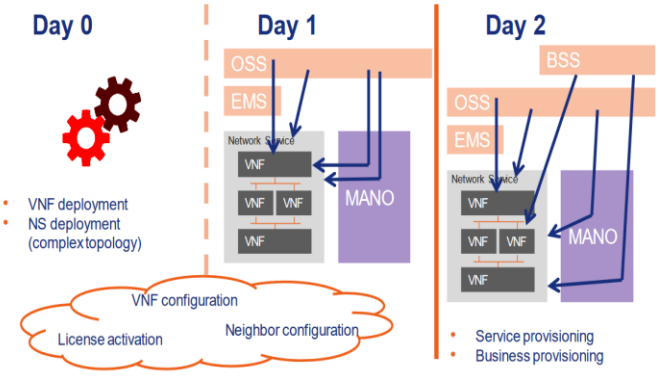


Figure 2 High Level Overview Day 0, 1 , 2 VNF [7]

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0

In **Day0** stage, the VNF is instantiated and the management access is established so that the VNF can be configured at a later stage. Furthermore, *cloud-init* files can be used to inject this minimal configuration to the VNF.

The main objective of the **Day1** stage is to configure the VNF so it starts providing the expected service. The service is defined in the initial configuration which will run automatically after the VNF in instantiated.

The main objective of **Day2** is to be able to re-configure the VNF so its behavior can be modified during runtime. In Day2, main Key Performance Indicators KPIs and their run scaling actions can be monitored.

Juju is a generic Virtual Network Function Manager (VNFM) in the ETSI NFV architecture [7]. Juju is a universal service modelling system; it models services, their relationships and scale, independent of substrate (cloud, virtualized or physical). **Juju** is an open source modeling tool, composed of a controller, models, and charms, for operating software in the cloud. It can handle configuration, relationships between services, lifecycle and scaling, which ensures that common elements such as databases, messaging systems, key value stores, logging infrastructure and other 'glue' functions are available as charms for automatic integration, reducing the burden on vendors and integrators.

A **charm** is a collection of actions and hooks that encapsulate the operational logic of an application. A charm is a piece of software that runs scripts over some targets. Traditionally the charms wrote in Juju are used inside an application or in the same machine as an application.. Charms make it easy to reliably and repeatedly deploy applications, and then scale them as required with minimal effort.

Charmed OSM is an Open Source MANO distribution, developed and maintained by Canonical, which uses Juju charms to simplify its deployments and operations. Charmed OSM enables Telecommunication Service Providers (TSPs) to easily deploy pure upstream Open Source MANO in highly available, production-grade and scalable clusters. Hooks manage the lifecycle events of an application, from installation, configuration (day-0), and scaling, in a repeatable and reliable way. Actions are on-demand functions that can handle day 1 and day 2 configuration. Type of charm depends on type of Network Function, as it can be seen in Figure 3.

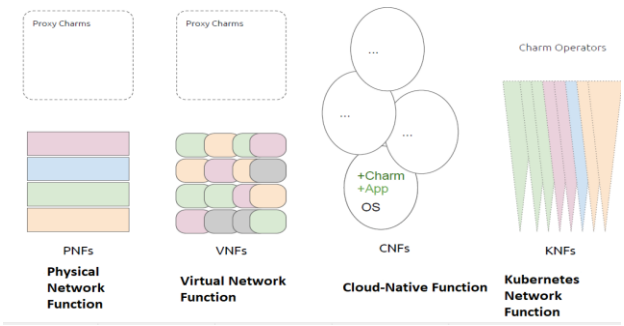


Figure 3. Generation of Network Function with OSM [7]

Types of Charms: Proxy – the focus on this paper

- Used for Physical and Virtualized Network Functions
- Runs in a Linux LXD container, separate from the VNF
- Only handles day 1 and day 2 configuration
- Usually communicates with VNF via SSH

Types of Charm: Machine

- Used for Cloud-native Network Functions (CNF) CNFs are like VNFs, but they run on lighter-weight containers, providing greater agility and ease of deployment compared with VMs
- Runs on the same machine as the VNF

Types of Charm: KNF

- Used for *Kubernetes Network Functions*

Charm runs as Operators in Kubernetes and manages lifecycle and day 0, day 1 and day 2 configuration.

VNF primitives in OSM are declared in the VNF Descriptor and their initial-config-primitive (Day-1) is invoked by the LCM at instantiation time. This is where the special 'config' primitive is invoked, setting ssh credentials. The config-primitive (Day-2) is invoked by the LCM at operator demand (or demanded through the NBI e.g., from an OSS). These primitives are a 1:1 map to a charm action or the 'config' hook.

IV. PROXY CHARM BUILD EXAMPLE IN OSM AND BUG FIX EXAMPLE

As opposed to classical "Native charms", **Proxy charms** run from outside the application. In particular, it run within a model instantiated in a LXC container that configures the VNFs through their management interface, as it can be observed in Figure 4. Proxy charms cover day-1 and day-2 configuration [8].

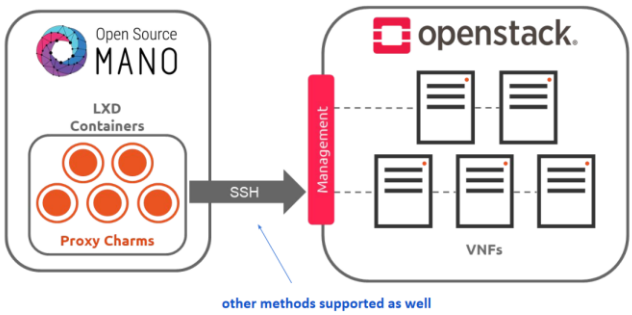


Figure 4. Proxy Charms [8]

The steps needed to build a proxy charm are the following:

1. *1) Setting up a charming environment (sudo snap install charm --classic # already installed in using shared OSM)*
   a) Create needed directories for building the charm
   b) Juju and charms environment variables

2) Creating a Proxy charm layer

a) Metadata.yaml includes all the high level information of our charm

b) Layer.yaml states all the layers on which our layer is based

c) Actions.yaml contains the high level description of the actions that will be implemented in the charm

d) Reactive/simple.py contains the actual code of the Proxy charm

3) Implementing the action

a) Append the implementation of the action to reactive/simple.py

Further on, the objective is to provide the guidelines for including all the necessary elements in the VNF Package and to provide the guidelines for including all the necessary elements in the VNF Package for its successful instantiation and management setup, so it can be further configured at later stages. The way to achieve this in OSM is to prepare the descriptor so that it accurately details the VNF requirements, prepare cloud-init scripts (if needed), and identify parameters that may have to be provided at later stages to further adapt to different infrastructures.

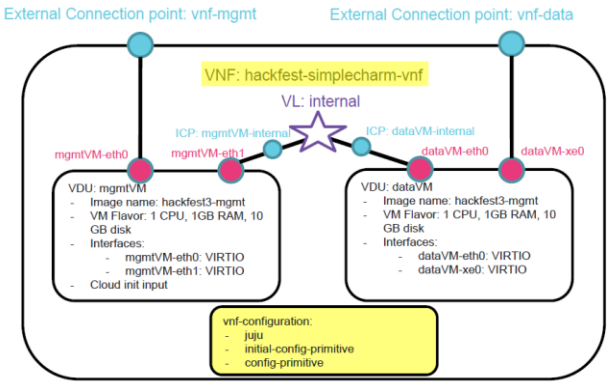


Figure 5. Practical example of adding charm [8]

The initial-config-primitive section takes care of **Day-1**

- The seq section states the order in which the initial config primitives will be called.
- The Proxy charm has ssh access to the VNFs thanks to the config primitive.
- The touch primitive is our Day-1 action created in the simple charm

The config-primitive section contains the available on-demand actions for **Day-2**.

Day-2 primitives are actions invoked on demand, so the config-primitive block is used instead of the initial-config-primitive block at the VNF or VDU level.

Proxy charms for implementing Day 2 primitives are built exactly in the same way as when implementing Day 1 primitives. In this stage, at day2 different monitoring parameters can be added:

- Collecting NFVI metrics
- Collecting VNF indicators
- Adding scaling operations

During charm implementation, some problems or bugs may occur. A common one can appear when the proxy charm presented above has been added at day-1 stage at the ssh access. The contribution of the paper has been to develop a python script in order to ensure that the workload status is set to active only when SSH proxy is properly configured.

The main principal flow is that charms would set their own workload status. The flow for status is the following:

- the charm start out and sets “maintenance”
- during its scope, it stuff up in “active” when the workload is ready
- or if there is any probem which needs intervention, it sets “blocked” status.

Charms, in general provide layers so that someone can build a proxy charm and evaluate a risk. A piece of software is valid for a charm and it can be reused-that is why layers are used. One of the layers is the “sshproxy” layer that includes ways to connect automatically to any VNF through SSH.

The implemented script scope is to set the proxy charm’s state to active so the LCM knows it is ready to work only when SSH proxy is ok configured.

As it can be seen in Figure 6, the first part invokes the “reactive/simply.py” code, then it sets active status when ssh is configured, then, the last step is to map the action to the commands to be run.

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0

```
from charmhelpers.core.hookenv import (
    action_get,
    action_fail,
    action_set,
    status_set,
)
from charms.reactive import (
    clear_flag,
    set_flag,
    when,
    when_not,
)
import charms.sshproxy


@when('sshproxy.configured')
@when_not('simple.installed')
def install_simple_proxy_charm():
    """Post-install actions.

    This function will run when two conditions are met:
    1. The 'sshproxy.configured' state is set
    2. The 'simple.installed' state is not set

    This ensures that the workload status is set to active only when the SSH
    proxy is properly configured.
    """
    set_flag('simple.installed')
    status_set('active', 'Ready!')


@when('actions.touch')
def touch():
    err = ''
    try:
        filename = action_get('filename')
        cmd = ['touch {}'.format(filename)]
        result, err = charms.sshproxy._run(cmd)
    except:
        action_fail('command failed:' + err)
    else:
        action_set({'output': result})
    finally:
        clear_flag('actions.touch')
```

Figure 6. Fixed typo in simple proxy charm

CharmHelpers provides an opinionated set of tools for building Juju charms. This script imports actions from charmHelpers, reactive and ssh.proxy,then creates a function that mentains the workload status of the proxy charm when SSH is configured.

V. CONCLUSIONS AND FUTURE WORK

This paper continued the work from our previous one [9] and got deeper into the structure of a VNF and network service from descriptors creation until management setup and instantiation. The chosen framework is OSM, which is an ETSI-hosted project to develop an Open Source NFV Management and Orchestration software stack aligned with ETSI NFV. The reason why SONATA wasn't used in this case is that it belongs to OSM community and is not so modular as OSM framework is.

As future work, new experiments will be done in OSM and contact with OSM community will be maintain in order to adjust and address possible other bugs or issues with charms and primitives.

REFERENCES

[1] A. Ţapu, C. Conţu and E. Borcoci, "Multiple Chained Virtual Network Functions Experiments with SONATA Emulator", The 12th International Conference on Communications 2018.

[2] http://www.blueplanet.com/products/multi-domain-service-orchestration.html accessed 2020-10/09

[3] D. Kreutz et.al.,"Software-Defined Networking: A Comprehensive", Survey. Proceedings of the IEEE 103, 14{76. URL: http://ieeexplore.ieee.org/document/6994333/, doi:10.1109/JPROC.2014.2371999,2015.

[4] R. Mijumbi et.al., "Network Function Virtualization: State-of-the-Art and Research Challenges", IEEE Communications Surveys & Tutorials 18, 236{262. URL: http://ieeexplore.ieee.org/document/7243304/, doi:10.1109/COMST.2015.2477041,2015.

[5] https://osm.etsi.org/wikipub/index.php/OSM_Release_FIVE.

[6] https://osm.etsi.org/docs/user-guide/02-osm-arhitecture-and-functions.html, retrived on 2020.

[7] https://osm.etsi.org/wikipub/images/0/07/Introduction_to_OSM_Zero_Touch_Carrier_Autmation_Congress_FJ.pdf, retrived on 2020.

[8] http://osm-download.etsi.org/ftp/osm-6.0-six/8th-hackfest/presentations/8th%20OSM%20Hackfest%20-%20Session%207.1%20-%20Introduction%20to%20Proxy%20Charms.pdf, retrived on 2020.

[9] A.Tapu, C.Conţu and E. Borcoci " Study on Use-Cases of Open Source Management and Orchestration Framework in 5G Projects", The Nineteenth International Conference on Networks ICN 2020.

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0

5G Programmable Infrastructure Orchestration Using ONAP

Horia Ştefănescu, Marius Iordache, Bogdan Rusti, Cătălin Brezeanu, Jean Ghenţa
Technology Department
Orange Romania
Bucharest, Romania
e-mail: Horia.Stefanescu@orange.com, Marius.Iordache@orange.com, Bogdan. Rusti@orange.com, Catalin Brezeanu@orange.com, Jean.Ghenta@orange.com

Michał Chabiera*, Łukasz Rajewski*†, Grzegorz Panek*†
*Orange Polska,
†Warsaw University of Technology,
Warsaw, Poland
e-mail: first.second@orange.com

***Abstract* - Telco operators are currently reinventing their operational model by adopting agile principles and making the shift from hardware centric towards software centric networks. 5G is not coming only with a very advanced radio layer, but also with the evolutionary concept in which the network is programmable and accessible through Application Programming Interfaces (APIs). This stands as a game changer and it will allow verticals (e.g., transport, media, energy) to seamlessly integrate their applications within the 5G ecosystem. This paper presents a programmable infrastructure leveraging Open Network Automation Platform (ONAP) capabilities developed within two European projects 5G-EVE and 5G-VICTORI. The focus is on presenting the work performed for Virtual Network Functions (VNFs) onboarding, deployment and in life management together with the 5G slicing capabilities for Radio Access and Core networks.**

Keywords-ONAP; programmable infrastructure, VNF; vRAN; vEPC; Openstack.

I. INTRODUCTION

The evolution from 4G to 5G is disruptive for the telco operators as they do not only need to embrace new concepts such as software centric networks, but also to change their operational model completely which has been effectively functional for decades.

The new software centric network model will bring two main benefits such as:

1. Being able to apply DevOps principles in the telco world, therefore facilitating a faster pace in developing and implementing new functionalities or integrating verticals.
2. End-to-end automation of day 0, 1 and 2 operations considering cognitive Artificial Intelligence (AI) enhanced capabilities, which will truly simplify operations.

In order to benefit from these advantages, the telco operator will need to adapt its operating model from the traditional one, in which engineering and operation are separated to a DevOps one. This is not simple as it will require update in all the existing processes, which will lead to a serious organizational transformation. Nevertheless, the telco operators are also aware of the high complexity of the ecosystem. Firstly, this will require more advanced skills from the network engineers and secondly, as more frequent operations will be performed, it will be more prone to errors and incidents, therefore strict control mechanism should be enforced. At the same time, the telco operator will need to adapt and evolve its infrastructure, making it fully programmable. Besides costs, this comes with a lot of technological changes, as the existing model based on monolithic equipment is disrupted.

This paper presents the technical capabilities of the programmable infrastructure deployed within 5G-EVE[1] and how it will further be utilized by the vertical use cases from 5G-VICTORI [2]. At the time of writing this paper, the programmable infrastructure allows several Virtual Network Functions (VNFs) on boarding and deployment based on ETSI specification for Network Function Virtualization Management and Orchestration (NFV-MANO) [3]. The VNFs are deployed in an OpenStack [4] environment: the Radio Access Network (RAN) VNFs are container based using Kubernetes or VM based, while the Core VNFs are only VM based. Slicing mechanism and proper radio resource allocation are also supported in the RAN through the Software Defined Network (SDN) controller Open Air Interface (OAI)-FlexRAN implementation [5], but also in the Core through proper resources and Quality of Service (QoS) assurance in the proposed communication network deployment. The programmable infrastructure is orchestrated using ONAP [6] with specific features implemented for Service Design, Service Deployment and Service Operation. The paper emphasizes the technical capabilities available at this moment, including the specific configuration needed to deploy the RAN and Core VNFs using ONAP.

The paper is organized as follows. This section is dedicated to the introduction. Section 2 relates to a previous published paper and it highlights the evolutions presented by the current paper. Section 3 summarizes the current ONAP capabilities enabled over the OpenStack infrastructure providing also a practical guide for deploying end-to-end network slices. We conclude the paper in Section 4 where we

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0

also focus on our future work from 5G-EVE and 5G-VICTORI projects.

II. RELATED WORK

In a previously published paper [8] in 2019, by one of the co-authors of this paper, the 5G ecosystem intended to be developed within 5G-EVE was presented. The work from this paper is evolutionary and it is depicting practical guidelines for orchestrating VNFs and end-to-end slices using ONAP. The related work from standards, working groups, and other 5G-PPP European projects is exhaustively presented in [8] and will not be repeated here as, from our knowledge, no major change took place.

The research for end-to-end network slicing in 5G is intensive, as this is one of the key capabilities of the new technology with major outcomes for the vertical industries. The research is not limited only to the end-to-end slicing mechanism per se (which is also documented in the 3GPP standards), but extended to the whole ecosystem/framework which enables automated and business agile provisioning and maintenance of the slices. In the following paragraphs, some related works are briefly presented. These papers help in understanding the concept of slicing and the different technical challenges that need to be overpassed in order to achieve a fully functional framework.

The work from [9] presents a framework for automating the slicing management in the RAN part, with special attention dedicated to the functions, interfaces and information models. The theoretical framework is applied in a Proof of Concept with limited implementation of Radio Resource Management (RRM) algorithms.

In [10] the authors focus on the technical implementation of end-to-end slicing providing resource isolation and programmability on resources. The slicing is achieved by using OAI-RAN capabilities (similar with this paper) for the radio part and by introducing a deeply programmable node architecture (FLARE) for the core slicing with complete segregation among data and control plane.

The work from [11] presents the implementation of end-to-end slicing using Mosaic5G software packages [12] which are also utilized in this paper. The authors present different configuration scenarios for the Virtual Machines (VMs) in order to achieve the required performance for a certain VNF, which is also very useful for the work from this paper.

III. ORCHESTRATION OF PROGRAMMABLE INFRASTRUCTURE USING ONAP

In this paper, we are presenting a novel end-to-end network slicing mechanism implementation comprising both RAN and Core network elements using the Open Network Automation Platform (ONAP) [6] for NFV/VNFs onboarding and service deployment in 5G Non Standalone Architecture (5G NSA) [7]. The slicing mechanism is based on RAN Public Land Mobile Network (PLMN) identity (ID) techniques for the core network selection.

ONAP is an open source project developed by Linux Foundation, which allows design and creation of VNFs and network services orchestration. A set of blueprints is made available with every new platform release. ONAP Architecture provides two frameworks which enable the user to split the design and the creation phases (Design Time Framework) from the deployment phase (Run Time Framework). The ONAP implementation for 5G-EVE is depicted in Figure 1.

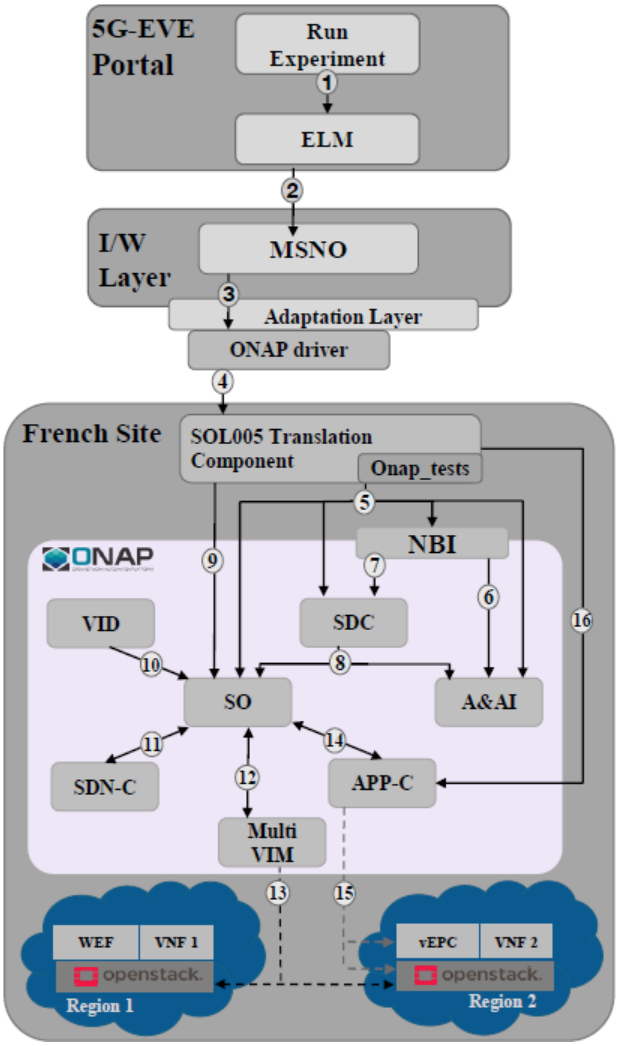


Figure 1. ONAP implementation in 5G-EVE

The main component of Design Time Framework is Software Design and Creation (SDC). Network services are represented as forwarding graphs composed of multiple VNFs. Each of them is represented in ONAP Catalog as Virtual Service Product. Deployment process begins with creation of the vendor's entitlements. In order to onboard the VNF, the user has to provide HEAT template, describing this network function, which has to meet ONAP specific requirements. The correctness of each template is checked

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0

automatically during uploading. Once the template is onboarded, the certification process could be started. The above-mentioned steps, Vendor Software Product (VSP) creation, template onboarding and certification, have to be done for each VNF. The network service could be composed of the network functions stored in the catalog and afterwards it is submitted for testing before being deployed to the production environment. The service model is stored in SDC which notifies the Service Orchestrator (SO) and Active and Available Inventory (A&AI). Once onboarding steps are finished, ONAP can deploy service instance on virtualized infrastructure based on models distributed to SO. Virtual Infrastructure Deployment (VID) enables the selection of the service and the triggering of the instantiation process in SO. In order to deploy the service, a la carte flow involves performing a set of steps and actions called building blocks. Service and VNFs objects need to be created first.

During the preloading phase, some VNF parameters values required during the instantiation are provided to SDN Controller (SDN-C). Once this is done, the VF modules that host VNFs could be instantiated. In fact this is triggered by the SO utilizing MultiVIM component, which is responsible for collecting information about tenants and clouds registered into ONAP. It also enables deploying VF modules in different type of infrastructure managers like OpenStack or Kubernetes. ONAP does not manage images of network functions. They have to be provided by the vendor and registered in specific cloud.

In 5G-EVE project, the operations from both Design and Run Time Framework are automated with open source *onap_tests* library [13]. The user has to provide HEAT templates for VNFs and network service composition, and requests to relevant ONAP components APIs are made in appropriate order. It is important to note that the images are previously uploaded to infrastructure managers.

The 5G-EVE Portal is the functional entity that provides access to the 5G-EVE Platform for verticals. Portal allows execution of functionalities such as instantiate, monitor KPIs and reconfigure. Each request from the Portal is passed through Experiment Lifecycle Manager (ELM) to the Interworking Layer which is composed of Multi Site Network Orchestrator (MSNO) and Adaptation Layer. MSNO decides which site should host the requested service and through the Adaptation Layer, it communicates with the specific site's orchestrator. SOL005 [14] Translation Component has been implemented on top of the ONAP instance dedicated to French site orchestration. It enables to communicate with ONAP Northbound API interface (NBI) (5) in order to perform operations ordered to execute by 5G-EVE Platform (4).

The VNF management capabilities of ONAP are assured by the joint interaction of the Service Orchestrator component with SDN-C(11) or Application Controller (APP-C)(14) components. The post instantiation configurations can have two origins. They are the result of the service reconfiguration request initiated by the end-user or the result of the control loop automation actions. The first one is triggered by SO APIs (9) or VID portal (10) where external applications or end-users can trigger orchestration workflows. Such workflows are either embedded into the orchestration logic, or they are network service and network function specific. In the latter case, the service designers can compose custom orchestration workflows with Workflow Designer capability of C portal. The standard or user-defined orchestration workflows can include day-2 reconfiguration operations or other VNF life cycle management operations into one orchestration workflow.

APP-C component is leveraged by SO for execution of selected life cycle management actions and reconfiguration actions on VNFs (14). The first includes such actions (15) like start, stop, or restart which are invoked on OpenStack directly. The reconfiguration actions can utilize NetConf configuration protocol or Ansible reconfiguration utility. For the latter one, APP-C is equipped with dedicated Ansible server which ensures communication with devices and it performs delegated reconfiguration tasks. APP-C enables also the design of configuration templates for each action with the Controller Design Tool dashboard, where the selection of communication protocol and translation of input parameters can be specified. It is worth mentioning that APP-C actions can be executed without SO and APP-C can be integrated directly with third party applications as a key functionality within 5G-EVE Portal (16).

The slicing implementation developed within the 5G-EVE project is achieved using the ONAP over the OpenStack cloud infrastructure [15]. The goal is to show the deployment of several telco VNFs - virtual Evolved Packet Core (vEPC) and virtual Radio Access Network (vRAN) for supporting different 5G use cases like Ultra Reliable Low Latency Communication (URLLC) and Massive Machine-Type Communications (mMTC), which will further be developed in the 5G- VICTORI project.

Figure 2 depicts the two end-to-end network slices (slice 1 represented in orange and slice 2 represented in green), using the described NFV/VNFs components.

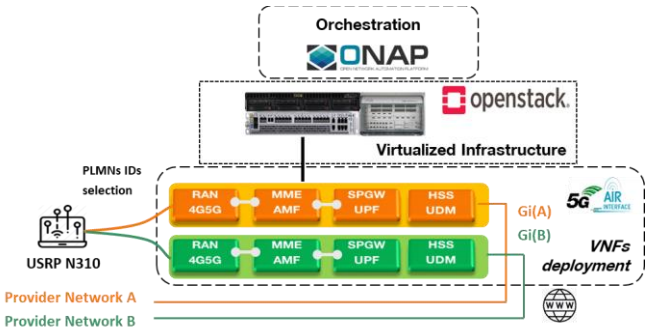


Figure 2. 5G testbed for programmable infrastructure orchestration using ONAP

The virtualized infrastructure was prepared in advance in terms of communication network, so throughout the ONAP deployment, each of the two slices are connected to its own OpenStack provider network. In this implementation, we used as Remote Radio Unit (RRU) the X310 [16] device, which is a high-performance, scalable Software Defined Radio (SDR) platform for designing and deploying next generation wireless communications systems, providing reliability and fault-tolerance for deployments, simplifying the control and the management of a network.

Based on the Mosaic5G group of software packages available [12], the vEPC core function network components – virtualized Mobility Management Entity (vMME), virtualized Serving/Packet Gateway (vS/PGW) and virtualized Home Subscriber Server (vHSS) – are deployed in a single Virtual Machine (VM), using HEAT templates. The first VM (Figure 3) deployed is a 64-bits operating system, using 4 cores Intel i7, 16GB RAM and 2 Ethernet interfaces for service connectivity and all the specific vEPC application configuration files related to: hostname, Domain Name Server (DNS), S6a diameter, S1-AP, authentication functions, Full Qualified Domain Names (FQDNs), communication networks and PLMNs.

```
oai-vepc-1-m5gcore-vm-1:~$ sudo oai-cn.status-all
Service       Startup  Current  Notes
oai-cn.hssd   enabled  active   -
oai-cn.mmed   enabled  active   -
oai-cn.spgwd  enabled  active   -
```

Figure 3. Mosaic 5G Core implementation

The second VNF (Figure 4) deployed through ONAP over the OpenStack infrastructure in one VM is the RAN eNodeB. The VM is using Ubuntu 18.04 LTS 64-bits as operating system and with a kernel version greater than 4.10.x, 4 cores Intel i7, 16 GB of RAM and two Ethernet interfaces.

```
oai-vepc-1-m5gcore-vm-1:~$ sudo oai-cn.status-all
Service       Startup  Current  Notes
oai-cn.hssd   enabled  active   -
oai-cn.mmed   enabled  active   -
oai-cn.spgwd  enabled  active   -
```

Figure 4. Mosaic 5G Core implementation

In order to be able to instantiate and run the VNF in virtualized environment, it is required to ensure several specific computing environment prerequisites configured in OpenStack infrastructure.

For the use case implementation presented, the specific infrastructure configuration was applied only on the physical host where the VM is intended to be deployed through ONAP. The physical host must have available the following special CPU flags enabled: **avx**, **avx2**.

Also, KVM mode needs to be enabled to be used in the OpenStack nova file path */etc/nova/nova.conf* for the *compute_driver* parameter *libvirt.LibvirtDriver*, setting the *libvirt* section configuration for virt_type to kvm and specifying the CPU model parameter cpu_mode value to host-model. An OpenStack specific flavor has been configured for this VNF as depicted in Figure 5.

```
openstack flavor create --ram 8092
                        --disk 50
                        --vcpus 8
                        --property trait:HW_CPU_X86_AVX=required
                        --property trait:HW_CPU_X86_AVX2=required
                        --property hw:cpu_cores=4
                        --property hw:cpu_sockets=1
                        --property hw:cpu_threads=1
                          oai_flavor
```

Figure 5. vRAN OpenStack special VM flavor

After the vRAN VM is running, the proper VM CPU allocation and configuration for *avx* settings were checked. In order to automatically instantiate the RAN VNF application, the script inside the VM is used (*start.mosaic5g*). This script instantiates the vRAN application after deployment, as this is not implicitly done.

Figure 6 depicts the success of this instantiation. The other specific application configurations related to hostname, S1-MME, communication networks and PLMNs, are being done through the HEAT templates.

```
[INFO] [X300] X300 initialization sequence...
[INFO] [X300] Maximum frame size: 1472 bytes.
[INFO] [X300] Radio 1x clock: 184.32 MHz
[INFO] [GPS] Found an internal GPSDO: LC_XO, Firmware Rev 0.929a
[INFO] [0/DmaFIFO_0] Initializing block control (NOC ID: 0xF1F0D00000000000)
[INFO] [0/DmaFIFO_0] BIST passed (Throughput: 1319 MB/s)
[INFO] [0/DmaFIFO_0] BIST passed (Throughput: 1316 MB/s)
[INFO] [0/Radio_0] Initializing block control (NOC ID: 0x12AD100000000001)
```

Figure 6. vRAN function running in VM

The correct functioning of the end-to-end slice is demonstrated by checking the statistics on the vMME in Figure 7.

```
=================================== STATISTICS ====

                |   Current Status| Added since last dis
Connected eNBs  |          1      |              1
Attached UEs    |          1      |              1
Connected UEs   |          1      |              1
Default Bearers |          1      |              1
S1-U Bearers    |          1      |              1
```

Figure 7 vMME network statistics

At this moment, our research is showing successful deployment of end-to-end slices with proper QoS enabled and automated network resource allocation leveraging ONAP capabilities over an OpenStack infrastructure.

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0

IV. CONCLUSIONS

In this paper, we have exposed a programmable infrastructure leveraging ONAP capabilities, with focus on presenting the onboarding and deployment of the VNFs, showing the 5G slicing capabilities in the RAN and Core subsystems in 5G Non Standalone Architecture. The network slicing end-to-end mechanism is based on the RAN PLMN ID techniques for the selection of the Core Network. The novelty for network slicing implementation resides in using of ONAP over OpenStack cloud infrastructure. The aim is to demonstrate the deployment of a small number of telco VNFs (vEPC, vRAN) for providing services to different 5G use cases like URLLC and mMTC.

In the 5G-EVE project, a dedicated portal was developed having the functionality of providing access for verticals at 5G-EVE Platform. Through this entity, there is a possibility to instantiate, monitor KPIs or reconfigure the service. A set of information is mandatory to be fulfilled as a pre-requisite: the vendor of VNFs must provide the HEAT templates and the service graph, which are stored in a specific cloud. The requests to related ONAP components APIs are made accordingly with the network service composition.

Based on the Mosaic5G group of software packages, the vEPC, containing vMME, vPGW and vHSS, is deployed in one VM. A similar approach is used also for deploying of VNF RAN eNB. This model is capable of providing multiple e2e flexible and efficient network slices (the time of deployment is less than 1 minute), automatic network resource allocation and VNF life-cycle management (in the future). The research will be enhanced within 5G-EVE with in life management capabilities resulting in a fully functionable test bed to be further utilized for the 5G-VICTORI usecases, as described in Section 5.

V. FUTURE WORK

The work from this paper stands as basics for the future evolutions envisioned in the 5G-EVE and the 5G-VICTORI European projects. Therefore, it is important for the reader to understand the future plans as the research and development will continue for the next years.

In-life slice configuration will be performed by increasing the network compute capacity triggered by service reconfiguration request initiated by the user through the portal or as a result of control loop automation actions. In the run-time step, a set of in-life management actions are involved: upgrade, network slice instance (NSI) scaling, changes of NSI capacity, changes of NSI topology.

The entire functionality will be further evaluated on 5G-VICTORI France/Romania cross site orchestration cluster using 5G-EVE test bed facilities in two use cases: transportation and energy. The test bed physical infrastructure will contain control and compute servers, storage, eNodeB PNF, 4G/5G video cameras, radio licensed spectrum, IP/MPLS between 5G-VICTORI Bucharest facility and Alba Iulia Municipality infrastructure; L3VPNs and IPsec connectivity between 5G-VICTORI infrastructure and 5G-EVE French Cluster ONAP, MEC for video data analytics. The virtualized infrastructure will be deployed using OSMv5; Docker; ONAP and SDNC suite and Prometheus will be used for performance monitoring. NFV/VNF suite will be provided through 4G5G RAN OAI Mosaic5G for radio part, 4G5G Core OAI Mosaic5G for core part and the application software for use case application experimentation. Grafana is used for Service data visualization and KPI performance validation.

The first use case is addressing the transportation service that addresses URLLC feature requirements and pop-up network on-demand creation capabilities. Infotainment versus public safety services will be orchestrated when a threat is identified by the system, the infotainment resources are back-logged and a public safety high quality live stream is established. The use case will cover all deployment phases: infotainment and public safety application design; 5G deployment and instantiation, immediate setup time (triggered) service, service in life management and automation, service instantiation & MEC analytics function, QoS experimentation and service optimization. The creation of the slice will be triggered either manually or automatically by an external event alert, the functionality of the digital mobility use case will be assessed trough a set of tests spanning from vertical applications performance evaluation to solution functionality.

The second use case is a Low Voltage (LV) smart energy metering use case addressing mMTC capabilities to show that 5G mMTC services can be used for an advanced energy metering deployment. It will be demonstrated using the same 5G-EVE test bed facilities to provide energy metering services for energy consumers like public buildings and street lighting in the Alba Iulia Smart City environment and for energy sources like photovoltaic panel or national grid. Smart Energy metering use case demands advanced requests of high data processing capacities, flexible provisioning capabilities and service customization to create automated capabilities for deployment and in-life management over network virtualized infrastructures. The scenario assumes metering data collection from endpoints scattered across a city and requires scalability of network slice from get-go. The collected measurements will be transferred to the central cloud facilities where they will be stored, processed and analyzed by the telemetry platform. Advanced analytics will be used to predict future power demands.

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0

ACKNOWLEDGMENT

This work was partly funded by the European Commission under the European Union's Horizon 2020 program - grant agreement number 815074 (5G-EVE project) and 857201 (5G-VICTORI). The paper solely reflects the views of the authors. The Commission is not responsible for the contents of this paper or any use made thereof.

REFERENCES

[1] 5G-PPP Phase 3 Project 5G-EVE. [Online]. Available from: https://www.5g-eve.eu/ 2020.06.12

[2] 5G-PPP Phase 3 Project 5G-VICTORI. [Online]. Available from: https://www.5g-victori-project.eu/ 2020.06.12

[3] Network Functions Virtualisation (NFV); Management and Orchestration, Group Specification, Dec. 2014.

[4] Openstack framework. [Online]. Available from https://www.openstack.org/ 2020.06.12

[5] OAI-FlexRAN. [Online]. Available from: https://gitlab.eurecom.fr/mosaic5g/mosaic5g/wikis/tutorials/oai-ran?fbclid=IwAR3V__MC60MMPV6KqFIJpOox0EnUphKgeH0hApCGiIsSDs3NwVNe70oBAZ8 2020.06.12

[6] ONAP Architecture Overview. [Online]. Available from: https://www.onap.org /architecture 2020.06.12

[7] 5G NSA: Universal Mobile Telecommunications System (UMTS); LTE; 5G; NR; Multi-connectivity; Overall description; Stage-2 (3GPP TS 37.340 version 15.5.0 Release 15.

[8] L. Yala, M. Iordache, A. Bousselmi, S. Imadali, "5G mobile network orchestration and management using open-Source", 2019 IEEE 2nd 5G World Forum (5GWF), doi: 10.1109/5GWF.2019.8911690

[9] R. Ferrús, O. Sallent, J. Pérez-Romero and R. Agustí, "On the automation of RAN slicing provisioning: solution framework and applicability examples", Eurasip journal on wireless communication and networking, Dec. 2019, pp. 1-12, doi: 10.1186/s13638-019-1486-1.

[10] A. Nakao et al.," End-to-end network slicing for 5G mobile networks", Journal of Information Processing 25, Feb. 2017, pp. 153-163, doi: 10.2197/ipsjjip.25.153

[11] I. Afolabi et al., "End-to-end network slicing enabled through network function virtualization", 2017 IEEE Conference on Standards for Communications and Networking (CSCN), Sep. 2017, doi: 10.1109/CSCN.2017.8088594

[12] Mosaic5G, https://gitlab.eurecom.fr/mosaic5g/mosaic5g/-/wikis/tutorials

[13] Orange Opensource onap-tests library. [Online]. Available from: https://gitlab.com/ Orange-OpenSource/lfn/onap/onap-tests/-/tree/dublin/onaptests 2020.06.12

[14] ETSI GS NFV-SOL 005. [Online]. Available from: https://www.etsi.org/deliver /etsigs/NFV-SOL/001099/005/02.04.0160/gsNFV-SOL005v020401p.pdf 2020.06.12

[15] Openstack Infra: OpenStack installation tutorial for Ubuntu. [Online]. Available from: https://docs.openstack.org/newton/install-guide-ubuntu/overview.html 2020.06.12

[16] x310 RRU USRP. [Online]. Available from: https://www.ettus.com/all-products/x310-kit/ 2020.06.12

Copyright (c) IARIA, 2020. ISBN: 978-1-61208-802-0