

On the Benefits of RINA over Programmable Optical Networks for Dynamic and Smart Resource Management

Sergio Leon¹, Jordi Perelló¹, Eduard Grasa², Davide Careglio¹, and Salvatore Spadaro¹

¹Universitat Politècnica de Catalunya (UPC) - BarcelonaTech, Barcelona, Catalunya, Spain

²i2CAT Foundation, Barcelona, Catalunya, Spain

ABSTRACT

In this paper, we describe the basics of the Recursive InterNetwork Architecture (RINA) and its advantages as a packet transport technology over programmable optical networks. In particular, its seamless integration over an SDN-controlled network infrastructure and its intrinsic capability of providing QoS is analysed and further discussed.

Keywords: recursive network model, programmable networks, optical networks.

1. INTRODUCTION

Traditional network models lack the dynamism and automation needed to respond to the increasing requirements of fast bandwidth-on-demand services of nowadays' markets. Typically, this lack of dynamism is counteracted through a large overprovisioning of resources. Such overprovisioning is a common method adopted in optical networks, where high bandwidth demands are usually allocated for core backbones and inter-datacentre networks services. It implies increased Capital and Operational Expenditures (CapEx and OpEx), thus making it something neither scalable nor desirable. In addition to the lack of dynamism, networks tend to be highly heterogeneous, making their interoperability and automated configuration difficult, requiring either manual intervention or configuration via ad-hoc scripts, making any operation slow, not optimal and error prone.

In response to the aforementioned problems, the paradigm of programmable networks introduces a whole new family of networking models, where control and physical resources are clearly separated in order to provide a better control of the network. Specifically, Software-Defined Networking (SDN) [1] provides an abstraction of resources and functionalities to network administrators and allows the centralization of the network control. This centralization allows fast network configuration and automation, reducing overprovisioning and removing possible configuration errors.

Although SDN networks can solve the problem of controlling and configuring highly dynamic bandwidth demands, the current Internet model suffers of *ossification* [2], a condition where any technological innovation meets natural resistance, as exemplified by the lack of wide-scale deployment of technologies such as multicast or IPv6. In this context, the Recursive InterNetwork Architecture (RINA) [3][4][5] is a clean slate Internet solution that focuses on providing the service that applications require over a recursive and fully configurable architecture. The recursive architecture and the knowledge of application requirements makes RINA a smart Internet model capable of knowing how new applications with new requirements can be served better, and what services it needs from the underlying physical layers. When working with backbone networks, this extra knowledge about requirements can be used to directly inform the network controller about these requirements, making then faster and better decisions on resource allocation.

In this paper we give a brief look at both SDN and RINA and show how their joint operation can improve future data transport networks, having a smart network that focuses on applications requirements and can be configured fast and automatically, optimizing then the allocation of resources.

The rest of the paper is organized as follows. Section 2 introduces programmable networks and SDN. In Section 3, we introduce RINA and its benefits against the current Internet model. Section 4 shows some use cases of RINA/SDN. Finally, Section 5 concludes the paper.

2. PROGRAMMABLE NETWORKING AND SDN

In traditional computer network architectures there are two distinct planes that work jointly in order to provide the desired communication services, namely, the data plane (or forwarding plane) and the control plane. The data plane is strongly linked to the hardware and focuses on making fast local decisions to provide the connectivity, forwarding, error detection, etc., needed by the services using the network. On the other hand, the control plane has to make global and smart decisions and configure the data plane in order to ensure that the requirements of the services are effectively provided to the supported data flows.

In conventional distributed networking, devices tend to be vertically integrated black boxes, where the control and data planes become tightly bounded into the same physical device. The data plane typically consists of multiple devices of different vendors, with distinct capabilities, performance, interfaces, etc., composing thus a heterogeneous plane. Given this heterogeneity and the lack of common interfaces, they are usually configured manually using proprietary management systems. As a result, the overall management of a network composed by several devices is something not trivial and prone to errors.

In contrast to traditional networking, the paradigm of programmable networks focuses on a clear distinction between control and data planes, providing a common interface between them. This clear separation enables the placement of the control plane outside the devices, so that the control decisions can be centralized in a single entity with a global knowledge of the network status. Specifically, the most important benefits of programmable networks against traditional networks are:

- Fast and automated response to failures and changes on the services' requirements.
- Optimized allocation of bandwidth and resources.
- Improved flexibility to topology/device/protocol changes by separating the control from the hardware.
- Reduced cost of network devices by removing the computational needs of the control plane.

2.1 SDN and OpenFlow

As aforementioned, SDN is a new networking paradigm in the context of programmable network family that allows network administrators to perform management decisions through an abstraction of the functionality and resources of the lower levels (i.e., the hardware) [6]. SDN divides the control plane into the controller layer, which manages the data plane, and the application layer, where business logic is located and service demands are generated (Fig. 1).

While the northbound API that connects the SDN controller with the application layer consists primarily in ad-hoc solutions (although efforts are being made to make it more standardized), for communication via the southbound interface there are only few protocols commonly used, such as OpenFlow [7]. OpenFlow, developed by the Open Networking Foundation (ONF), is an industry standard for the southbound interface communication, considered to be the first and most well-known southbound interface for packet switching networks, providing a common interface used by most of big vendors (HP, IBM or Infinera are some of the vendors producing OpenFlow enabled devices).

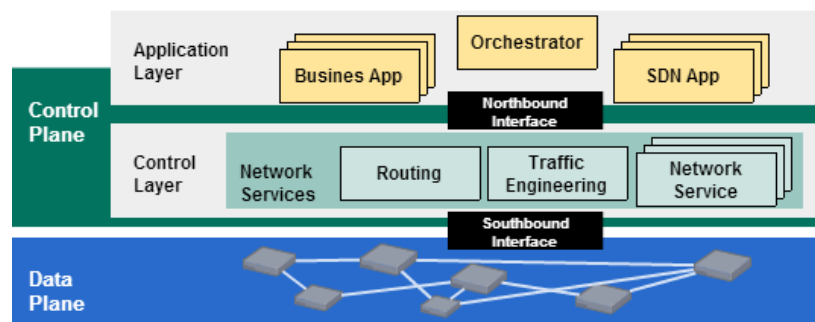


Figure 1. SDN architecture diagram.

2.2 SDN in optical networks

While optical networks are critical for handling the high bandwidth demands and low latency that are needed at core and datacentre networks mainly, they traditionally tend to be pretty static and difficult to configure. Given this, network administrators are forced to overprovision resources in order to support short flows, or flows that have a large variance on bandwidth usage at distinct time frames, etc., to use the optical resources, something highly expensive given the cost of this type of networks.

An SDN-enabled optical network can perform changes in less than a minute, changes that could take weeks in traditional networks [8]. That allows for network orchestrators to optimize periodically the allocation of optical resources. Given that, SDN can yield much higher resource utilization, thus removing the needs of a high overprovisioning and extra network costs.

As SDN was initially proposed for packet networks, its implementation in optical networks is in an earlier state. While OpenFlow was designed specifically for packet networks, a modified version of it is already in process, making the changes required for it to work with optical networks. Given that, in the future, OpenFlow could then provide a common standard for southbound interfaces towards any physical devices.

3. RECURSIVE INTERNETWORK ARCHITECTURE (RINA)

RINA is a clean slate Internet model based on the idea that networking is Inter-Process Communication (IPC) and only IPC [3]. RINA presents a recursive layering of Distributed IPC Facilities (DIF), where each DIF has the same invariant mechanisms and in/out interfaces, but their specific operation can be completely configured via policies (Fig. 2a). In RINA, applications can request flows with distinct requirements, like minimum ensured bandwidth, maximum delay, forced order, maximum probability of dropping, etc.

Within the same system, an IPC Process (IPCP) of a DIF N is considered an application in N-1 DIFs. It means that it is a recursive model where the same structure is kept at each layer and applications simply request flows with some QoS requirements that need to be ensured within that scope. These QoS are defined by policy, so that

each DIF can define these sets in a way that is best suited to the policies in use for scheduling, routing, resource allocation, etc., that allows a best provisioning of resources for flows within each, as each DIF can define them as the set that better represents its capabilities and services that it can provide.

Two of the RINA main targets are scalability and security, something that accomplishes to a large degree thanks to its recursive layering and an addressing. Addressing in RINA is based on Jerry Saltzer's idea [9] that in a good addressing scheme, naming (who), addressing (where) and routing (how) should be separated in a way that a name does not necessarily implies address nor addresses how to reach the destination, avoiding them the need of default ports or tie routing to interfaces. Given this type of addressing scheme, flows between neighbour nodes in a given N-level DIF must be requested and allocated via N-1 DIFs in order to know the N-1 address and port, which not only increases security within the network, as data is only accepted from allocated ports, but also facilitates monitoring flow requirements, something that can be used to better ensure their needs.

Figure 2a) depicts the modules and policies of an IPCP. The IPCP components are naturally organized in i) groups of functions that are simple, but performed more often (data transfer); ii) groups of functions that are more complex, but performed less often (data transfer control) and iii) groups of functions that are the most complex, but also performed less frequently (layer management). The details of all components can be found in [5].

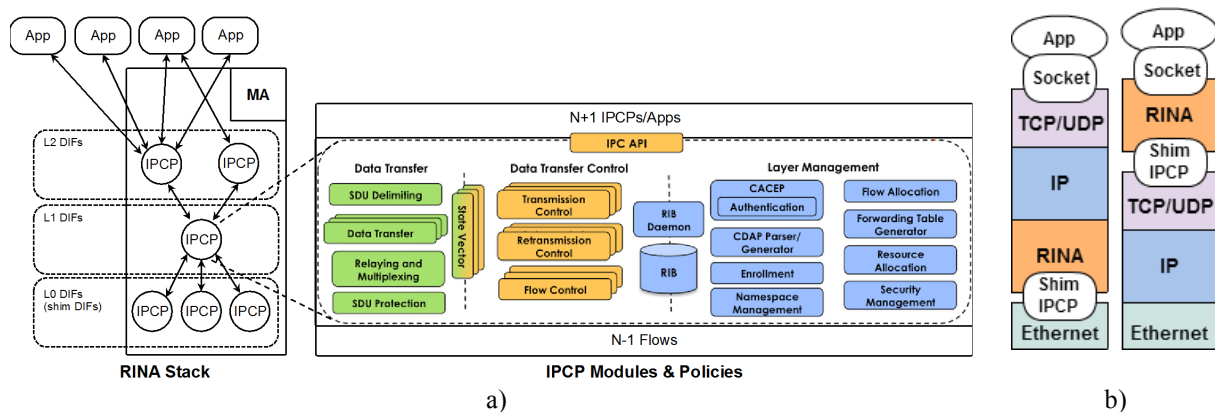


Figure 2: a) RINA Stack and IPCP Modules/Policies Structure;
 b) IP over RINA and RINA over IP stacks.

Given the recursive layering structure of DIFs, the addressing schemes and the configurable policies, RINA represents a smart substitute for the current IP model. Although RINA is a clean slate model clearly distinct to the current model, it is compatible with it. Indeed RINA does not require turning off the entire Internet to be deployed, but allows a progressive migration at little expense. Such a migration may start either with the replacement of lower layers with RINA while keeping IP networks and services on top of it, or with any network protocol (e.g. Ethernet, WDN, TCP, IP, UDP, etc.) used as a layer 0 in RINA, using a shim IPCP specific for that protocol (see Fig. 2b).

4. BENEFITS OF RINA OVER OPTICAL SDN ENABLED NETWORKS

As aforementioned, one of the benefits of programmable network and SDN in particular is that it makes network configuration something easier and less prone to error thanks to the centralized decisions and its large degree of automation. Nonetheless, much of these enhancements are limited by the constraints of the Internet model itself, as it was designed considering neither the current level of dynamism nor the current applications requirements. RINA takes a new step forward, providing a networking model that considers both how to use the resources and where to place the functionalities according to requirements of the applications. Besides, RINA can also monitor the current allocated flows, making then possible to request directly the requirements to the lower layers without needing human interaction.

Given the possibilities of both SDN and RINA, a lot of scenarios could benefit of a RINA/SDN architecture.

One of the more critical environments nowadays is datacentre network. It is indeed one of the scenarios that can be improved better with the use of RINA/SDN. First, RINA naming/addressing schemes and recursive architecture, allows to, for example, easily isolate VMs of distinct customers/applications, provide distinct QoS depending on the application requirements, move VMs between hosts without affecting applications, etc. Having flows of distinct customers/applications aggregated in lower layers allows RINA to have a great overview of the network requirements that can be shared between the managers of the core routers. With this information, RINA layer can request an SDN shim DIF for new requirements, which will speak directly with the SDN controllers via its northbound interface and dynamically allocate resources when needed.

