SDN-Sim: Integrating System Level Simulator with Software Defined Network

S. Ghosh, S. A. Busari, T. Dagiuklas, M. Iqbal, R. Mumtaz, J. Gonzalez, S. Stavrou, and L. Kanaris

***Abstract*—With the introduction of diverse technology paradigms in next-generation cellular and vehicular networks, design and structural complexity are skyrocketing. The beyond-5G use cases such as mobile broadband, 5G-V2X and UAV communications require support for ultra-low latency and high throughput and reliability with limited operational complexity and cost. These use cases are being explored in 3GPP Release 16 and 17. To facilitate end-to-end performance evaluation for these applications, we propose SDN-Sim - an integration of a System Level Simulator (SLS) with a Software Defined Network (SDN) infrastructure. While the SLS models the communication channel and evaluates system performance on the physical and data link layers, the SDN performs network and application tasks such as routing, load balancing, etc. The proposed architecture replicates the SLS-defined topology into an SDN emulator for offloading control operations. It uses link and node information calculated by the SLS to compute routes in SDN and feeds the results back to the SLS. Along with the architecture, data modeling and processing, replication and route calculation frameworks are proposed.**

***Index Terms*—System Level Simulator, SDN, GNS3, Mininet-wifi, OpenDaylight**

I. INTRODUCTION

Towards 5G/B5G, the third generation partnership project (3GPP) is finalizing Release 16 and defining Release 17[1]. In the area of vehicular networks, the 3GPP, in partnership with the Fifth Generation Automotive Association (5GAA), is driving the efforts on 5G-based vehicle-to-everything (V2X) paradigm which adds advanced features to the LTE-V2X from Release 14, particularly in the areas of support for ultra-reliable and low-latency communication (URLLC) applications for the future intelligent transport systems (ITS) [1], [2], [3]. In the evolution path from LTE-V2X to 5G-V2X, the authors in [1] advocated the incorporation of software-defined networking (SDN) in the architecture to enhance the system perfromance through SDN's capabilities in facilitating intelligent multihop routing, dynamic resource allocation and advanced mobility support, among others.

To evaluate the performance of proposed algorithms, techniques and frameworks for any new era of communication networks, numerical simulations, mathematical analyses and field trials are the three main approaches being employed. Though analytically tractable, mathematical methods (e.g., stochastic geometry tools) are often constrained with simplifying assumptions that potentially limit their use in modeling large-scale, highly-complex and dynamic networks. Realistic performance can be measured in live operating environments. However, the economic and operational requirement are costly and practically-infeasible for the early design and development stages. Hence, since the past few decades, simulations have become remarkably important tools for the assessment of networks performance due to the obvious cost and implementation advantages [4].

Depending on the performance metrics under investigation, simulators can be categorized into three: Link Level Simulator (LLS), System Level Simulator (SLS) and Network Level Simulator (NLS). The LLS examines detailed, bit-level physical (PHY) layer functionalities of a single link. The SLS evaluates the performance of links involving many base stations (BSs) and user equipment (UEs) at medium access control (MAC) layer (with the PHY abstracted). It focuses on the radio access network/air interface and facilitates analyses of resource allocation, capacity, coverage, spectral and energy efficiencies, among others. The NLS, however, assesses the performance of protocols across all layers of the network, including control signaling and backhaul/fronthaul issues. Performance is characterized using metrics such as latency, packet loss, etc [5].

Besides metric-based classification, simulators can also be grouped based on radio access technologies supported (cellular, vehicular, WiFi etc), coding language (MATLAB, Python, C++, etc), licensing option (open source, proprietary, free of charge for academic use) or network scenario capabilities (LTE, 5G, B5G etc) [5][2]. While the SLS does not simulate beyond the MAC layer, the NLS simulate networks up to the application layer. However, the implementation and computational complexity of NLS becomes very high when a large number of nodes are involved [4].

Another major paradigm shift in network design took place with the advent of SDN [6]. It decouples the control (signaling) plane from data (forwarding) plane and runs applications in the Application Plane to manage the network. This brings transparency in network design and lets software developers write applications for managing the networks, keeping the internal design in abstraction. Each layer uses several interfaces to communicate with each other. Tthe control plane communicates with both Application and Data planes using

---

S. Ghosh, T. Dagiuklas and M. Iqbal are with the London South Bank University; S. A. Busari is with the Federal University of Technology Akure, R. Mumtaz and J. Gonzalez are with GS-Lda; S. Stavrou is with the Open University of Cyprus and L. Kanaris is with Sigint Solutions

[1] https://www.3gpp.org/news-events/2058-ran-rel-16-progress-and-rel-17-potential-work-areas

[2] Representative simulators include the Vienna LTE-A and 5G simulators for LLS and SLS (https://www.nt.tuwien.ac.at/research/mobile-communications/vccs/), and the 5G-K Simulators for the NLS (http://5gopenplatform.org/main/index.php).

North and Southbound interfaces respectively. In case of a cluster of controllers, East and Westbound interfaces are used for communicating among them.

OpenFlow being the default southbound protocol for SDN, uses Flow Tables (FT) to perform packet forwarding. Each FT entry is a forwarding rule determined by the controller. A forwarding rule has mainly three major fields, a “match”, an “action” and a “priority”. A “match” is some criteria for an inbound packet to be checked. If a packet satisfies the criteria, it is termed as a “table hit”, otherwise it is a “table miss”. For each case, an action is defined such that, the OpenFlow switch executes on the subjected packet. If a packet satisfies matches from multiple flow rules, priority is used to break the tie. Flow entries are populated by the SDN Controllers. For every table miss, OpenFlow switch requests the controller and the controller replies with a flow entry. If the controller can not resolve an action, it is set as a “drop” and the switch does not process the packet. The decoupled control plane reduces computational cost on forwarding devices by offloading the control packet processing tasks to the controller. Therefore, SDN offers better modularity, programmability, agility, automation and load balancing capability than of traditional networks. Also, the SDN-based approach is used in network design practices for cloud computing and 5G.

In this work, we present a novel SDN based System Level Simulator simulator (SDN-Sim) platform where the SLS-Stage runs in MATLAB and the NLS-stage is based on python3. By inheriting all the benefits of SDN, the architecture is designed to reduce the overall computational complexity of the system considerably. The computationally-demanding upper layer network functions (e.g. inter-cellular routing) are offloaded to the virtualized cloud infrastructure. Low-level network infromation (e.g. Channel model, topology etc.) are maped from SLS to SDN. Python is used for application development. Virtual infrastructure is setup using VMWare ESXi servers. OpenFlow and RESTConf are used as south and northbound protocols respectively. Opendaylight is used as the SDN controller while GNS3 and Mininet-wifi are used for data plane emulation.

This paper proposes an architecture to integrate MATLAB-based SLS with SDN using OpenDaylight, Mininet-Wifi and GNS3. The implementation is described with data modeling of the middleware and sequence modeling of messages exchanged among various entities. Three application (*topobuild*, *toposense* and *toporoute*) are also presented for data fetching, interfacing and processing. The remainder of this paper is organized as follows. Section II describes the system architecture and implementation. The experimental setup and results are presented in Section III followed by conclusions and future research directions in Section IV.

II. SYSTEM ARCHITECTURE AND IMPLEMENTATION

Figure 1 depicts the system architecture of SDN-Sim, with the SLS at the bottom and the SDN infrastructure running on the top. When the SLS does the channel modeling and scheduling, the SDN takes care of the upper layer functionalists such as IP routing and traffic control, described as follows.

A. System Level Simulator

The tasks of the SLS (described hereunder) are executed in loops of transmission time intervals (TTIs) and the results are averaged over several simulations runs or channel realizations [7].

- **Scenario Setting**: The layout depicts network of BSs and UEs, configured with 3D locations of the nodes. UEs are attached to their serving BSs with parameters, line-of-sight probability, distance, and signal to noise ratio (SNR).
- **Channel Modeling**: For all links, the path loss (PL), shadow fading (SF), transmit and receive and antenna gains, and fast fading are calculated to estimate the channel of each user, for both desired and interfering links.
- **Scheduling**: Radio resources are allocated to users based on the scheduling algorithm. Resource blocks (bandwidth) and power are allocated either in quasi-random fashion (for open loop configuration) or based on feedback from the users in closed loop systems. The channel state information (CSI) feedback, together with other factors such as the traffic type of users, link adaptation strategy employed and quality of service (QoS) demands, are used as decision determinants at the scheduling stage.
- **Link Quality and Performance Estimation**: The signal to noise and interference ratio (SINR) of the links are then estimated. Using the SINRs and the link abstraction model (for the block error rate (BLER)), the throughput of the users and the capacities of the cells are calculated.

Recent developments in SLSs are mainly focused on solving problems on channel modeling [8], high frequency communication [9], [10], coexistence and performance optimization [11], energy efficiency, latency, scheduling and load balancing over a heterogeneous network [12], among others. The 5G public-private partnership project (5GPPP) have described several aspects in softwarization, service management and orchestration in their architectural reference [13] which includs, SDN, cloud computing, virtualization, etc. Therefore, cloud computing is being employed to enhance the scalability and computational efficiency of 5G SLSs by offloading the computational load. [14].

B. Software Defined Network

The tasks of the SDN is described as follows.

- **Channel Information Update**: The SLS running in a MATLAB server with Open Database Connector (ODBC) driver updates the topology and channel modeling parameters to a centralized Database.
- **SDN Emulation**: The topology information is reactively fetched and emulates using *TopoBuild* in an SDN Data-Plane through GNS3 and Mininet-wifi. The control plane starts communicating with Open-Flow
- **Route Calculation**: The application plane fetches topology from the control plane, translate it into a graph(using *TopoSense*) and finds the shortest paths between node pairs (using *TopoRoute*). These paths are fed back to the database and the controllers.

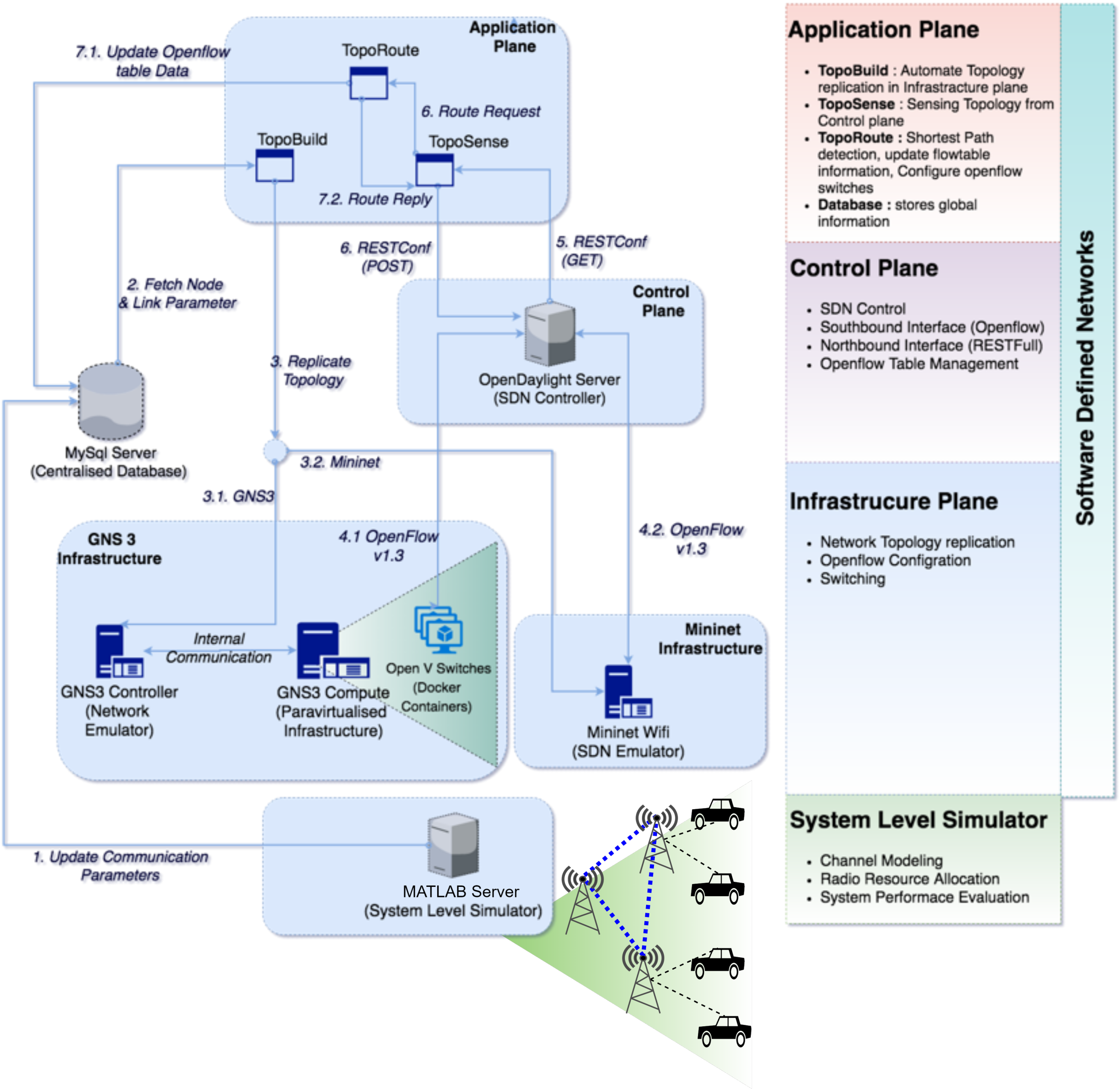

Fig. 1. Schematic system architecture of SDN-Sim with Full stack setup along with their core functions

The work-flow and communication sequence taking place between the various elements of the architecture are as given in Figure 1.

1) *Update Communication Parameters*: The SLS performs channel modeling, radio resource allocation, and system performance evaluation. After optimizing the model for a given scenario, it generates several channel parameters (bandwidth, distance, path-loss, latency, and delay) and BS parameters (position, range, RSSI, transmission power etc.). Since parameters are calculated per BS, each BS updates their local dataset to the centralized database.
2) *Fetch Node and Link Parameters*: *Topobuild* is a bespoke program to fetch SLS parameters from the central database and relays to the SDN Data plane for replicating the topology. BSs are placed as wireless access points, running OpenFlow protocol. Links, depending on types (i.e. fronthaul or backhaul) are assigned to BSs. A fronthaul link is wireless, therefore carrying several radio parameters such as path loss, frequency band, RSSI, etc.
3) *Topology replication*: *Topobuild* translates parameters obtained from the central DB into a series of commands. In the proposed architecture, there are two possible infrastructures (GNS3 and Mininet-wifi) with three possible deployment options (full GNS3, full Mininet-wifi and hybrid). Based on the deployment type and connection specifications, Topobuild.py generates a script to replicate the topology and injects it to the specific engines (GNS3 and/or Mininet-wifi). A feature comparison between GNS3 and Mininet-WiFi is given in Table I and briefly described as follows.

TABLE I

A COMPARATIVE STUDY OF GNS3 AND MININET WIFI AS DATA PLANE ENGINE

	GNS3	Mininet-Wifi
Similarities	Both are open source, offer GUI based network design, support API based Interfacing with the external environment, have accessible physical interface, host OVSs and dummy workstations for data plane	
Advantages	Not Limited to only SDNs Use of actual OVSs Test-result accuracy Realistic testcases Scalable via Clustering Native docker Support Extensibility with apps from GNS3 market- place	Rapid Implementation, no SDN/OpenFlow configuration needed link configuration wireless support native SDN support Inbuild mobility & propagation models
Demerits	No wireless support Manual configuration for SDN/OpenFlow and other supporting devices No link configuration Requires more physical resource to setup test environment	Limited to SDNs Test-result accuracy Scalability via multi-threading No clustering supports No paravirtualization No app based functional extensibility

- *GNS3 Infrastructure:* GNS3 is an open-source network emulation software. It comprises of a controller unit that interfaces through RESTful APIs, manages topology etc. and a cluster of para-virtualized compute nodes that host VNFs such as routers, switches, firewalls as containers or VMs. In SDN-Sim, *Topobuild* communicates with GNS3-Controller using RESTful API and the Open-V-Switches (OVS) run as docker container within GNS3-Compute cluster. Therefore, GNS3 offers realistic test cases with testbed and produces more authentic results than of a simulation. On a downside, it lacks support for wireless networking emulation, however it allows access to a physical wireless card.

- *Mininet-WiFi Infrastructure:* With a Wi-Fi extension, inheriting all the features of Mininet, it can now emulate wireless SDNs. With support of its python API, programming and configuring Mininet-Wifi is more user-friendly. Mininet-Wifi runs in a single sandbox with its own interactive command line. Thus, *Topobuild* uses raw sockets to inject mininet-wifi commands for deployment and parameters updates. It also offers four mobility and five propagation models. Since Mininet-Wifi uses HWSIM drivers to simulate wireless networking, emulating a very large network is constrained. Also, it does not support clustering,.

4) *Accessing Control Plane with OpenFlow:* Both GNS3 and Mininet-wifi host OVSs, thus, they communicate with the controller using OpenFlow v1.3 (OF1.3) as southbound protocol. OF1.3 uses bidirectional messaging to communicate with the switches. A switch requests a controller with a Packet-In message and the controller replies back with a Packet-Out. SDN-Sim uses *OpenDaylight - Beryllium SR4* (ODL) as controller, that runs as VM. Both GNS3 (4.1) and Mininet-wifi (4.2) use TCP port 6633 and/or 6635 for communication. Control plane supports binding of several controller nodes with clustering to maintain scalability, high availability and persistence in data plane. ODL supports 'Akka' clustering for this purpose.
5) *Interfacing with Application Plane with RESTConf:* SDN-Controller interacts with the application plane using Northbound APIs. ODL uses RESTConf (RFC-8040). It is a RESTful version of NETCONF (RFC-6241) protocol, uses JSON (RFC-7159) format to transfer data among REST enabled devices. In SDN-Sim, we use two RESTConf resources (Inventory and Topology). “Operational” resources are to read and “Config” are to write. The inventory resource provides node-wise OpenFlow tables, and the topology of the network is provided by the Topology resource. The App *TopoSense* makes use of the resources to model a graph by fusing topology and flow-table information. *TopoSense* invokes *TopoRoute* to calculate route for a given topology.
6) *Route Calculation: TopoRoute* uses *Stochastic Temporal Edge Normalisation (STEN)*[15] technique to find routes. It receives the topology and flow tables from *Toposense*. The link parameters are already present in the database. By fusing them, a single source shortest path algorithm, is run over every pair of vertices. a set of all possible routes are generated.In a traditional network, local routes are shared among neighbours to form routing tables.With the size of the network, due to propagation delay, the routing becomes significantly slow. However, in an SDN paradigm, discovery is done by the controller hence the topology graph is mapped proactively. Therefore, the shortest paths between every pair can be calculated in parallel. This speeds up the routing process and allows scalability in the network.
7) *OpenFlow Tables Update: TopoRoute* does event-driven update of the flow-tables in the Central DB (7.1). The calculated routes are fed back to the *TopoSense*, which eventually replies the routes back to the controller using RESTConf Inventory-Config API.

Once the central DB is updated, SLS picks the information and an inter-cellular route is discovered to leverage the lower (physical and data link) layer operations. The complete operational sequence diagram is given in Figure 2.

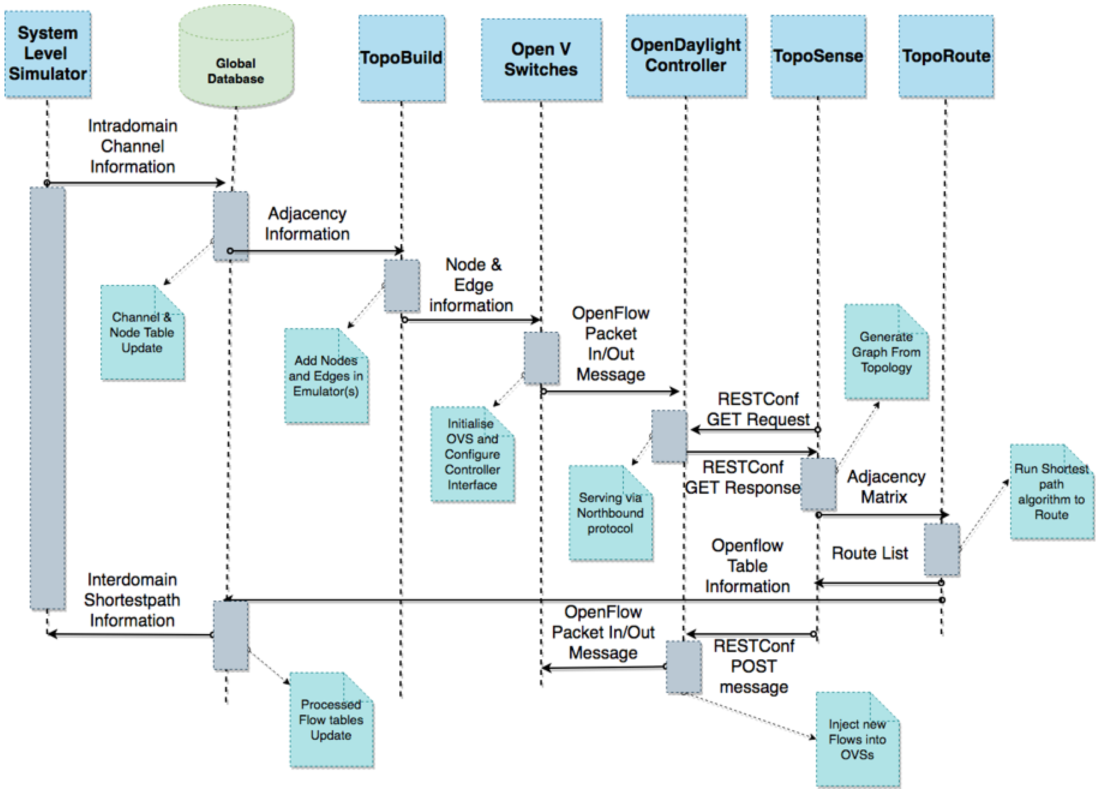

Fig. 2. Sequence diagram for various message exchange between components of SDN-Sim

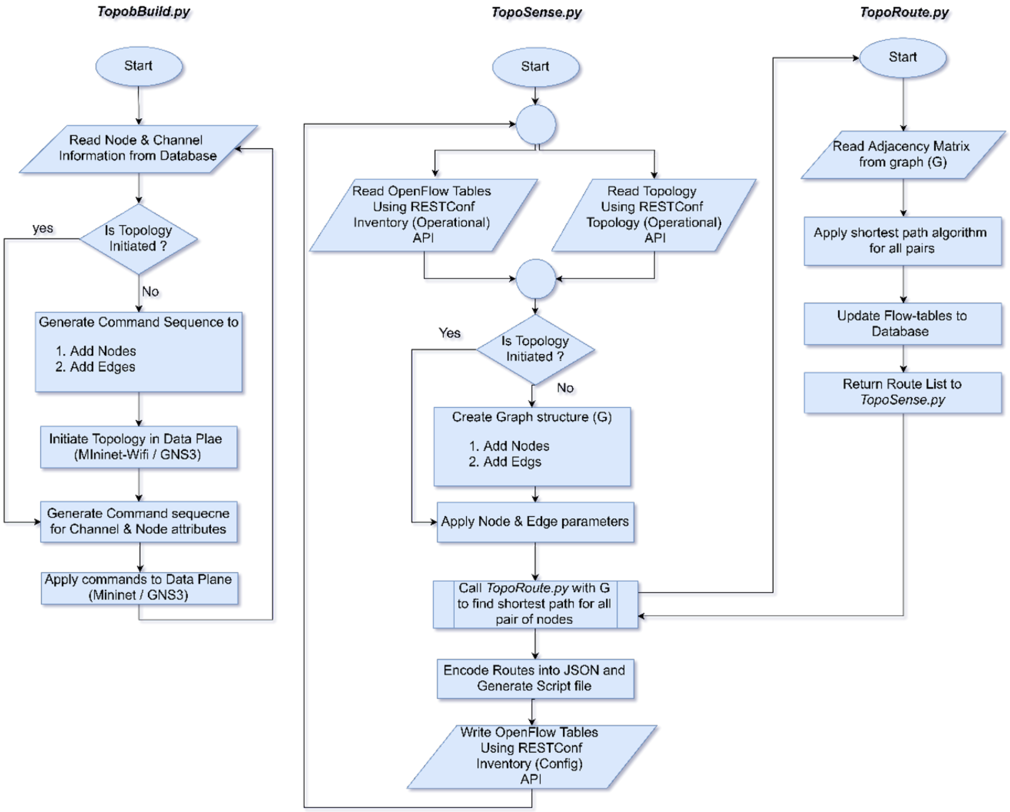

Fig. 3. Detailed procedure of three Application Plane programs (TopoBuild.py, TopoSense.py & TopoRoute.py)

TABLE II
DESCRIPTIONS OF ATTRIBUTES OF THE DATA MODEL

Node Attributes	
Node ID	Primary key, Unique ID for each node
Type	Type of the node (AP or Host)
Range	Communication Range of the node
Position	Location of node (3D cartesian coordinate)
Channel	Operating Channel Number
Frequency	Operating Frequency (In Hz)
Mode	Operating Mode (B/G/N/AC)
Tx_Power	Transmit Power in mW
IP_Address	Nodes IP Address
MAC_Addess	Nodes MAC Address
Access Point Attributes	
Station Association	List of Host the AP is associated
Host Attributes	
AP_Association	ID of the AP to which the host is associated
RSSI	Relative Received Signal Strength (at host end)
Flow Table Attribute	
DPID	Unique Data path ID of the AP
Source_IP	Match field for source IP in ingress packet
Destination_IP	Match field for Destination IP in ingress packet
Source_MAC	Match field for source MAC in ingress Frame
Destination_MAC	Match field for source MAC in ingress Frame
Action	OpenFlow Action opcode
Timeout	Timeout (Dead) timer
Packet_Count	Total packet count statistics
Byte_Count	Total bytes count statistics
Duration	Hold time for OpenFlow Entry
Node_Channel_Map Attributes	
Channel_ID	Foreign key to Unique Channel ID in Table
Node_1	Foreign key to first Node ID
Node_2	Foreign key to second Node ID
Channel Attributes	
Channel_ID	Primary key, Channel Identifier
Bandwidth	Channel Bandwidth in MbPS
Distance	Distance between incident nodes in meters
Pathloss	Pathloss of the channel
Latency	Average latency (in ms)
Delay	Average RTT (in ms)

C. *Data Modelling & Design of Central Database*

The central database, playes the role of a middle-ware between the SLS and the SDN. Table II shows the related entities and their corresponding attributes for the data model. There are three main entities: Node, Flow Table and Channel. A node can be of either Access Point (AP) or Host. One host is associated with one AP and one AP can associate with many hosts. Each AP is an OVS thus it contains flow table(s), and an unique *Data Path Identifier* (DPID). A pair of nodes makes a channel, with a unique channel ID (mapping is recorded at *Node_Channel_Map* table). A channel between a Host and an AP is called fronthaul (Wireless), and between a pair of APs is a backhaul (Wired). No channel exists between two hosts. Since *TopoSense* reads information from the APs hence, the route calculation by *TopoRoute* takes place over the backhaul network.

D. *Design of Application Plane*

Three major apps in the Application Plane run three python apps: TopoBuild.py, TopoSense.py and TopoRoute.py. In the previous sections, their usage has been mentioned. Figure 3 depicts their working principles using flowcharts.

III. RESULTS

Figure 4 depicts the time both complexity of various phases and the latency for three use-cases (1, 7 and 19 sites and 3 sectors/site with each sector consists of 1, 25, 50, 75, 100 users) depicted on the Figure 4 (A), (B) and (C), respectively. The Backhaul topology is configured wired and evaluated for Linear (minimal) and Mesh (Maximal) connectivity. The time to build the network on SDN platform is termed as *Build Time* and *Response Time* the SDN takes to initiate traffic flow. Figure 4(D) compares the breakdown of time consumption, number of links and routing time, a multi-threaded implementation of Shortest path algorithm limits the reactive route-selection into sub-second interval.

A. *Experimental Setup*

During experiment, the following compute-node setup was followed. The SLS runs in a MATLAB server VM with the Database tool running and ODBC adapter connects to MySQL Database. Mininet-wifi VM hosts BSs as APs, and OVSs and UEs as stations. ODL runs in an Ubuntu 64 bit 14.04 VM and Application server VM hosts MySQL Database server along with TopoSense.py, TopoBuild.py and TopoRoute.py apps. VMWare ESXi 6.5 server is used for the virtualization.

B. *Latency and Time Complexity*

Figure 4 depicts the latency of several stages of integration, SLS shares the most part of it. The setup phase consumes a significant amount of time depending on the network size and topology, this includes channel allocation and scheduling, SDN setup, Flow table population, Proactive route calculation etc. However, the run-time is reactive in nature and response in millisecond scale (Figure 4(E)), since all the routes are pre-calculated and network runs on a centralised virtual platform, this eliminates control packet exchange between devices to learn network topology. The time complexity of proactive phase is of a high degree polynomial class and the reactive phase is constant, thus once the SDN is deployed, response time comes down to millisecond scale.

IV. CONCLUSION AND FUTURE WORK

In this paper, we proposed an architecture and methodology to integrate a system level simulator with a software-defined networking infrastructure using a relational database as middleware. A comparative analysis on using GNS3 and Mininet-wifi based data-plane is given. Sequence model shows the data and control flow among several percolating entities. The data model of the middleware describes the information structuring and a set of algorithm to fetch and replicate will extend this architecture by appending an analytics plane on top of it. Data analytics empowered by deep learning algorithms will learn the run-time behavior of the network and help to improve the network automation, self-organization and state prediction ability. The proposed simulator facilitates the end-to-end performance evaluation of 5G and beyond 5G use cases such as 5G-V2X where latency is a critical performance metric.

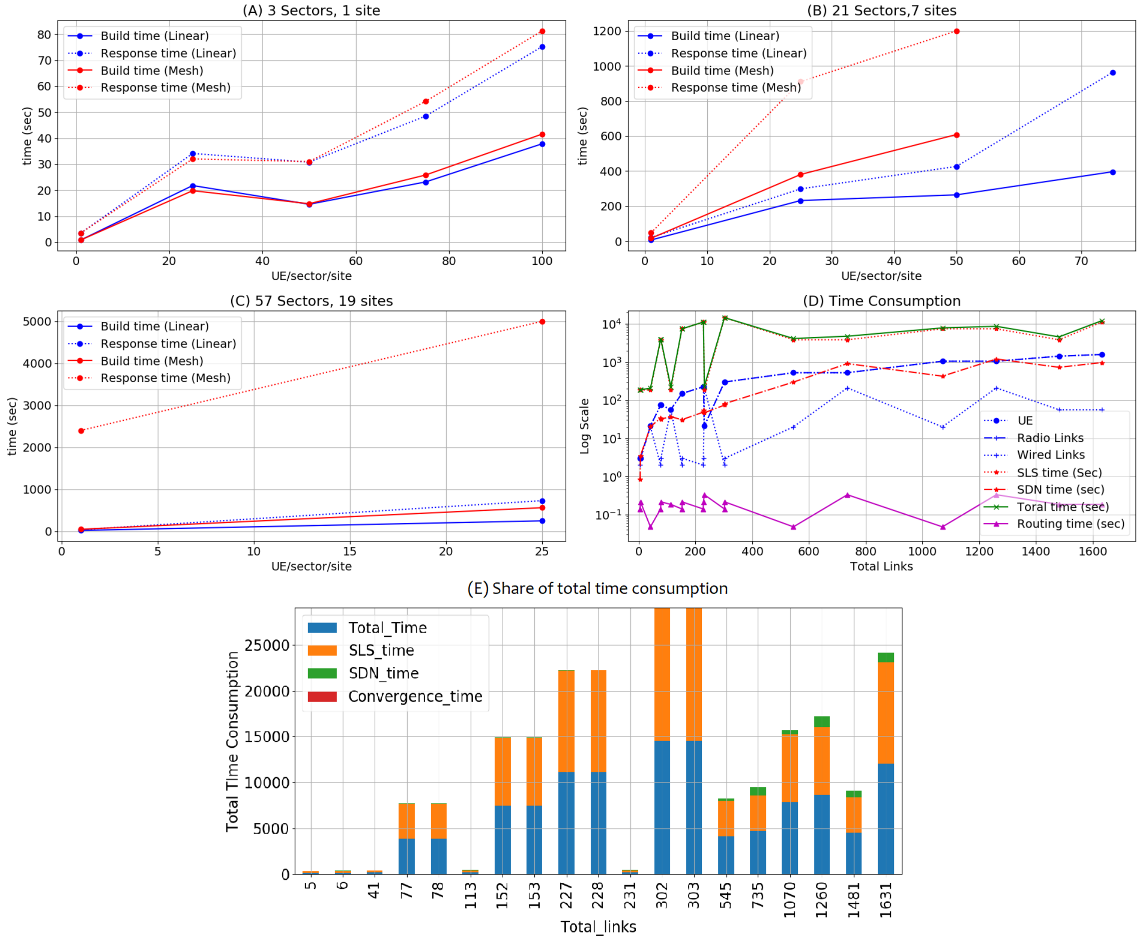

Fig. 4. Sub-plots (A), (B), (C) depicts the build and response time for minimal (linear) and complete (mesh) topology of 1, 7 and 19 sites, respectively with 3 sectors per sites. (D) depicts total time consumption is predominated by the SLS channel scheduling, SDN tasks are comparatively lightweight and Routing time bounded by sub-second interval. (E) shows the total time consumption has a constant convergence time (it is too small to be visible on the staked bar chart)

V. ACKNOWLEDGEMENT

This project is funded by MSCA-RISE - Marie Sklodowska-Curie Research and Innovation Staff Exchange (RISE) scheme under the project SONNET (Self-OrganizatioN towards reduced cost and eNergy per bit for future Emerging radio Technologies) with grant agreement no. H2020-MSCA-RISE-2016-734545.

REFERENCES

[1] L. Zhao, X. Li, B. Gu, Z. Zhou, S. Mumtaz, V. Frascolla, H. Gacanin, M. I. Ashraf, J. Rodriguez, M. Yang, and S. Al-Rubaye, "Vehicular Communications: Standardization and Open Issues," *IEEE Communications Standards Magazine*, vol. 2, no. 4, pp. 74–80, December 2018.

[2] S. A. Busari, M. A. Khan, K. M. S. Huq, S. Mumtaz, and J. Rodriguez, "Millimetre-wave massive MIMO for cellular vehicle-to-infrastructure communication," *IET Intelligent Transport Systems*, vol. 13, no. 6, pp. 983–990, June 2019.

[3] S. A. Busari, K. M. S. Huq, S. Mumtaz, J. Rodriguez, Y. Fang, D. C. Sicker, S. Al-Rubaye, and A. Tsourdos, "Generalized Hybrid Beamforming for Vehicular Connectivity Using THz Massive MIMO," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 9, pp. 8372–8383, Sep. 2019.

[4] M. K. Müller, F. Ademaj, T. Dittrich, A. Fastenbauer, B. Ramos Elbal, A. Nabavi, L. Nagel, S. Schwarz, and M. Rupp, "Flexible multi-node simulation of cellular mobile communications: the Vienna 5G System Level Simulator," *EURASIP Journal on Wireless Communications and Networking*, vol. 2018, no. 1, p. 227, Sep 2018.

[5] P. Alvarez, C. Galiotto, J. van de Belt, D. Finn, H. Ahmadi, and L. DaSilva, "Simulating dense small cell networks," in *2016 IEEE Wireless Communications and Networking Conference*, April 2016, pp. 1–6.

[6] D. Kreutz, F. M. V. Ramos, P. E. Verssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, Jan 2015.

[7] M. Rupp, S. Schwarz, and M. Taranetz, *The Vienna LTE-Advanced Simulators: Up and Downlink, Link and System Level Simulation*, 1st ed., ser. Signals and Communication Technology. Springer Singapore, 2016.

[8] S. A. Busari, K. M. S. Huq, S. Mumtaz, and J. Rodriguez, "Impact of 3D Channel Modeling for Ultra-High Speed Beyond-5G Networks," in *2018 IEEE Globecom Workshops (GC Wkshps)*, Dec 2018, pp. 1–6.

[9] M. Xiao, S. Mumtaz, Y. Huang, L. Dai, Y. Li, M. Matthaiou, G. K. Karagiannidis, E. Bjrnson, K. Yang, C. I, and A. Ghosh, "Millimeter wave communications for future mobile networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 9, pp. 1909–1935, Sep. 2017.

[10] M. Mezzavilla, M. Zhang, M. Polese, R. Ford, S. Dutta, S. Rangan, and M. Zorzi, "End-to-End Simulation of 5G mmWave Networks," *IEEE Communications Surveys Tutorials*, vol. 20, no. 3, pp. 2237–2263, thirdquarter 2018.

[11] L. Zhang, J. Liu, M. Xiao, G. Wu, Y. Liang, and S. Li, "Performance

analysis and optimization in downlink noma systems with cooperative full-duplex relaying," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 10, pp. 2398–2412, Oct 2017.

[12] G. Yang, M. Xiao, M. Alam, and Y. Huang, "Low-latency heterogeneous networks with millimeter-wave communications," *IEEE Communications Magazine*, vol. 56, no. 6, pp. 124–129, June 2018.

[13] 5GPPP, "View on 5G Architecture White Paper," Open Networking Foundation, Tech. Rep., 2017, accessed: 16 July 2019. [Online]. Available: https://5g-ppp.eu/wp-content/uploads/2018/01/5G-PPP-5GArchitecture-White-Paper-Jan-2018-v2.0.pdf

[14] Y. Wang, J. Xu, and L. Jiang, "Challenges of System-Level Simulations and Performance Evaluation for 5G Wireless Networks," *IEEE Access*, vol. 2, pp. 1553–1561, 2014.

[15] S. Ghosh, T. Dagiuklas, and M. Iqbal, "Energy-aware ip routing over sdn," in *2018 IEEE Global Communications Conference (GLOBECOM)*, Dec 2018, pp. 1–7.

BIOGRAPHIES

S. Ghosh obtains his M.E. in software Engineering and, B.Sc.(Hons.) & M.Sc. in Computer Science and M.Sc. in Smart Networks. He is also recipient of GATE and Erasmus-Mundus scholarship. Currently working towards his PhD in Computer Science & Informatics, supported by Marie-Curie Fund with the research area of Machine Learning's application in Self Organised SDN for 5G. His area of interests includes network programmability and automation, IoT and Deep Reinforcement Learning.

S. A. Busari received the B.Eng. and M.Eng. degrees in electrical and electronics engineering from the Federal University of Technology Akure, Nigeria in 2011 and 2015, respectively. He is currently an Assistant Lecturer with the aforementioned University, contributing towards the teaching of the undergraduate module in Electronic and Electrical Engineering. His research interests include wireless channel modeling, millimeter-wave and terahertz communication protocols, massive MIMO and system level simulation methodologies for 5G and beyond networks.

T. Dagiuklas is the Leader of the SuITE Research Group at London South Bank University. His research interests include smart internet technologies, media optimization across heterogeneous networks, QoE, virtual reality, augmented reality, and cloud infrastructures and services.

M. Iqbal, Senior Lecturer in Mobile Computing in the Division of Computer Science and Informatics, School of Engineering. He is an established researcher and expert in the fields of mobile cloud computing and open-based networking for applications in disaster management and healthcare; community networks; and smart cities.

R. Mumtaz, has more than 12 years R&D experience in the wireless communications industry, where his present appointment is Senior Research Scientist and Technical Manager at GS, Portugal. Prior to his current position, he worked as a Research Intern at Ericsson and Huawei Research Labs in 2005 at Karlskrona, Sweden. He received his MSc from the Blekinge Institute of Technology (BTH) Karlskrona (Sweden) in 2011, and obtained his Doctoral degree in 2016. His research interests include: 5G radio communication protocols and architectures, and Quantum Communications.

J. Gonzalez has more than 15 years R&D experience in mobile communications and practical experimentation. He obtained his MSc., and Ph.D degree in Telecommunications from the University of Surrey (UK) in 1999, and 2004, respectively. He then became a Senior Researcher at the University of Surrey, where he was responsible for project development and research on radio protocols. In 2011, he founded GS (Portugal), and currently is the Project Coordinator for the H2020-SONNET project. His research interests include: simulation methodologies, and 5G radio resource management.

S. Stavrou holds a BEng Degree in Computers and Communications from the University of Essex (U.K.), and a PhD degree in Electronic Engineering (Telecommunications) from the University of Surrey (U.K) where he was faculty member between 1997 - 2009. Currently he is the dean of faculty of pure and applied sciences at the Open University of Cyprus leading the Telecommunication Systems Research Lab. His research interests include localization, wireless sensor networks, smart cities and IoT networks. He is a Fellow of the Higher Education Academy U.K. and the chairman of the Cyber configuration of the European Security and Defence College.

L. Kanaris received a BSc in Aviation Science at the Hellenic Airforce Academy in Athens. After obtaining an MBA, he worked in several research projects of the Cyprus MoD and represented the Republic of Cyprus in several technical telecom workgroups of international organizations. He is a member of the R&D department of SIGINT. He has actively participated in various EU and national research projects investigating wireless positioning aspects. He is currently studying towards a PhD (TU/e) concentrating on wireless positioning techniques and the evaluation of fingerprint databases.