WHITE PAPER

Service Providers

Optimisation of Edge Networks and their Distributed Applications

Authors

Radhika Loomba
Intel Labs Europe

Keith A Ellis
Intel Labs Europe

Johan Forsman
TietoEVRY

Frank Fowley
DCU

Theo Lynn
DCU

Sergej Svorobej
DCU

Peter Willis
BT

Table of Contents

1. Document Overview

This document describes the components of RECAP, a novel integrated architecture and optimisation framework for optimising edge networks and distributed applications deployed on them. Models for understanding capacity provisioning for distributed edge networks are described, followed by a solution for intelligent and dynamic resource placement optimisation and orchestration. This is illustrated through a case study of an NFV-based LTE network. The results show how the proposed RECAP architecture can be used to optimise customer QoS and QoE while at the same time minimising infrastructure provider CAPEX and OPEX.

2. Audience

This white paper is intended for service providers, network operators, network monitoring and management solution providers or those planning to deploy automation solutions for service and application deployment on distributed clouds and edge networks.

3. Introduction

The traditional cloud[1] topology is not designed for emerging services in the 5G era. The original cloud computing concept was based on the economies of scale achieved through centrally hosting multi-tenant applications on elastically scalable pay-as-you-use virtualised resources. Recent advances in mobile radio technology, Internet of Things (IoT) and network services, such as over-the-top (OTT) streaming, have resulted in an explosion of data for storage and processing on core cloud infrastructure. This has resulted in very peaky traffic and increased network capital expenditure (CAPEX) and operating expenditure (OPEX) for operators. Simply, the traditional cloud topology does not suit some services, for example, low-latency use cases or streaming live events. The net result has seen a proposed move towards fog[2] and edge[3] computing.

There is a need for edge networks to exhibit the economies of scale that were characteristic of the original cloud computing business case. To justify the capital investment, the edge must be efficiently utilised and capable of serving multiple services from multiple tenant operators. Moreover, the complexity of distributed edge infrastructure leads to a need for automated orchestration of applications, providing end-to-end management of networking, infrastructure and workload placement.

The fundamental challenge is to balance 1) the need for dynamically deployable applications that meet specific targets, often relating to geographically and temporally dynamic usage profiles, and 2) the need to minimise the cost and energy consumption of finite physical hardware resources, and 3) the need to adhere to the service level agreement (SLA) commitments of infrastructure service availability and performance.

This white paper describes an integrated architecture and optimisation framework implemented to address these challenges. The approach was developed within

the H2020 RECAP project, in which Tieto, British Telecommunications Plc, Dublin City University and Intel were partners. The framework utilised an automated optimisation process whereby near real-time and future application placement and infrastructure optimisation is achieved, balancing application provider and infrastructure provider interests, optimising the placement of resources and functionality to improve performance, reduce cost and/or avail of specific hardware features. The paper outlines the application of the proposed framework to network function virtualisation (NFV) management.

4. Design Principles

As the number and type of service chains, resources, components and customers increase, the possible mapping permutations become too great to be practically tractable. Essentially, the problem becomes one of mixed criticality, complexity, heterogeneity and scale, which can only be addressed through the adoption of (i) machine learning and general algorithmic approaches, and (ii) a model-centric architecture design.

4.1 An Algorithmic Approach

Given the significant challenge at hand, it will come as no surprise that there is no 'silver bullet'. Rather, a framework is required that can accommodate multiple algorithmic approaches. For example, the approach taken in this NFV use case is to utilise a stochastic evolutionary algorithm to identify sub-optimal, that is, good enough solutions. These are then programmatically evaluated relative to key performance indicators (KPIs) using utility functions. The quality of any selection is determined by its fitness relative to constraints (a zero-fitness meaning a best placement with no constraint breaches). The algorithm is fed a set of candidate selections and the fitness is calculated. Candidates with lower fitness values are retained. Then, either new candidates are added, or existing candidates are mutated. The process ends when there is a placement with zero fitness (zero breached constraints) and the placement with the highest utility is chosen.

The algorithm and utility function comparison must balance two potentially conflicting objectives: one being provider-centric (that is, infrastructure provider view), and the other being user-centric, as represented by the application service provider. Provider-centric optimisation objectives include cost, distribution, capacity planning, SLA, energy, data location, reliability and security. User-centric objectives include throughput, latency, response time, service availability, service creation time and service restoration times.

4.2 Model-centric approach

Model centrism is a design principle that uses machine-readable, highly abstract models developed independently of the implementation technology and stored in standardized repositories (Kleppe et al. 2003). For capacity provisioning of distributed clouds, at least six models are needed: (i) infrastructure models, (ii) user models, (iii) workload distribution models, (iv) workload translation models, (v) application models, and (vi) quality of service (QoS) models.

Infrastructure Models are graphs that present the physical and virtual structure, configuration and topology of a given network and are called 'landscapes'. Landscapes together with telemetry and KPIs are also needed for infrastructure optimisation and simulation. These models may be configured to reduce the granularity of the system to make optimisation practicably achievable.

User Models are based on an agent-based modelling of users, for example, citizens navigating through a city and utilising mobile services or people at home using the Internet. Data from the city of Umea in Sweden was used which included information about the geographical topology of the city and its areas, population densities and their statistical communication preferences, as well as radio access network cell site locations and topology.

Workload Distribution Models are informed by **User Models.** Workload distribution accounts for the mobility of application components and the impact of component migration on application performance. They are based on the results of load balancing after a component migrates and on user mobility models, which drive component migration.

Workload Translation Models are used to map application load configurations (output of application optimisation) to physical capacity. They correlate the virtual resources (virtual machines (VMs)/containers) to physical resources, and the physical resource utilisation with the application component KPIs (throughput, response time, availability, speed of service creation and speed of service remediation). They provide a mapping of actual (specific in time) telemetry metrics of physical resource consumption (utilisation metrics) to application components workloads (that is, the utilisation of resources by the components that are running on those physical machines). Effectively, this maps the application placement with the performance of components so placed.

The workload models, together, describe the relationships between control and data plane traffic, between end-to-end latency and traffic, and between traffic and resource usage. They were built based on the data analysis of historical trace and synthetic workload data using statistical and machine learning techniques. These models are used in application optimisation to calculate the cost of component migration when selecting an optimisation option. Applications are described, in this context, as graphs of components with interdependencies and constraints in the form of graph links. Application components are split into front-end and back-end layers (modelling load balancing within components and management of component functionality, respectively) that can be autoscaled independently.

QoS models are based on the definition of specific QoS characteristics, which are measurable and related to end-user experience. A QoS model needs to be built specifically for an application and the associated deployment scenario, setting the QoS requirements and defining the metrics to be used. Service QoS targets need to be mapped to the performance metrics related to these distributed application components and the underlying physical infrastructure. In this case, for example, the model needs to calculate the aggregate packet loss for a VNF or virtual

link, which is the combined packet loss for each forwarding graph passing over that component. The model must support the associated packet buffering and queueing, and the packet loss needs to be determinable from the metrics provided by the component.

Using these models, workload arrival patterns can be augmented to predict how a load propagates through distributed applications, and how the component workloads impact the resources running them.

5. An Integrated Architecture and Optimisation Framework

RECAP provides a modelling framework and integrated architecture that can be used to facilitate an optimal deployment of a cloud/edge network based on agreed cost, performance and quality constraints.

Figure 1 below outlines the components in the RECAP architecture and shows the process flow loops in the optimisation framework. The **Landscaper Component (1)** acquires information on the state and configuration of the physical and virtual infrastructure resources and represents the same in a graph database. The **Monitoring Component (2)** uses probes to collect telemetry metrics needed for the modelling and optimisation tasks, including CPU consumption, disk I/O, memory loads, network loads and packet statistics as well as virtual network function (VNF)-specific counters, both from virtual and physical resources. These are input to the optimisers and the output is used to orchestrate and enact resource changes in the cloud network.

The **Application Optimiser (3)** is used to optimally derive application configurations, for example, service chains that meet a given user workload. Additionally, it is responsible for autoscaling, both horizontal scaling (by adding more virtual machines (VMs) into a pool of resources) or vertical scaling (whereby one adds more power (CPU, RAM, etc.) to an existing VM). All of which must be supported by the underlining infrastructure. Applications can be scaled locally or globally and may be in response to run-time traffic limits or resource levels being reached or may be controlled by data analytic workload predictive systems.

The application to be deployed is composed of multiple connected service components in the form of service function chains (SFC), which need to be placed together. In order to achieve optimal/sub-optimal application deployment onto a distributed virtual cloud infrastructure, it is necessary to introduce sufficient functional granularity into the application structure to allow separate components to be provisioned and scaled independently.

Application optimisation is essentially a best-effort graph mapping of application components and dependencies to the network of computing resources based on information available to the **Application Optimiser (3)**. The mapping is done subject to application-specific rules or constraints relating the individual resource requirements for components (minimum/maximum instance constraints) and their mutual co-hosting needs (affinity/anti-affinity constraints). The overall aim is to deliver an optimal overall KPI target such as maximum latency or minimum throughput or maximum usage cost.

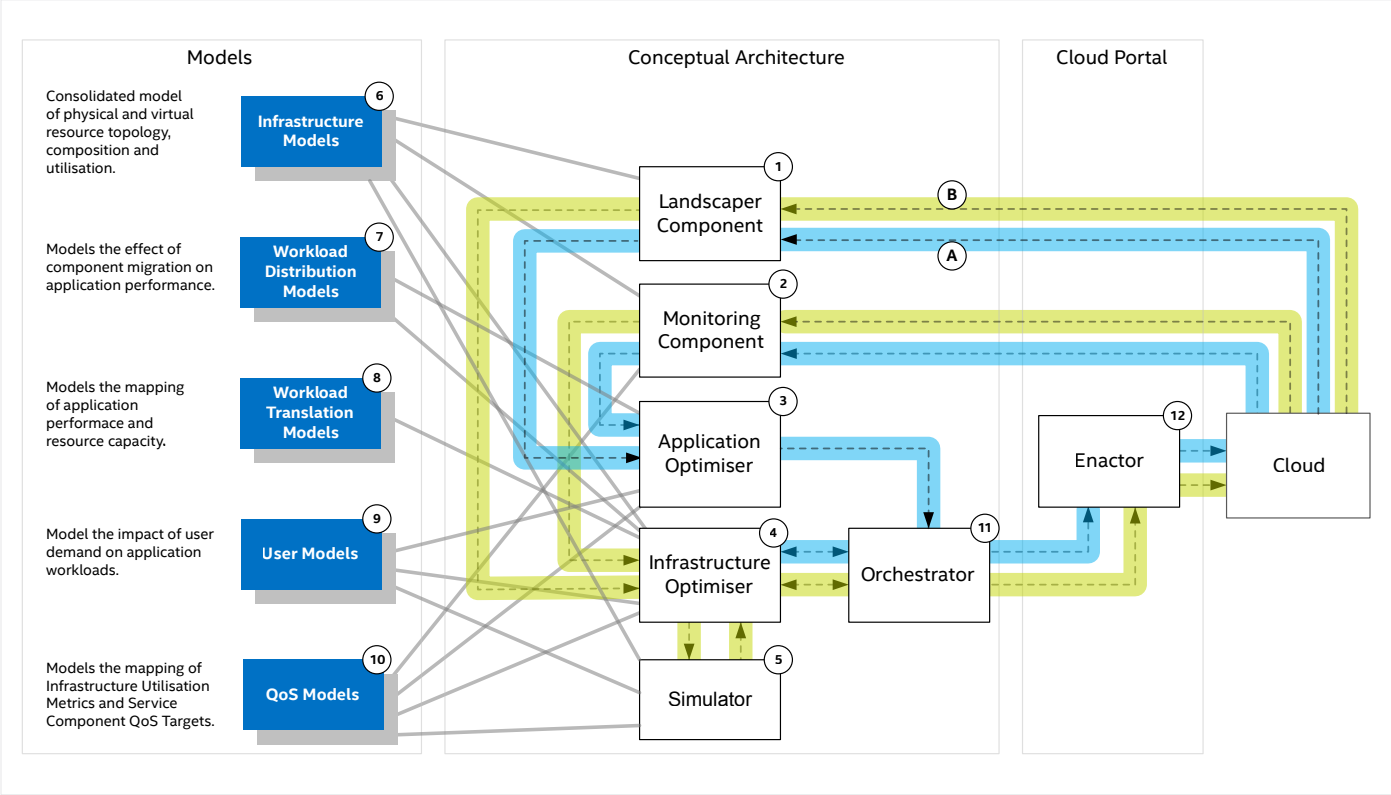


Figure 1. RECAP Optimisation Framework

Having collected application optimisation requests and recommendations for placement from multiple Application Optimisers, the **Orchestrator (11)** merges and transforms them into input for the **Infrastructure Optimiser (4)**. The **Infrastructure Optimiser (4)** ingests and augments these initial placement decisions by considering additional and more granular information pertaining to the available physical infrastructure, infrastructure specific features, infrastructure policies and service level agreements (SLAs). The **Infrastructure Optimiser (4)** output is then fed to the **Orchestrator (11)**, which transforms it into enactment instructions for the **Enactor (12)**. This allows the **Infrastructure Optimiser (4)** to retain effective control of the infrastructure resources. The **Simulator (5)** is utilised by the **Infrastructure Optimiser (4)**, in a human-in-the-loop fashion, to formulate deployment mapping selections and calibrate its algorithmic process. **The Simulator (5)** validates the results of the optimisation and assists with 'what-if' scenario planning.

The optimisation process flow can be considered as two sequential optimisation loops:

Process Flow A: The Application Optimiser is fed with output of the **Landscaper (1)** and **Monitoring (2) Components**, which represents the current resource capacity and utilisation, as well as the **Workload Distribution Models (7)**, **User Models (9)** and **QoS Models (10)**, which represent the application workload and performance targets. The **Application Optimiser's (3)** prediction engine produces a recommended deployment of components and outputs this to the **Orchestrator (11)** and subsequently through the **Infrastructure Optimiser (4)** for evaluation and orchestration. The **Application Optimiser (3)** can be subsequently triggered dynamically to handle variations in application workloads and user behaviours so that placement and autoscaling can take place. In its most proactive mode, the **Application Optimiser (3)** can create virtual resources, placing and autoscaling based on machine-learning models that are run against workload and user metrics in real time.

Process Flow B: The **Infrastructure Optimiser (4)** uses the more granular output of the **Landscaper (1)** and **Monitoring (2) Components**, which represent the current resource capacity and utilisation, as well as the **Infrastructure Models (6)**, **Workload Distribution Models (7)**, and **Workload Translation Models (8)** to optimise the utilisation of the physical hardware resources based on required service level targets and policies.

6. Case Study: Optimal Deployment of an NFV-LTE Network on an Urban Edge Infrastructure

The use case in this study focused on the optimal deployment of an LTE network service supporting multiple simultaneous user services, including web browsing, email, instant messaging, music streaming, voice call and video call. Simulators were used to simulate the user request workload from user devices as well as the resulting downloaded service traffic to the network. The user traffic profile had inherent growth trends and user mobility behaviour.

6.1 Case Site

Figure 2 represents the logical cloud network structure onto which the aforementioned application components were deployed for the workloads under test. The network is a hierarchical network with four layers as shown. The lower tier is the access or edge layer with aggregation and core tiers acting as connectivity transit points and nodes of greater resource intensity. This facilitates network management and policy enforcement as well as network resilience. The network comprises 45 dual-parented nodes, with nodes in the core tiers being ring-connected. The physical compute and networking capacity is pre-defined for each node depending on its tier level. The network capacity on the links between nodes is also pre-defined based on the layers of the end nodes. The link latencies were calculated based on the geographical topology distances, fibre propagation and routing and switching delays. Although the latencies are found indirectly, the telemetry provides utilisation statistics for links and nodes.

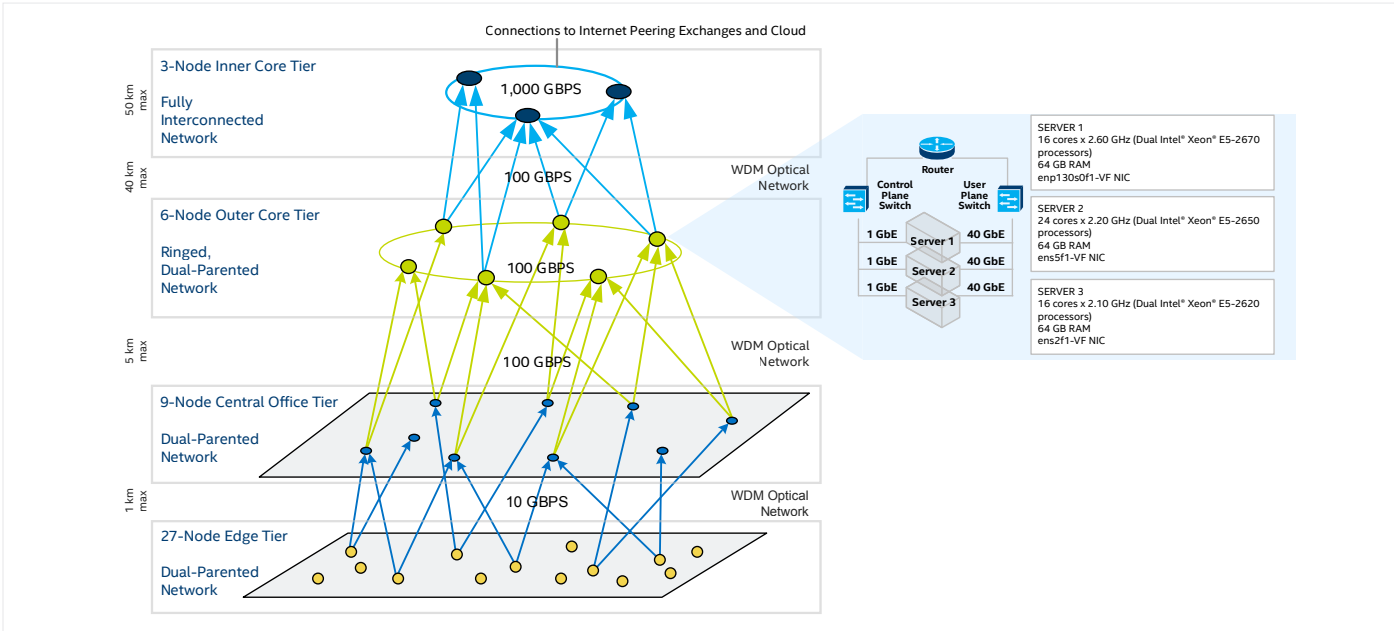


Figure 2. Physical Infrastructure/Tiered Topology

6.2 Problem Formulation

The schema below represents the functional components and dependencies of an LTE mobile infrastructure and the inherent functional division between the user plane and control plane that is part of the VNF architectural design of LTE and 5G networks.

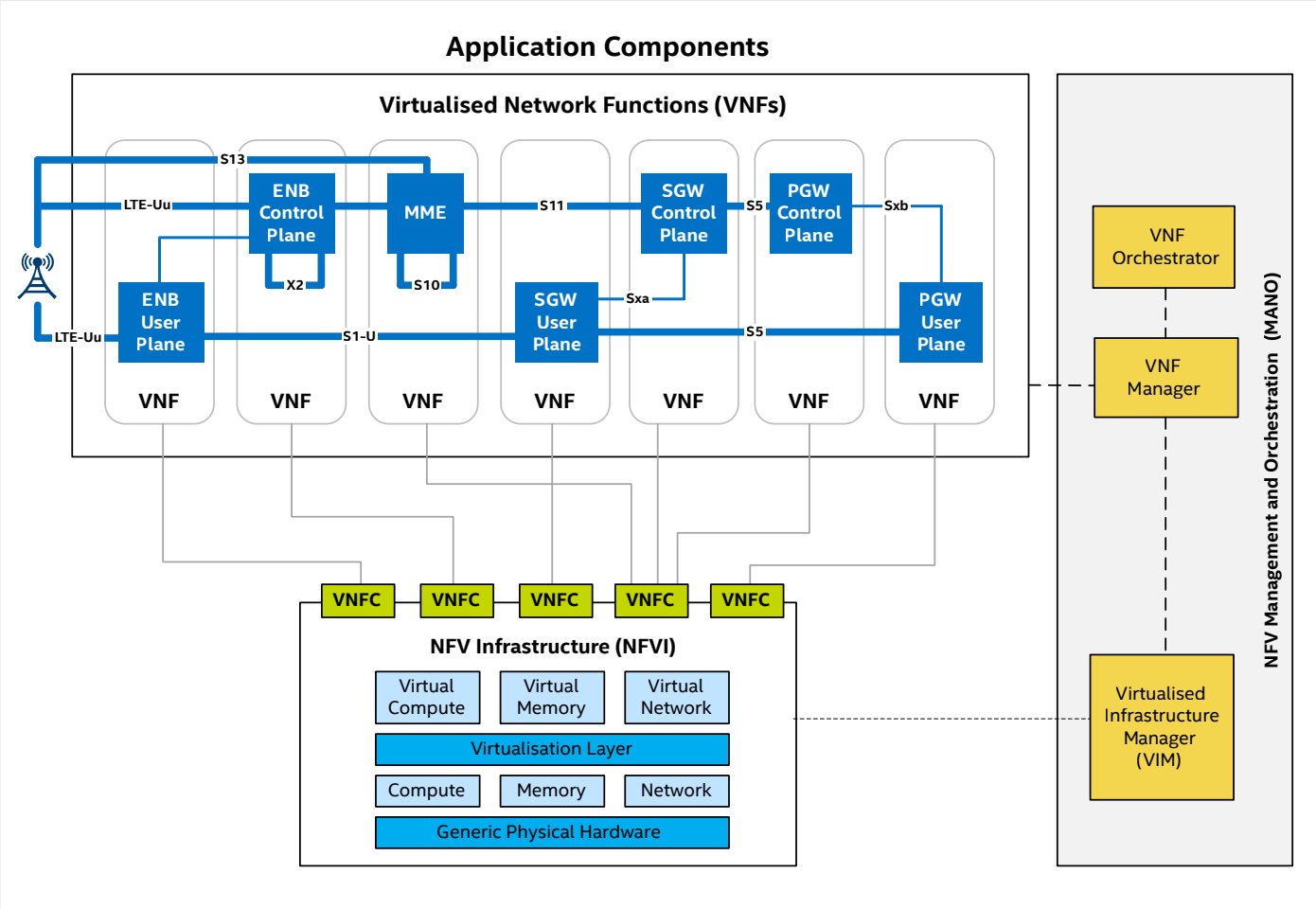


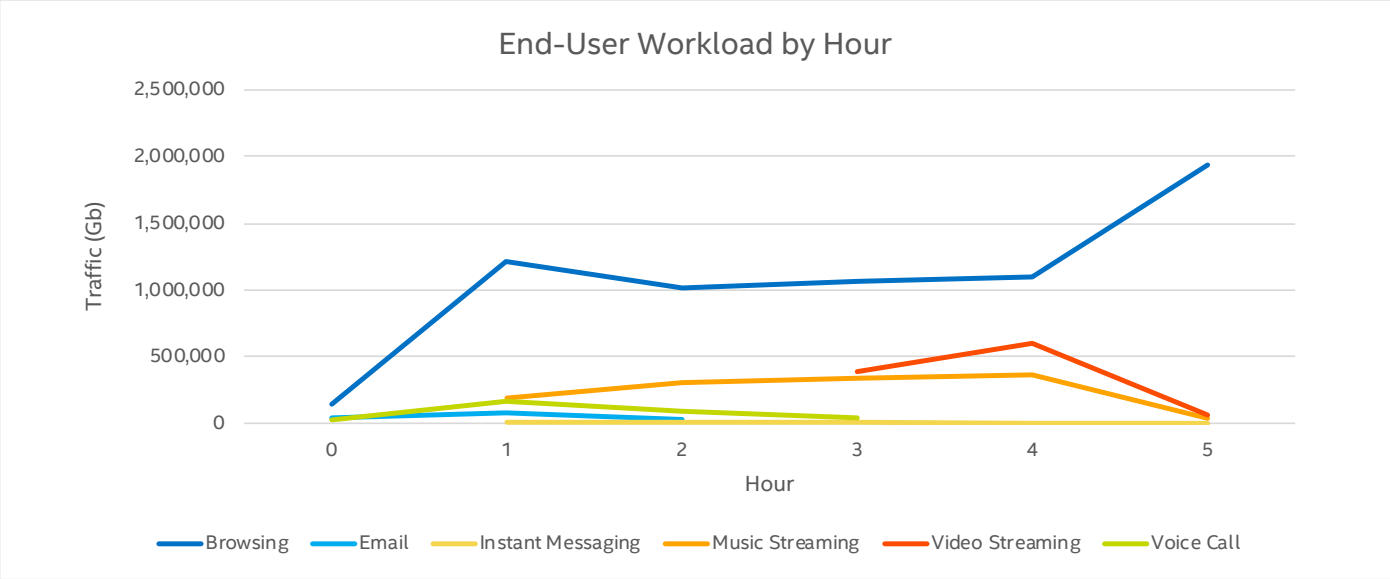
Figure 3. Logical VNF Application Structure

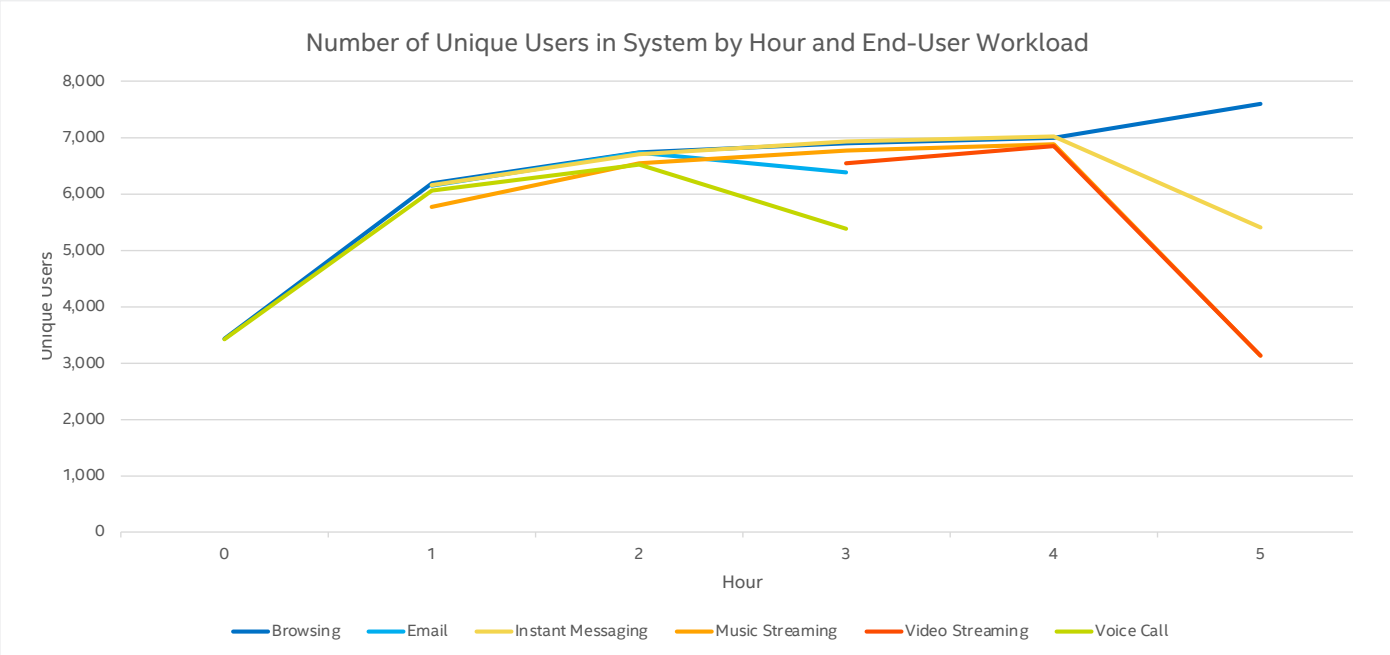
Essentially, the various components have different resource needs and constraints while the underpinning infrastructure will be the same. For example, the mobility management entity (MME) may need more compute capacity, and the serving gateway (SGW) may need more networking capacity. Each VNF may run on one or more VMs. The VNFs are logically separated so no VNF will interfere with or use the physical resources of the others. Different placement scenarios involve different resultant network links between components, which introduce latency constraints. Some components are more compute hungry, while some are more network sensitive

The challenge is, therefore, given each of the VNF components maps to an individual VM, how does one optimally place VNF components on the available VMs while meeting the application providers/service chains QoS, and whilst optimising the underlining infrastructure capacity in the network and the interests of the infrastructure provider.

6.3 Analysis and Results[4]

Multiple simultaneous services were implemented on a testbed in Sweden designed and operated by Tieto and augmented with simulation. As discussed, several services where instantiated, including web browsing, email, instant messaging, music streaming, voice call and video call services. Simulators were used to simulate the user request workload from user devices as well as the resulting downloaded service traffic to the network. The user traffic profile had inherent growth trends and user mobility behaviour. User models based on data from the city of Umea in Sweden were used to estimate statistical arrival patterns, which were then fed to the workload models and simulator as described above. The tables and graphs in Figure 4 provide a summary of the user and workload volumes related to the test case.





USERS	229,728
CELLS	388
BASE STATIONS	131
NETWORK NODES	45

REQUEST TYPE	REQUESTS PER USER PER HOUR (AVERAGE)	REQUEST SIZE IN BITS (AVERAGE)
USER DOWNLOAD	2,808	13,927
USER UPLOAD	224	8,572
CONTROL	6	219

Figure 4. Test Case User and Workload Volumes[4]

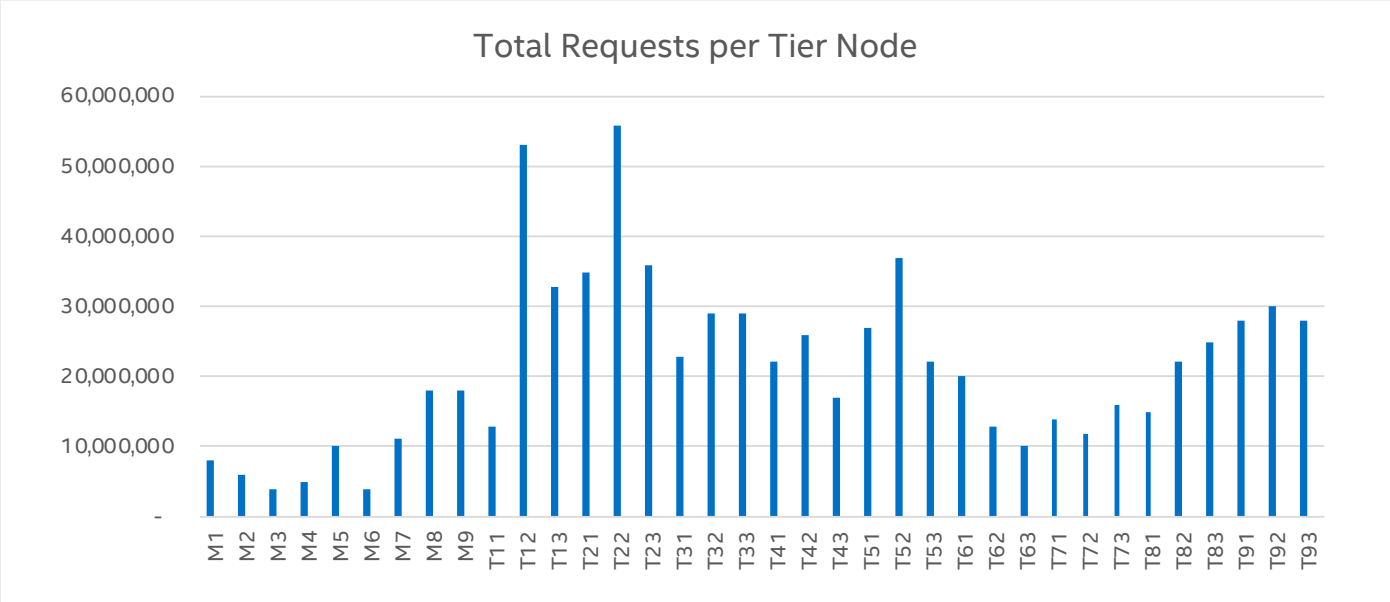


Figure 5. Total Requests per Tier Node[4]

Figure 5 shows the total service requests received at each entry node, which include the edge and central office nodes in the network.

The load translation modelling for LTE involved the definition of VNFs for the logical LTE components and their placement on to the physical infrastructure. VNF optimal placement strategies were used to deploy the VNFs as OpenStack compute instances.

The Infrastructure Optimiser selected an application placement that was optimal/sub-optimal in terms of provider and customer/user objectives (Loomba et al. 2018), as outlined. The user objective metric was the minimum bandwidth used and minimum latency; the provider objective metric was the minimum resource utilisation and minimum deployment and maintenance costs. The placements were grouped in accordance with distribution policies to reduce the problem space. Each policy led to a pre-set placing of VNFs onto a particular tier in the network hierarchy. Table 1 summarises the placement policies adopted for the test case. The RECAP Simulator was used to simulate the network infrastructure and the network traffic demands related to the different VNF placements. Multiple placement combinations were evaluated, and the resultant physical infrastructure consumption was graphed and analysed.

PLACEMENT POLICY	EDGE	CENTRAL OFFICE	CORE
1	ENB-U, ENB-C, SGW-U, PGW-U		MME/SGW-C/PGW-C
2		ENB-U, ENB-C, SGW-U, PGW-U	MME/SGW-C/PGW-C
3	ENB-U, ENB-C	SGW-U, PGW-U	MME/SGW-C/PGW-C
4	ENB-U, ENB-C		SGW-U, PGW-U, MME/SGW-C/PGW-C
5	ENB-U	ENB-C, SGW-U	PGW-U, MME/SGW-C/PGW-C

Table 1. Initial placement plans of VNFs

The output from the Infrastructure Optimiser are visualised in Figure 6 for the five placement policies described in Table 1. The bars in the bar chart represent the normalised maximum objective of the provider and customer/user for all the placements analysed in the algorithmic process.

The values on the chart represent a normalised total of the peak values of the provider and user optimality metrics combined, namely, minimum bandwidth and latency, and minimum resource utilisation and maintenance costs.

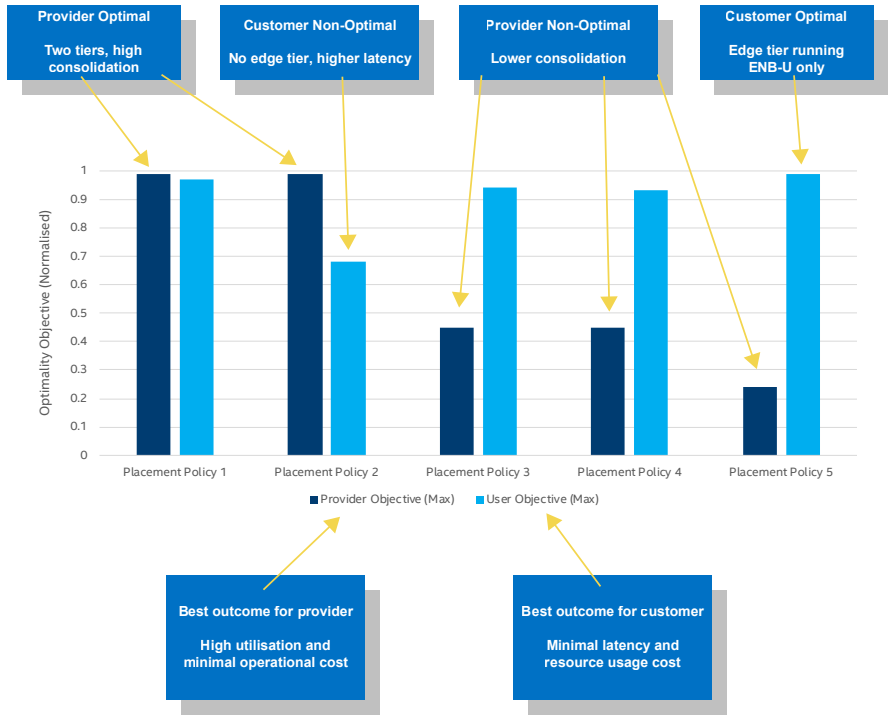


Figure 6. Infrastructure optimisation outputs (maximum provider and user utility of each placement policy)[4]

Further analysis of the results shows that it is possible to set minimum and maximum bounds for both these optimality figures and run the optimiser to output the optimal placement or sub-set of placements accordingly. Figure 7 maps, for one service chain, the normalised provider and normalised user utility of each VNF placement; it suggests that a provider can manage their deployments by fixing the provider utility or user utility in a way that balances business considerations. For example, provider utility is centred on 50% to ensure user utility is centred on 75%. The intersection of threshold lines, the highlighted section in grey, identifies a set of placements that are optimal for each individual forwarding graph of the use case whilst satisfying defined constraints, including the application and infrastructure provider perspectives. The provider could choose Distribution 1, 2 or 4. However, Distribution 2 has poorer user utility (no edge infrastructure, higher latency) and so is disregarded. Distributions 1 and 4 utilise edge and core infrastructure and have comparable user utility. However, Distribution 1 has the higher provider utility, so would be the best option. That is, it would be the best option if consolidation was the most important factor to the business, but not necessarily the best option if flexibility to service future requests was more important. In the latter case, Distribution 4 is a better option because one's current customer is happy (comparable to Distribution 1) but the provider has significant up-swing capacity.

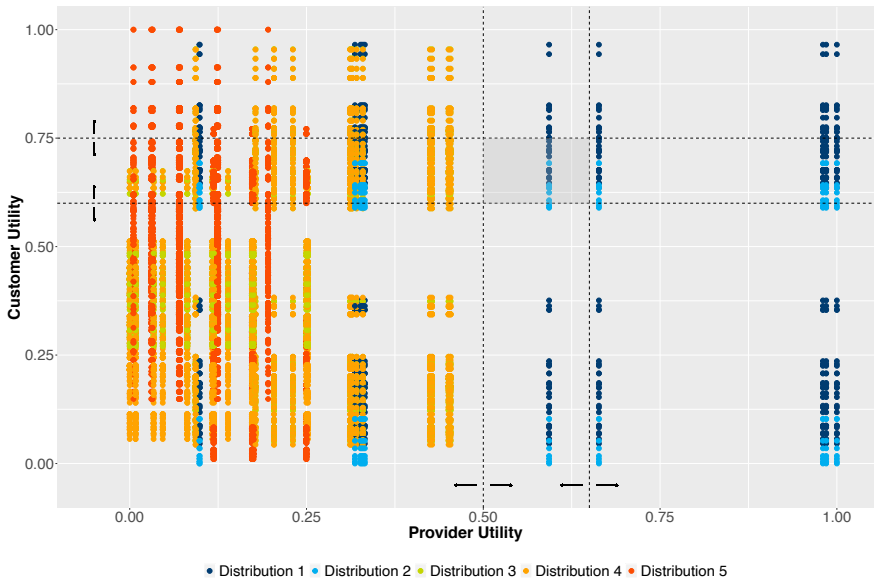


Figure 7. Provider utility vs. user utility for different distributions for one service chain[4]

Figure 8 at right illustrates simulation results for the same placement options, but for multiple service chains, excluding placement Distribution 2 as it had no edge infrastructure capacity and does not meet all SFC requirements. The graph is for all infrastructure and all remaining placement distribution options. Utility is combined for provider and user (y-axis) and is graphed against three scenarios (x-axis): normal, event and 24% growth. For the same scenarios and constraints as in Table 1 above, Distributions 1 and 4 remain the best options. As can be seen clearly from Figure 8, Distribution 1 remains the best option, with the simulation suggesting it could cope with the defined event and growth scenarios. But what is a little less obvious is that its utility remained essentially static, while the utility for Distribution 4 starts to trend upwards from normal to event to 24% growth scenario. This is primarily driven by improvements to provider utility as utilisation of physical assets improve. But this scenario offers considerably greater capacity for future growth/ event scenarios.

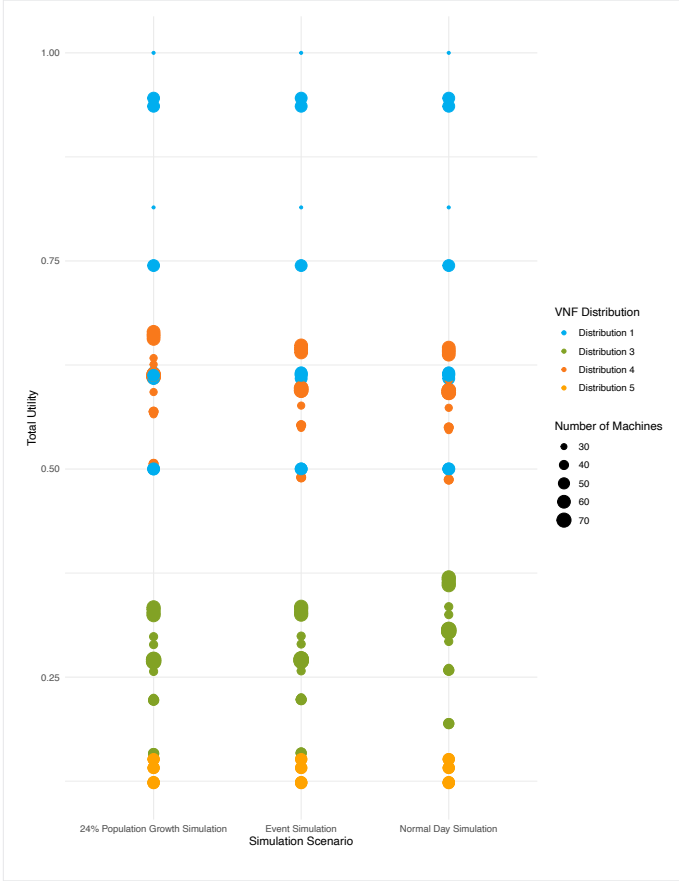


Figure 8. Total utility for normal, event and growth scenarios for all service chain placements[4]

7. Conclusion

The study shows how rich platform telemetry, combined with modelling, machine learning, simulation and algorithmic approaches can be used for infrastructure planning and automation of optimised application placement on distributed edge networks. In particular, current results contribute to more effective decision making for infrastructural resource dimensioning and planning for LTE and future 5G communication systems. The results show how the proposed RECAP architecture can be used to optimise customer QoS and QoE while at the same time minimising infrastructure provider CAPEX and OPEX.

Appendix 1: References

Kleppe A., J. Warmer and W. Bast (2003) 'MDA Explained: The Model Driven Architecture Practice and Promise.' AddisonWesley Professional, London.

Iorga, Michaela, Larry Feldman, Robert Barton, Michael J. Martin, Nedim S. Goren, and Charif Mahmoudi. 'Fog Computing Conceptual Model' (NIST, March 2018), https://www.nist.gov/publications/fog-computing-conceptual-model.

Loomba, R., T. Metsch, L. Feehan and J. Butler, 'A Hybrid Fitness-Utility Algorithm for Improved Service Chain Placement,' 2018 IEEE Global Communications Conference (GLOBECOM), pp. 1-7. doi: 10.1109/GLOCOM.2018.8648033 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8648033&isnumber=8647127

Mell, Peter and Tim Grance. 'The NIST Definition of Cloud Computing' (NIST, September 2011), https://csrc.nist.gov/publications/detail/sp/800-145/final.

Learn More

For more information about RECAP, full partner list and additional use cases, visit recap-project.eu.

Find out more about 5G at intel.com/5G.
Read an article on 5G network transformation.

For more information about Tieto, visit tieto.com/pds.

For more information about BT, visit bt.com.

For more information about DCU, visit www.dcu.ie.

Appendix 2: Terminology

ABBREVIATION	DESCRIPTION
5G	Fifth Generation
API	Application Programming Interface
BT	British Telecommunications Plc
CPU	Central Processing Unit
DCU	Dublin City University
ENB-C	Enhanced Node B – Control plane
ENB-U	Enhanced Node B – User plane
GB	Gigabytes
GbE	Gigabit Ethernet
GBPS	Gigabytes per Second
GHz	GigaHertz
IOT	Internet of Things
KM	Kilometre
KPI	Key Performance Indicator
LTE	Long Term Evolution
ML	Machine Learning
MME	Mobility Management Entity
NFV	Network Function Virtualisation
NIC	Network Interface Controller
NIST	National Institute of Standards and Technology
OTT	Over-the-Top
PGW – C	Packet Data Network Gateway – Control plane
PGW – U	Packet Data Network Gateway – User plane
QoE	Quality of Experience
QoS	Quality of Service
RAM	Random Access Memory
SDI	Software Defined Infrastructure
SDN	Software Defined Networking
SFC	Service Function Chain
SGW - C	Serving Gateway – Control plane
SGW - U	Serving Gateway – User plane
SLA	Service Level Agreement
SR-IOV	Single Root Input/Output Virtualisation
TCO	Total Cost of Ownership
VM	Virtual Machine
VNF	Virtual Network Function
vNIC	Virtual Network Interface Controller
WDM	Wave Division Multiplexing

RECAP HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON 2020 RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT NUMBER 732667.

[1] The National Institute of Standards and Technology (NIST) defines cloud computing as a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction (Mell and Grance 2011, p. 2).

[2] NIST defines fog as a computing paradigm based on storing and processing data on infrastructure closer to the user device (Iorga, Feldman, et al. 2018).

[3] With edge computing pertaining more specifically to the network layer encompassing the peripheral end-devices. Drivers for edge include reduced latency; physical constraints related to regulatory compliance, security, privacy; reduction in bandwidth consumption; service continuity in the event of internet connectivity failure.

[4] Testing conducted by Tieto, Dublin City University, and Intel in October 2019. Server hardware configurations: Compute-1: 2x Intel® Xeon® E5-2670 CPUs at 2.60 GHz (microcode: 0x70d), Intel® Hyper-Threading Technology (Intel® HT Technology): off, BIOS: American Megatrends Inc. v. 5004, Memory: 64 GB at 1,333 MHz; Compute-2: 2x Intel Xeon E5-2650 v4 at 2.20 GHz (microcode: 0xb00001b), Intel HT Technology: off, BIOS: American Megatrends Inc. v. 3304, Memory: 64 GB at 2,400 MHz; Compute-3: 2x Intel Xeon E5-2620 v4 at 2.10 GHz (microcode: 0xb00001b), Intel HT Technology: on, BIOS: American Megatrends Inc. v. 3304, Memory: 64 GB at 2,400 MHz. Server software configurations: Ubuntu 14.04.5 LTS, kernel 3.13.0-129-generic used on all three servers.