Using an ICN Approach to Support Multiple Controllers in OpenFlow

Ameen Banjar, Pakawat Pupatwibul, Abdallah AL Sabbagh, and Robin Braun
Centre for Real-Time Information Networks (CRIN)
University of Technology, Sydney
Sydney, Australia
{ameem.r.banjar, pakawat.pupatwibul}@student.uts.edu.au, {abdallah.alsabbagh, robin.braun}@uts.edu.au

***Abstract*— Information Centric Networking (ICN) is an innovative direction for next generation networks. It is a concept of networking paradigm which is considered as a new technique for future search activities. ICN is based on caching contents in several nodes and replicating these contents. It provides contents requested by users from the nearest node instead of creating a communication channel between sender and receiver just for calling information. This paper aims to scale OpenFlow network in traffic engineering by reducing number of transactions, predicting and pre-populating flow entries using the ICN approach. In addition, the paper shows the advantages of implementing ICN designs within OpenFlow. The proposed approach aims to implement ICN concepts to enhance OpenFlow network. This will enable the deployment of networking solutions in the existing network infrastructure and will lead to more flexibility in OpenFlow network. In addition, OpenFlow will have a global management view for all connected networks managed by different controllers. The proposed solution can fulfill current management and utilization of network demands. The paper then debates the implementation of ICN's design and features based on Software Defined Networking (SDN).**

***Index Term*— Named Data Object; Information Centric Networks; Software-Defined Networks; Next Generation Network.**

I. Introduction

Software Defined Networking (SDN) has become an interesting research area where OpenFlow instantiates the first standard implementation of this paradigm [1], [2]. SDN offers the network to be programmable. It allows programmers to change the network behaviour by implementing applications on the network's control plane which is called as a controller [3], [4]. This controller is logically centralized; however, it controls the connected distributed switches using its applications which can operate on a global network view [1]. SDN architecture provides three type of abstractions which are global management, state distribution and forwarding abstraction [5]. The global abstraction allows programmers to interact with the entire network instead of interacting with each individual node. The other abstraction is the state distribution, where a computational algorithm is created the central controller for the network process. The forwarding abstraction is provided to program network hardware through a common Application Programming Interface (API) [5], [6]. A challenging question is that, can the performance of the centralized controller achieve proper responsiveness and scalability. This paper argues that the control plane should be physically distributed to achieve those goals. This is because that the physical centralized controller is insufficient and has some limitations in terms of responsiveness and scalability [7], [8].

The main focus of this paper is to propose an approach that can improve the velocity process of forwarding, which is based on flow table that contains flow entries. The flow entries guide the flows to reach their destinations [9], [10], [2]. In addition, these flow entries can be updated by two methods: proactive and reactive [11]. The proactive rule starts when OpenFlow switches initiate connection with the controller which can pre-populate the flow table with all possible ingress and egress ports; therefore, the flows will be processed to reach their destinations [11]. However, when the OpenFlow switch receives an unknown flow, it will follow the reactive rule, which sends the flow to the connected controller where it processes and updates the flow entries [9], [2]. The implemented flow rules are performing properly. Therefore, the paper discusses the process of sending flows to another network managed by a different controller. In this scenario, the connection between controllers is not an OpenFlow protocol [9].

The increasing number of flows transaction between switches and controllers need to be resolved [8], [12]. When the flows are traversing to a different network managed by a different controller, it needs to be processed if there are no matching rules. This will lead to network traffic issues. Therefore, the Information Centric Networking (ICN) solution has been proposed to scale OpenFlow network.

The proposed approach has been motivated by the following advantages of OpenFlow. Firstly, the capability of experimenting new protocols in OpenFlow environment without disturbing existing organization network [2]. Secondly, OpenFlow has the flexibility on disturbances and failures [2]. In addition, the implemented solution will allow OpenFlow to have a global management view for all connected networks, including the networks connected to different controllers. Therefore, it is important to reduce the number of transactions between switches and controllers in order to provide faster delivery to the desired destination [12], [8].

Some approaches debate the limitation of current OpenFlow network such as HyperFlow [13]. However, HyperFlow is not sufficient in regards to forwarding flows in large scaled networks like data centers. Therefore, this proposed approach provides new type of communication between controllers, which is not supported by OpenFlow

protocol [10]. The communication in the proposed approach carries the ICN concept which is replicating flow entries among controllers. This will improve OpenFlow's global network management.

Because ICN is the concept of a new networking paradigm, many researchers used ICN in their approaches to replicate the data contents such as Data-Oriented Network Architecture (DONA) [14], Content-Centric Networking (CCN) and Publish-Subscribe Internet Technology (PURSUIT) projects [15], [16]. Whereas in our approach, the data objects is different than previous approaches where their data contents are web pages, video and documents, etc. The differentiation in the proposed approach is that the networking information such as packets is used instead of normal contents [16]. The packet will be embedded with an ICN-ID for the synchronization with the other connected controllers.

Adopting ICN concepts can provide enhancement in the forwarding process by synchronizing the network information between controllers. In ICN, a number of components can fulfill global network management such as Named Networking Objects (NNO), routing by ICN-ID, API, and ICN Database.

The remainder of the paper is organized as follows. Section II presents the related works to address the limitations of distribute network state upon control plane. In section III, the concept, components and some advantages of ICN approach is explained. The proposed approach for augmented OpenFlow network which is supported by using ICN is introduced in Section IV. This will lead to enhance traffic engineering and reduce transactions between switches and controllers. Finally, the paper is concluded in Section V.

II. Evaluation of Multiple Controller Approches

Several approaches in the literature have attempted to enhance the control plane by focusing on network global view. However, these approaches have limitations. In this paper, we introduce a number of factors that could impact on distributed control plane, such as timescale and number of interactions between switch and controller. In addition, the matching rules of micro and macro flows are discussed [17]. Moreover, proactive versus reactive flow regarding to populate flow table are considered [11]. Those impacts can be considered as motivation of our proposal.

This solution provides a new application that could be implemented on the controllers and over the API to communicate between controllers. In contrast, the related work falls in application and updated synchronization for optimizing functionality [18]. As an example, HyperFlow approach has multiple controllers to manage the entire network, where each controller is respectable for switches assigned to them [13]. HyperFlow is an application implemented on top of the NOX controller, using some concept of load balancer if one of the controllers fails. In addition, HyperFlow reuses an existing NOX application with minor modifications [19].

HyperFlow is based on passive synchronization of events between connected controllers for the network global view [13]. As a result, HyperFlow gives ability to each controller to manage the whole network. It duplicates the controllers rather than distributed state management of the control plane. HyperFlow is adequate for proactive rules and pre-populating flow table; however there are other challenges have not been covered [11]. For example, the reactive rules for updating flow entry need to be processed by the other controllers, if packet needs to. Moreover, HyperFlow focus on synchronizing events passively between controllers, which is limit to process the micro and macro, flows if it sends to a deferent controller [13], [17].

Likewise, Onix approach aims to have distributed controllers [20]. It provides programming API to communicate between network applications. It can also access the network by providing control logic using cluster environment that is responsible of distributing state of the network. Onix contents are distributed by using applications on top of controllers to coordinate for management and scalability purposes [20]. Onix deals with register per-packet instead of dealing with events such as HyperFlow, for less frequent in synchronization. Moreover, it provides a Network Information Base (NIB), which gives access to several state synchronization frameworks with different consistency and availability requirements. Thus, using the per-packet registration can scale large networks and provide flexibility for production deployments [20]. On the other hand, Onix approach lacks of timescale because of long processing time when sending the whole packet to other controller, which is not faster than transaction between switch and controller. Also, it does not support proactive rules where it uses per-packet registration [11]. Onix just focus on processing the micro and macro flows as distributed state management [17].

Light upon these two approaches, distributed control plane in HyperFlow and Onix are not reliable in large data centers, as they are not focused on reducing the timescale and the number of transactions between switches and controller [13], [20]. Thus, our approach instantiation the centralized paradigm with centralized benefits. However, it is scalable and physically distributed with the capability of distributed state management of controllers. In addition, we use updated synchronization that can operate on events, packets, and flows with considering the proactively and reactively, micro and macro flow management and timescale for transactions.

III. Information Centric Networking for State Distribution Management

OpenFlow can solve the problems of current networking paradigms by deploying the concepts of ICN, in terms of scalability and responsiveness.

Flow processing is the core function of the OpenFlow switch, and the independent process by the OpenFlow switch and controller will be more flexible by integrating ICN concepts [9]. Thus, the core idea is to distribute the network state to different networks managed by other controllers, aiming to achieve the goals [21].

Integrating the 12 matching fields of OpenFlow with ICN-ID by wrapping them for synchronization purposes will enhance the packet processing in the current OpenFlow

network. Moreover, there are two ways of matching a packet, one is matching micro flow and the other is matching macro flow [17]. Where fields assigned by the controller, in macro for example can be just IP distention and the rest are wildcard. However, for the micro flows every field assigned specifically for matching [9]. Thus, the significant impact on packet processing is done by the control plane, so our approach aims to enhance that without caring about hardware.

For distributed network state, we employ the ICN approach to establish communication between controllers, and use the following basic ICN functionality: NNO, routing by ICN-ID, API, and ICN Database. These main elements are implemented using application hosted within each controller. This section will describe these main components, which were the most important aspects of the proposed design and other ICN approaches. Firstly, the Named Networking Objects (NNO) is considered as the main abstraction of ICN. For example, the MAC addresses, IP addresses, Ethernet type and all network information required for networking that can be accessed from the global management view. NNO has a unique name and place in the database. Each copy of the same NNO will have its own name and location in a deferent database and can serve any requests and inform other controllers. Secondly, in this approach the ICN-ID is directly routed to the request message from the requester to one or multiple controller within the network. The routing algorithm is also explained. After the source has received the request message, the data is routed back to the controllers, which requested those messages. Thirdly, API (Application Programming Interface) – this defines the request and respond messages to NNO. The other controllers create the NNO to be ready for any requests from other controllers. In this approach, the synchronization and corresponding happens if any new state is available including ICN-ID and location for accurate matching for the flows arrival. Finally, the ICN Database will store all network state and NNOs and accumulate processes for ICN approach, where all controllers has its own database and can serve all other controllers at initialization phase.

ICN can bring many advantages to OpenFlow networks. For example, ICN can provide network scalability by distributing the flow information and network state across different networks managed by multiple controllers. Moreover, in case of controller failure, all the affected switches can be reconfigured to any active nearby controller. Individual controller will directly manage the connected switches as well as indirectly programs or queries the rest of network state not only in the initialization phase but also upon unknown packet arrives.

IV. OpenFlow Controller Enhancement Algorithm

This section proposes an algorithm by adopting the ICN concepts for distributed network state among multiple OpenFlow controllers. It also describes a simulation scenario of the approach including flowcharts of initialization phase and packet processing.

A. The proposed simulation scenario

This subsection demonstrates the forwarding capabilities of multiple OpenFlow controllers by deploying the ICN framework to synchronize the events in run time at a line rate. Each controller implements an application to collect its own network state and install them to the ICN Database. Source controller publishes the messages to other controllers via ICN interface. On the other hand, other controllers will reply to messages that are published in the source controller.

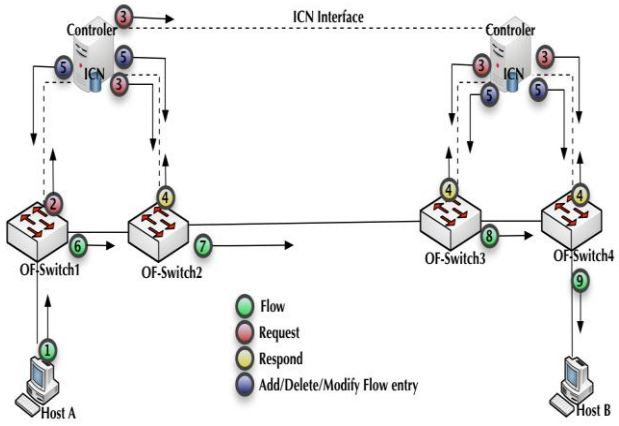

Fig. 1. Forwarding scenario overview via ICN interface

As shown in Fig. 1, Host A needs to send a packet to Host B, which belongs to a different network. When the first packet arrives at an OpenFlow switch (we name this first packet a "flow request"), the first packet is forwarded to the controller. This is because there are no flows configured in the switch's flow table that match the packet. The controller then sends *flooding* messages to all managed switches and retrieves the switches information for collecting the network state and install them in the ICN Database. Connected controllers then initialize communication and send *Request/Response* messages to synchronize network state according to each controller's ICN Database. Therefore, at this stage each controller should have a network-wide state of all managed switches. Because the desired destination Host B is in a different network, the Source controller needs to inform other controller that the corresponding flow will arrive. As a result, when this flow reaches to another network, it does not need to send a *flow request* to the controller again because all the necessary flow entries were already installed to every switch along the chosen path. This can significantly reduce the number of transactions between the switches and the controller.

B. Flow chart of packet processing

The flowchart of integrating the ICN approach is presented in this subsection. It consists of two phases. More details on these phases are shown below. Multiple OpenFlow controllers which are managing different networks can be more scalable by updating wide network information and the ICN Database.

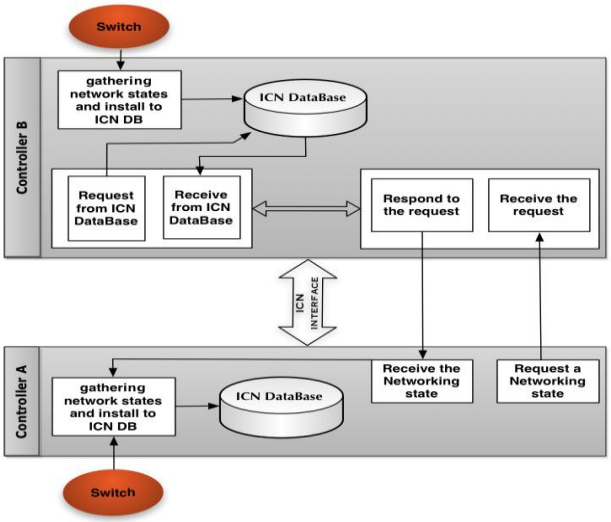

Fig. 2. Flowchart detailing the initialization phase.

Each controller will perform initialize communication with the neighbor controllers by sending a *Request_Info* to get the neighbor's network state and also to publish its own network state to neighbor controllers. The Source controller will publish NNO response via API using TCP or TLS to all request from other connected controllers. In more details, the corresponding controller on the other hand will grab its own network state from the ICN Database and send NNO response back. The same process in reverse will occur to other controllers. At this point, the received network state information will then be stored in each of the controller's ICN Database.

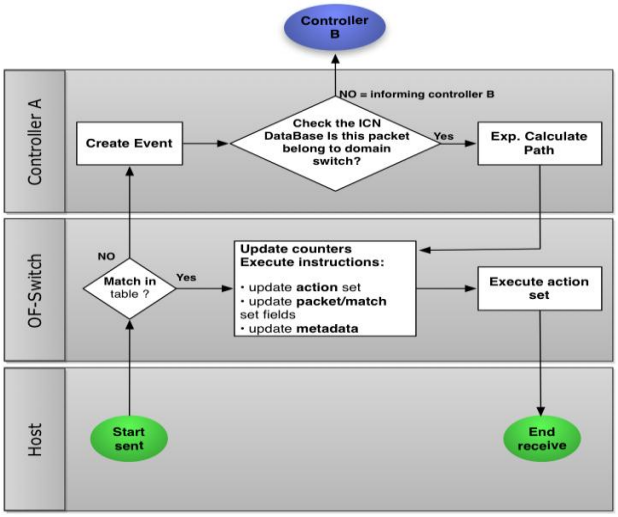

Fig. 3. Flowchart detailing the packet processes.

Whenever an unknown packet arrives at a certain controller, it will first create an event and check the ICN Database whether the destination host belongs to its managed switches. If YES, the controller will compute the path and install the flow entries to their switches. If NO, the controller will send a message informing the other controller with the destination host that this particular flow will arrive. The destination controller will then compute the routing and install flow entries to its managed switches according to the network state stored in its ICN Database. As a result, when an incoming packet arrives at a switch of a different network, it does not need to send *flow request* to the controller again for high-level decision makings. The flow will be matched to the predefine rules installed earlier.

C. Multiple OF controller application algorithm

The algorithm for exchanging network state between multiple controllers is presented in this subsection. More details on the algorithm of this application are shown below. Multiple OpenFlow controllers can be optimised to install necessary flow entries for the selected OpenFlow switches by initialising communication and exchanging network state between connected controllers. This algorithm can be tested in labs or can be experimented by using cloud service companies such as Amazon web service.

Algorithm 1 Multiple OpenFlow Controllers Application

 # Gather (initialization) all state view for global networks
Gathering
GatherSwitchesState (switches);
LearnedMAC (mac addresses);
Install networks state into each ICN database;
Request networks state from other ICN databases;
Update ICN database with the replied networks states;
End Gathering
 # Packet Processing (sent by OpenFlow switch)
for all p (where p is packet request sent by OpenFlow switch to the controller)
if packet is transmitted to a destination that belong to another controller
send packet's header to the destination's controller;
update the destination's controller's ICN database;
create flow entries in the original controller;
install flow entries to chosen path switches within the original controller;
create flow entries in the destination's controller;
install flow entries to chosen path switches within the destination's controller;
else
create flow entries;
install flow entries to chosen path switches;
end if
end for

V. Conclusion and Future Works

This paper proposes an algorithm to enhance multiple OpenFlow controllers by integrating ICN concept. The proposed approach aims to support the distributed network state for all connected networks, which uses different controllers for network management. It also aims to reduce the interaction between the switches and controllers in order to scale both physical distributed controllers and logical centralized paradigm. The paper has described the concepts of

ICN framework and shows how network state can be distributed between different networks managed by multiple controllers. The proposed packet migration solution will hopefully enhance these controllers to determine the appropriate switches that need the installation of flow entries. Therefore, it is expected to see a new generation of cloud computing for logically centralized controller which is flexible and easy to maintain and may even inspire interesting proposals from the OpenFlow community showing a wide range of innovation opportunities. The future work will be an implementation of the designed algorithm and a run of experiments using ICN approach in cloud services.

ACKNOWLEDGMENT

This work is sponsored by the Centre for Real-Time Information Networks (CRIN) in the Faculty of Engineering & Information Technology at the University of Technology, Sydney (UTS).

REFERENCES

[1] S. Shenker, M. Casado, T. Koponen, and N. McKeown, "The future of networking, and the past of protocols," *Open Networking Summit*, 2011.

[2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, pp. 69-74, 2008.

[3] M. Canini, D. Kostic, J. Rexford, and D. Venzano, "Automating the Testing of OpenFlow Applications," in *Presented at: The 1st International Workshop on Rigorous Protocol Engineering (WRiPE)*, 2011.

[4] P. Pupatwibul, A. Banjar, A. AL Sabbagh, and R. Braun, "An Intelligent Model for Distributed Systems in Next Generation Networks," in *Advanced Methods and Applications in Computational Intelligence, Topics in Intelligent Engineering and Informatics*, vol. 6, Springer International Publishing, Switzerland 2014, pp. 315-334.

[5] G. Goth, "Software-Defined Networking Could Shake Up More than Packets," *Internet Computing, IEEE*, vol. 15, pp. 6-9, 2011.

[6] A. Voellmy, A. Agarwal, and P. Hudak, "Nettle: Functional Reactive Programming for OpenFlow Networks," DTIC Document2010.

[7] Z. Cai, A. L. Cox, and T. S. E. Ng, "Maestro: A System for Scalable OpenFlow Control," Technical Report TR10-08, 2010.

[8] A. R. Curtis, J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, and S. Banerjee, "DevoFlow: scaling flow management for high-performance networks," *SIGCOMM-Computer Communication Review*, vol. 41, p. 254, 2011.

[9] Open Networking Foundation, "Openflow Switch Specification," version 1.3.0 (Wire Protocol 0x04), September 2012.

[10] A. AL Sabbagh, P. Pupatwibul, A. Banjar and R. Braun, "Optimization of the OpenFlow Controller in Wireless Environments for Enhancing Mobility," in *13th IEEE International Workshop on Wireless Local Networks (WLN'13), in conjunction with the 38th IEEE Conference on Local Computer Networks (LCN 2013)*, Sydney, Australia, October 21-24, 2013.

[11] B. Salisbury. (2013, 20-04-2013). *OpenFlow: Proactive vs Reactive Flows*. Available: http://networkstatic.net/openflow-proactive-vs-reactive-flows/

[12] M. Yu, J. Rexford, M. J. Freedman, and J. Wang, "Scalable flow-based networking with DIFANE," *ACM SIGCOMM Computer Communication Review*, vol. 41, pp. 351-362, 2011.

[13] A. Tootoonchian and Y. Ganjali, "HyperFlow: A distributed control plane for OpenFlow," in *Proceedings of the 2010 internet network management conference on Research on enterprise networking*, 2010, pp. 3-3.

[14] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in *ACM SIGCOMM Computer Communication Review*, 2007, pp. 181-192.

[15] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, 2009, pp. 1-12.

[16] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *Communications Magazine, IEEE*, vol. 50, pp. 26-36, 2012.

[17] B. Salisbury. (2013, 20-04-2013). *OpenFlow: Coarse vs Fine Flows*. Available: http://networkstatic.net/openflow-coarse-vs-fine-flows/

[18] D. Levin, A. Wundsam, B. Heller, N. Handigol, and A. Feldmann, "Logically centralized?: state distribution trade-offs in software defined networks," in *Proceedings of the first workshop on Hot topics in software defined networks*, 2012, pp. 1-6.

[19] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "NOX: towards an operating system for networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, pp. 105-110, 2008.

[20] T. Koponen, M. Casado, N. Gude, J. Stribling, L. Poutievski, M. Zhu, R. Ramanathan, Y. Iwata, H. Inoue, and T. Hama, "Onix: a distributed control platform for large-scale production networks," in *Proceedings of the 9th USENIX conference on Operating systems design and implementation*, 2010, pp. 1-6.

[21] F. Long, Z. Sun, Z. Zhang, H. Chen, and L. Liao, "Research on TCAM-based Openflow switch platform," in *Systems and Informatics (ICSAI), 2012 International Conference on*, 2012, pp. 1218-1221.

Ameen Banjar received his B.Sc from Taibah University (Saudi Arabia) and M.I.T advanced from University of Wollongong (Australia). He is currently working towards Ph.D. degree in Computing and Communications, at University of Technology Sydney UTS, Faculty of Engineering and Information Technology. He began his working career as a database designer and programmer at Taibah University, Information Technology Centre (ITC) in Saudi Arabia, for two years. He has a research interest in network management, especially in the area of intelligent agent-based network management systems.

Pakawat Pupatwibul is currently working towards the Ph.D. degree in Information Systems, faculty of Engineering and IT from University of Technology Sydney, Australia, having graduated from Naresuan University with a B.Sc. in Computer Science, and Master's of Information Technology from UTS. He has worked attentively as a network administrator for Suan Dusit Rajabhat University, a government sponsored university in Thailand, for 7 years. His research interests include next generation networks, data center

network, QoS and network management, especially in the area of intelligent agent-based network management systems.

Abdallah AL Sabbagh received his B.Sc in Information Technology and Computing (Hons) on 2006 from The Open University (UK), and a Master of Engineering Studies (MES) in Telecommunication Networks and a Ph.D. in Telecommunication Engineering on 2010 and 2013 respectively from the University of Technology, Sydney (UTS), Australia. He is currently a Lecturer at the School of Computing and Communications and a Research Engineer at the Centre for Real-time Information Networks (CRIN) within the faculty of Engineering and Information Technology (FEIT) at UTS. His recent research is in the area of networking and distributed computing systems including: next generation networks, heterogeneous wireless networks, Software Defined Networking (SDN), Information-Centric Networking (ICN) and mobile Internet Protocol (IP) networks.

Robin Braun received his B.Sc (Hons) from Brighton University (UK), and his M.Sc and Ph.D from the University of Cape Town. He holds the Chair of Telecommunications Engineering in the Faculty of Engineering and Information Technology of the University of Technology, Sydney, Australia. He is an executive member of the Centre for Real Time Information Networks (CRIN) at the University of Technology, Sydney (UTS). Prof. Braun was a member of staff of the Department of Electrical Engineering of the University of Cape Town from 1986 to 1998. He was the founder, and Director of the Digital Radio Research Group at the University of Cape Town, which supervised over 50 research degree candidates in the years that he was attached to it. Prof. Braun is currently a Senior Member of the Institute of Electrical and Electronic Engineers of the United States (IEEE).