Service-centric networking for distributed heterogeneous clouds

Pieter Simoens, David Griffin, Elisa Maini, T. Khoa Phan, Miguel Rio, Luc Vermoesen, Frederik Vandeputte, Folker Schamel, Dariusz Burstzynowski

***Abstract*—Optimal placement and selection of service instances in a distributed heterogeneous cloud is a complex trade-off between application requirements and resource capabilities that requires detailed information on the service, infrastructure constraints and the underlying IP network. In this article we first posit that from an analysis of a snapshot of today's centralized and regional data centre infrastructure, there is a sufficient number of candidate sites for deploying many services while meeting latency and bandwidth constraints. We then provide quantitative arguments why both network and hardware performance needs to be taken into account when selecting candidate sites to deploy a given service. Lastly, we propose a novel architectural solution for service-centric networking. The resulting system exploits the availability of fine-grained execution nodes across the Internet and uses knowledge of available computational and network resources for deploying, replicating and selecting instances to optimize Quality of Experience for a wide range of services.**

I. INTERACTIVE DEMANDING SERVICES IN THE CLOUD

There is vast diversity in cloud-hosted services today, ranging from mobile back-ends, over virtualized set-top boxes and gaming consoles to real-time services providing decision and control support for self-driving cars. These recent cloud services require a crisp experience and/or real-time processing of high data rate streams. High network delays and low throughput to a relatively small number of centralised remote data centres (DCs) may have a serious impact on the quality of experience (QoE). For instance, 30% of the US population has a too high latency to one of Amazon's EC2 DCs for cloud-based gaming [1]. Deploying such applications in distributed execution platforms closer to the users reduces network delays and is also the preferred approach for many data intensive applications. Shifting all the data to a centralised service could overwhelm the network and it is better to bring the computation logic closer to data sources and users at the network edge. As of today, Internet Service Providers (ISPs) already deploy Content Delivery Network (CDN) proxy servers in their network to save on transit costs and improve the quality of service for their customers [2].

Service developers are thus confronted with the twofold challenge of service instance placement and selection. The central problem in service placement is to determine the cost-optimal set of geo-distributed datacenters where to deploy an instance, and to configure the appropriate scaling policies in each of these datacenters to adequately cope with the expected demand. These distributed nodes have heterogeneous hardware, as they are owned by different entities or deployed at different moments in time. Service instance selection refers to the anycast-style resolution of a service identifier to the network endpoint of the best replica, taking into account service availability, network metrics and the location of the requesting user.

Service placement and instance selection in distributed clouds are best performed on the grounds of **both** network and service performance metrics. This knowledge is however distributed among different business entities in the value chain of application delivery, such as infrastructure providers, ISPs and service developers, and is highly impacted by the specific service requirements as well as the characteristics of the underlying heterogeneous cloud infrastructure. Misaligned objectives and incomplete visibility on policies due to IPR protection mechanisms can lead to suboptimal decisions in terms of service performance and deployment cost [3].

In this article, we introduce the concept of service-centric networking (SCN) as a framework that holistically addresses both service and network aspects when providing functionality for service resolution and placement in a distributed and heterogeneous cloud environment.

The remainder of this article is structured as follows. First we discuss existing frameworks enabling collaboration between ISPs and service providers and for distributed service management. We then zoom in on the need for close cooperation with the ISP in selecting service instances based on performance and bandwidth/cost grounds, as well as on the importance of DC capabilities being part of the service placement optimization problem. In the last part of the paper, we introduce the SCN architecture and its primitives for capability and performance awareness.

II. RELATED CONCEPTS

CDNs cache content closer to the user to reduce traffic in interconnection links, and to provide higher downloading speed and lower access delays. CDN typically uses Domain Name System-based resolution to select the appropriate server. End-user mislocations and the limited view of network bottlenecks have been major drivers for CDN-ISP collaboration to improve server selection and enable on-demand negotiation of CDN surrogates on ISP-owned

This work was supported in part by the European Commission through the FUSION project under grant agreement no. 318205".

P. Simoens is with Ghent University – iMinds, Belgium (e-mail: psimoens@intec.ugent.be).

D. Griffin, E. Maini, T. Khoa Phan and M. Rio are with University College London, UK.

D. Bursztynowski is with Orange Polska Labs, Poland and Warsaw University of of Technology, Poland.

F. Vandeputte and L. Vermoesen are with Nokia Bell Labs, Belgium.

F. Schamel is with Spinor GmbH, Germany.

datacenters [2]. CDNs are often combined with Application Delivery Networks (ADN) consisting of controllers deployed in datacenters that reduce the service load through load balancing or performing application accelerations like image transcoding or SSL offload. ADN middleboxes are over-the-top (OTT) proprietary solutions that optimize the service load, but they are black boxes to the ISP. Only the largest enterprises can carry the extensive costs of operating a private WAN that connects geo-distributed datacenters and peers with user ISPs [4].

CDNs and ADNs provide partial solutions to the targeted problems by SCN. CDNs choose between cached content replicas for lower network delays, while SCN also accounts for service-level performance information and service availability. SCN fills the gaps in network-wide service orchestration and introduces service resolution to provide intersection with traffic engineering in transport network and data centres.

Existing research on service resource allocation in geo-distributed clouds can be broadly categorized in approaches that place services in order to minimize latency [5], and approaches that instead focus on (re)placing service instances driven by variations in demand and infrastructure cost [6, 7]. The SCN primitives also account for ISP traffic optimization, service-specific performance metrics and cloud heterogeneity.

Several distributed service management architectures have been proposed. IRMOS [8] relies on strict QoS guarantees between service components so it fits best to managed networks and needs adoptions for wide area Internet. NGSON is an IEEE standardized overlay framework [9] that provides the means to flexibly interconnect existing deployed services but does not account for service placement and provisioning, scaling and heterogeneous virtualized capabilities.

While the integration of CDNs, ADNs, NGSON and other known solutions is possible at a conceptual level, it is hard to just take existing technologies in order to achieve the goals of SCN. The most important missing parts are network-wide service orchestration and support for the implementation and propagation of network policies to allow service resolution taking account of server load, DC resources and network costs and conditions. The SCN approach is holistic in addressing these problems, and provides additional functionalities oriented to recent evolutions in cloud hardware heterogeneity and lightweight virtualization.

III. Latency to Distributed DCs

It is often claimed data processing capabilities located at the extreme network edge are required to provide low-latency services. The realization of this edge computing paradigm obviously entails significant capital and operating expenses to ISPs. However, our studies show that the already existing DCs may provide sufficient performance to deliver many high-performance applications, such as cloud gaming, to the vast majority of users worldwide.

We calculated the haversine distance from all cities worldwide listed in the geonames.org database to the address of 3116 DCs identified at www.datacentermap.com. Figure 1 (a) shows the CCDF of the number of DCs within radii of 100, 500 and 2000km for all users. Network latency, in terms of round-trip-time, can be estimated from haversine distance using a conversion factor of approximately 55km/ms, as determined by the analysis of global Internet traffic [10]. This conversion factor accounts for queuing delays in intermediate switches and routers. Our model shows that 100% of users can reach at least one DC within ~36ms (2000km) and ~65% of all users can reach a DC within ~2ms (100km). It should be noted that this model assumes the best case for access network latency, for higher latency access networks, the RTT figures should be increased accordingly.

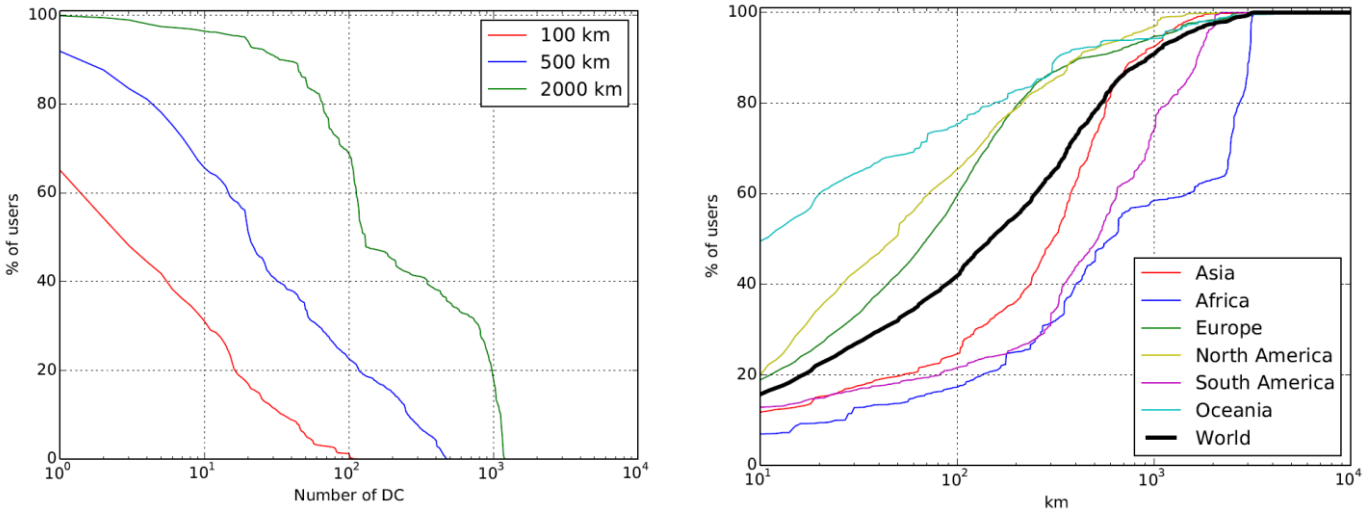
Figure 1(b) shows the CDF of the 5th closest DC to all users worldwide and per continent. This indicates that for 90% of users there is a choice of five or more DCs within 1000km (~15ms RTT) for provisioning services.

For 5T tactile services with a response time of 1ms or less [11], the existing DCs may indeed not be sufficient and additional micro-DCs within ISP-provided locations may be required to keep latency below 10ms. On the other hand latency-tolerant services, such as document editing, can be deployed in a handful of centralized locations. However, even for latency-tolerant services it might be appropriate to deploy replicas in more locations, especially when they are bandwidth-hungry, such as remote video processing or large-scale data analysis. A distributed deployment closer to users and data sources can drastically reduce bandwidth costs.

For the majority of applications that lie between these two extremes and require a response within 30-100ms, including audio-visual applications such as video conferencing and cloud gaming a deployment in a number of the existing DCs is sufficient to meet performance requirements. Service placement optimization is required in order to select the minimum number of locations to run services, and hence reduce cost, while ensuring that the selected DCs are within tolerable performance limits. Besides network metrics, also the infrastructural aspects of the DCs impact the service placement. We will discuss these in section V, but we will first study the added value of the ISPs knowledge on network metrics in placement and resolution.

IV. Network-Aware Service Placement and Resolution

Commercial solutions like Cedexis or CloudHarmony provide benchmarks of CDNs and cloud providers worldwide on end-to-end network metrics such as latency, jitter and throughput. Statistics are crowdsourced in an over-the-top manner, by clients accessing HTTP pages with embedded scripts to measure network statistics to selected sites. The accuracy and timeliness of these datasets depends directly on the number of participating clients. ISPs on the other hand have a detailed insight in the performance of their own network, and on the BGP routing topology towards other Autonomous Systems (AS). This inter-AS routing is subject to changes (e.g. due to link failures) and traffic routing policies. A key question is thus whether OTT measurement methods are sufficient for taking resolution decisions or whether this role is better assumed by the ISP.



(a) CCDF of number of DCs available within radii of 100, 500 and 2000 km for all users worldwide.

(b) CDF of the distance of the 5th closest DC for all users, split by continent and for the global population

Figure 1 Characterization of the geographical distance between users and DCs worldwide

We measured every 6 minutes the RTT to 209 DCs worldwide from the Orange Poland network in the period Jan 8 - Feb 8 , 2016. Each measurement consisted of downloading 12 times a Javascript that only contains an empty method, and taking the average of only the last 10 downloads to exclude warming-up effects.

We correlated these application-layer latency results to the directly observed changes in BGP inter-domain routing by the ISP. Figure 2(a) visualizes the impact of a link failure between the Orange Poland network and a Tier-1 network on the end-to-end delay between our probe and a subset of the DCs.

Link failures introduce a storm of BGP updates. After convergence of the BGP rerouting, the RTT of about 10% of monitored sites located in Europe and other continents (for the sake of visibility, only a subset are included in the figure) stabilizes on a new value. Although for most DCs the latency observed after BGP convergence does not differ noticeably from before the failure, there is still impact in terms of lost connectivity: the gaps in the figure correspond to failed measurements during the connectivity downtime.

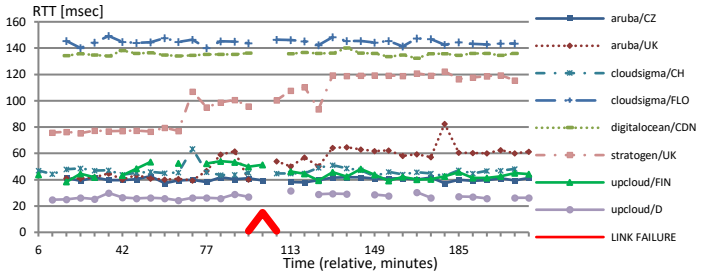
The period of broken connectivity extends for several minutes, which can have a negative impact on the QoE. Such interruptions can only be detected by OTT probes if measurements are taken very frequently and there are sufficient users in each AS crowdsourcing data. Real-time monitoring of BGP route updates is therefore a more scalable and practical proposition to detect interruptions quickly and to increase the responsiveness to changes in network conditions.

The next question is then how often such BGP route updates occur over time, and how much of the forwarding entries in the routing table are affected. Figure 2(b) provides insight into the scale of this phenomenon. The dashed plot describes the total number of route updates (forwarding entry changes) during the observation period (one month) such that the time elapsed from the previous update for a given prefix was not less than a given value. We note every such “active” prefix involves a set of IP addresses. Accordingly, the solid line shows the fraction of the IPv4 address space that correspond to the route updates described by the dashed line.

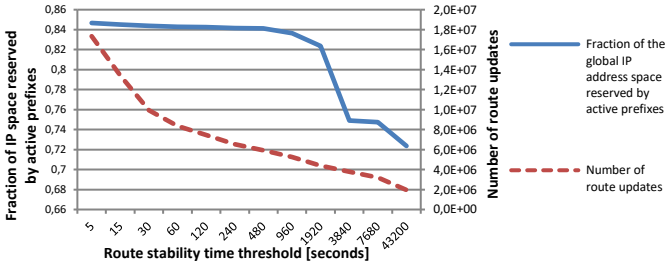
The general conclusion from this analysis is that BGP route changes are observed for a large portion of the IP address space and over a wide range of time scales, and that BGP route updates are a quick indicator of changes in network performance between end-users and DCs. Although BGP updates could in principle be monitored and processed by non-ISP third parties, this requires probes deployed in various vantage points around the globe. The quantity of information to be processed by OTT providers would easily become prohibitive: BGP route updates observed at different locations must be correlated and the impact on users from each AS must be calculated, which is a complex process considering that BGP changes in a single AS cause a high rate of globally propagated updates. Moreover, ISPs are unlikely to expose full details of their peering, transit and uplink connections with third parties, meaning that this information must be indirectly inferred by OTT parties.

In summary, if resolution decisions are made by OTT service providers they require a significant overhead in terms of network monitoring infrastructure and the result may be sub-optimal from the perspective of traffic costs of the network operators. ISPs are in a privileged position to make service resolution decisions due to the efficiency and accuracy of direct access to network performance information from the perspective of their users, with the added benefit of being able take network costs into account.

Participating in service resolution decisions has several other advantages to ISPs, in particular to reduce traffic cost. Service replicas will be located in a range of DCs and the routing paths to those in remote ASs will be over peering and transit links with different monetary costs to the ISP. The ISP is thus able to select service replicas with an appropriate trade-off between service utility and network costs to ensure QoE within acceptable traffic costs for the network operator.



(a) Impact of a BGP event on the end-to-end latency to DCs worldwide. The event was observed by the Orange Poland network on Jan 26, 2016 at 12:34:53 CET



(b) Number of route updates and the fraction of the active IP address space as a function of the minimum time between consecutive route updates, measured from Orange Poland network in the period Jan 8 – Feb 8 2016

Figure 2 Network and routing statistics from Orange Poland

V. Performance Variations In Heterogeneous Clouds

Network metrics are not the only factors to be considered in service placement. Demanding services often have specific hardware/software resource and performance requirements to deliver a consistent QoS. For example, media services may depend on certain GPU features such as specific OpenGL extensions, or vendor-specific APIs such as NVIDIA CUDA support.

However, even with identical hardware we can observe huge performance differences across DCs, owing to the configuration and management policies of the infrastructure provider. For economic reasons, infrastructure providers will co-locate many workloads on the same node, balancing resource isolation policies with resource oversubscription, thereby assuming that not all concurrently running applications need their full capacity at the same time.

To demonstrate the impact of resource isolation policies on service performance, we have measured the latency of a media encoding application for producing a single frame in a 720p video stream. Targeting a frame rate of 25 fps, this latency should be kept below 40ms. 48 application replicas were deployed on bare metal, in a VM managed by the KVM hypervisor, in a Docker container and on bare metal with NUMA-aware placement.

The CCDF plots in Figure 3 show the probability that the time to produce a single frame exceeds a given latency. The full lines report the average performance of the 48 instances, using the default best-effort settings for CPU isolation of a vanilla Linux kernel. The dashed line indicates the same metric for one instance that was configured with a higher priority class, while the other 47 were scheduled with best-effort. It can be clearly observed that the enabled Linux mechanisms result in much stronger guarantees on application performance for all tested virtualization technologies.

The type of hypervisor used and the implementation of the resource isolation mechanisms to provide strict performance guarantees may differ widely among infrastructure providers. Moreover, it is hard for infrastructure providers to come up with a single configuration that is optimal for all applications. First, server workload characteristics continuously change as application instances come and go. Second, there is a wide variety in performance bottlenecks: CPU-intensive, memory-intensive, high I/O, etc. An experimental study concluded that the best-performing configuration for an application in one cloud provider can become the worst-performing configuration for that application in another cloud [12].

Given the impact on service performance of hardware resources, infrastructure management policies and runtime conditions, it is clear that the cost-versus-quality trade-off of a DC for resource-demanding services is highly application specific. Moreover, the placement decision can only be performed in an optimal way when it is based not solely on static descriptions of DC capabilities, but involves an evaluation of the runtime condition on application-specific requirements.

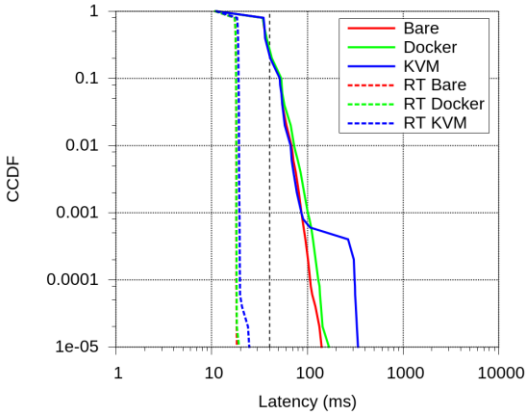


Figure 3 CCDF of the latency to produce a single 720p video frame. The experiments were conducted on a SuperMicro server blade, with a dual Opteron 6174 CPU and 64 GiB RAM. Full lines: average CCDF of 48 instances with best-effort CPU scheduling of the vanilla Linux kernel. Dashed lines: CCDF for a single instance that was attributed a higher CPU scheduling class, while the other 47 instances were scheduled best-effort.

VI. Service-Centric Networking

The previous discussion reveals that for both service placement and for service resolution, detailed knowledge is needed about the capabilities of heterogeneous nodes, the IP network topology and service performance metrics. This knowledge is scattered between different business entities, such as infrastructure providers, ISPs and service developers.

In the following we describe an intermediary *Service-Centric Networking* (SCN) framework that assists service providers to manage the deployment and operation of services over distributed heterogeneous clouds. This includes the optimal placement of service instances considering the capabilities of DCs, their proximity in terms of network metrics to user demand, dynamic service scaling to meet varying demand and the resolution of user queries to the best service instance, according to a combination of network metrics, available server capacity and other operational policies such as minimizing transit costs.

The framework is enabled by several primitives, including evaluator services, session slots and service catalogues to convey information that are abstract enough to avoid the exposure of IPR on network or service performance, yet contains sufficient detail for service placement and resolution in distributed heterogeneous cloud environments. Placement is performed on a deeper level than the limited set of regions offered by current geo-distributed DC providers, and the service-specific impact of hardware heterogeneity is taken into account when assigning resources to the deployed replicas.

A. Functional entities

The SCN framework covers service management and resolution functions implemented by multiple cooperating, but loosely coupled entities: service providers, service orchestrators, DC providers and service resolvers. The service lifecycle across these entities is depicted in Figure 4.

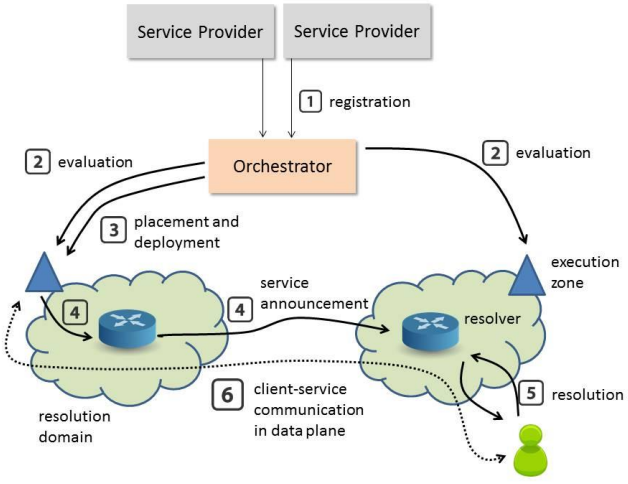


Figure 4: Service lifecycle in service-centric networking

1) Service providers register their service with an **orchestrator** via an (extended) TOSCA service manifest, containing information such as the service graph identifying service components and their relationship with one another, performance requirements and constraints, and deployment policies.

2) The orchestrator goes beyond cloud infrastructure brokering and also offers advanced instance placement, service lifecycle management and monitoring. The orchestrator carries out a detailed **evaluation** of the performance and runtime conditions of a large set of candidate execution locations, named execution zones (EZ). The computational resources may be a dedicated DC of a cloud infrastructure provider, or similar resources co-located with PoP, base stations, etc. provided by an ISP. The placement decision may be based on service-specific evaluator services, a concept further detailed in section VI.C.

3) The evaluation results are used to **deploy** service replicas in a subset of the EZs, taking into account the service requirements and policies listed in the service manifest.

4) EZs report on their service availability to the **service resolution** subsystem, which is responsible for creating dynamic forwarding paths for end-user queries to be resolved to EZs containing available instances of the requested service. Multiple domain resolvers exchange information on service availability, and each domain has a logically centralized resolver that answers queries from the domain's clients.

5) The resolver returns a **locator** of the service replica to the client. These locators can contain IPv4/IPv6 address, TCP ports, protocol numbers and/or tunnel identifiers. The location of the resolver for a specific service and/or a given user can be retrieved through standard DNS mechanisms.

6) The client then accesses the service replica over standard IP connection, **out-of-band** of the SCN framework.

B. Utility-based placement with evaluator services

Service placement involves a cost-vs-quality trade-off that is application specific. The service provider specifies in the manifest the service performance targets by means of a utility function. Utility is defined as a weighted combination of metrics relevant for the service performance and can range from zero to one. Further details on the utility function can be found in [13].

Placement algorithms in the orchestrator need to solve a multi-objective optimization problem to maximize the total utility of all users within budget constraints. We show in the Pareto frontier of the trade-off between placement cost and user utility for the EZs and user demand as described in section III.

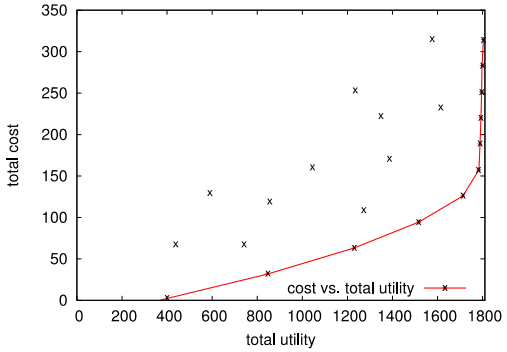


Figure 5 Pareto graph placement cost vs. utility

Costs are in arbitrary units and are proportional to the published cost of the closest Amazon EC2 for each EZ. The X-axis is the sum of utility each of 1800 user groups received by accessing services in the chosen EZ. Each point "x" on the plane represents a feasible placement solution, but only the points on the Pareto curve represents a maximum utility for a cost constraint value. Each strategy on the Pareto curve shows a particular trade-off between the utility and the cost. Based on this, the service provider can choose an appropriate operating point.

Performance impacting factors like multi-tenant resource isolation and hardware heterogeneity are only measurable at runtime and/or require in-depth and sensitive knowledge of the service implementation to assess the utility of an EZ. Describing such detailed hardware capabilities and performance dependencies in a static manifest is infeasible. Instead, we propose the concept of evaluator services. These are lightweight services deployed as probe in a selected number of EZs to verify deployment and execution requirements and predict the performance when the application would be deployed in the same environment. Before the service is deployed, the orchestrator deploys one or more evaluator service instances across the candidate EZs. An evaluator service calculates a numerical score for the execution environment. This value, together with network statistics and infrastructure costs, is used as input parameters in the utility function by the orchestrator.

The major advantage of the evaluator service concept is that orchestrators can follow the same evaluation procedure for all services. It is up to the service providers to provide the evaluator services. In the simplest case, the evaluator service only makes a small number of system API calls to verify whether a required hardware or software feature is available; in other cases, a more thorough performance evaluation may be necessary. There should however be a reasonable relation between the complexity of the evaluation and the service itself, as running a complex and time consuming evaluation for a short-lived service would introduce too much overhead.

Both the utility function and the evaluator services are described as policies in a TOSCA service manifest. TOSCA is an OASIS standard for the specification of topology and orchestration of cloud applications [14]. An example is given in Figure 6. The evaluator service needs to be executed in three regions, and the utility of an execution zone is an equally weighted sum of the end-to-end latency and the numerical score of the evaluator service.

C. Distributed resolution based on session slots

Service resolution algorithms find the "best" instance amongst possibly many replicas distributed over the Internet. Simply selecting the closest EZ for each user request or the one that maximises utility for that individual request can result in sub-optimal performance. As we show in [13] a utility-maximising service selection approach in SCN can reduce blocking and increase overall utility compared to a classical closest-based selection approach.

The exchange of service availability information consists of two distinct steps: catalogue sharing and service subscription.

Catalogue Sharing: Orchestrators deploy an agent in each EZ that announces the service ID, the utility function and a representative locator to at least one resolver. This information is further injected into the catalogue which is shared between resolvers using a DHT implementation. This information only updates when a new service is created, all service instances have been deleted or there are significant changes in network connectivity (e.g. change of traffic engineering policy). To keep full control of the load on some instances, resolvers may decide to hide the actual locators and replace them with an ALTO Provider-defined Identifier (PID). ALTO is an IETF standard for dissemination of network level information between different business entities [15]. The PID is a representative locator for e.g. a subnet or a metropolitan area that allows other resolvers to assess the potential performance of connections to instances running in that domain. Operators expose cost maps, assigning cost values (e.g. routing cost) to one-way connections between PIDs. Other resolvers can then evaluate the feasibility of service replicas exposed by one resolver, without having full knowledge of the internal network or the operator policies.

Service subscription: Based on the catalogue information, resolvers subscribe to a set of EZ. To obtain enough diversity of service availability, resolvers will contact close zones before, expanding the subscriptions to more distant zones until enough instances are found. Resolvers will start receiving updates from that EZ on the availability of the service(s) subscribed to. The availability information is conceptualized as session slots. A session slot is a unit of measurement representing how many users can be accommodated simultaneously in a given service instance, group of instances or EZ. The total number of session slots to be instantiated is decided by the orchestrator and the current number of available session slots is announced to the service resolvers to help drive the instance selection algorithms.

The resolution overlay can grow organically. In an early phase, orchestrators could act as resolvers to ensure reachability of their managed services. Over time, other parties could attach resolvers to the resolution overlay. As argued in section II, resolvers may be operated by ISPs.

```
topology_template:
  node_templates:
    my_service:
      type: tosca.nodes.Compute
      properties:
        # omitted here for brevity
      requirements:
        # other requirements omitted here for brevity
        - evaluator_service:
              node: my_evaluatorServiceFeatureA
              relationship: my_evaluator

    my_evaluatorServiceFeatureA:
      type: tosca.nodes.Compute
      # omitted here for the brevity

  policies:
     -    my_evaluator_placement_policy:
            type: my.policies.evaluator #derived from tosca.policies.Placement
             container type: region
             target_regions: [ regionA, regionB, regionC ]
             evaluator: my_evaluatorServiceFeatureA
             min_score: 250
     -    my_utility_policy:
            type: my.policies.latency_utility #derived from tosca.policies.performance
            R: 0.5*e2e_lat + 0.5*evalFeatureA

   # other resources not shown here ...
```

Figure 6 Sample TOSCA manifest. Two policies are defined, based on a geographic spreading as well as on utility. In this example, the evaluator service is also used at runtime to set the specific configuration.

VII. Conclusion

In this paper, we present a framework for optimal service placement and resolution in widely distributed heterogeneous cloud infrastructures. SCN leaves the data plane unmodified and therefore aligns with other efforts to improve service delivery, such as software defined networking to manage data flows, and 5G wireless technologies to improve wireless throughput and latency.

The SCN framework has been extensively modelled and prototyped in the FUSION project. Some of the challenges of deploying SCN, as discussed in this article, involve the definition of appropriate abstractions of service requirements and the inclusion of network and service monitoring data in placement and resolution decisions. The primitives of evaluator services, utility and session slots are able to capture the vast diversity in service requirements at an appropriate demarcation level between different business entities for orchestration and resolution. Together with these primitives, the adoption of standards such as TOSCA and ALTO ease the deployment of SCN. Deployment of SCN is also facilitated by it not requiring to be deployed as a single big-bang solution. For example, service resolution can initially be undertaken by service-specific centralised functions. For more popular services that are more widely deployed, and especially for those that require a more detailed knowledge of network performance metrics than can be provided by OTT monitoring, then the resolution function can be incrementally deployed by ISPs.

There are several areas of ongoing study including: modelling and mitigating policy mismatches between service placement and resolution when deployment and networking costs are not aligned. For extremely low-latency tactile services, additional edge computing nodes may need to be utilised to deploy service instances much closer to users. Globally centralised placement optimisation functions do not scale well at this level of detail and hierarchical placement frameworks may be needed where algorithms at lower levels in the hierarchy are able to make detailed placement decisions with local knowledge of edge nodes, user locations and network topology.

References

[1] S. Choy, et al., “The brewing storm in cloud gaming: a measurement study on cloud to end-user latency”. *Proceedings of the 11th Annual Workshop on Network and Systems Support for Games*, 2012

[2] B. Frank et al. “Pushing CDN-ISP collaboration to the limit”, *ACM SIGCOMM Computer Comm. Review*, vol. 43(3), 2013.

[3] Narayana, S., Jiang, W., Rexford, J. and Chiang, M., 2013, December. Joint server selection and routing for geo-replicated services. In *Proceedings of the 2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing* (pp. 423-428). IEEE Computer Society.

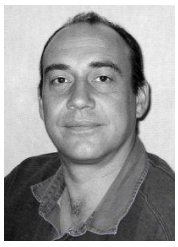
[4] Paul, S., Jain, R., Samaka, M. and Pan, J., 2014. Application delivery in multi-cloud environments using software defined networking. *Computer Networks*, *68*, pp.166-186.

[5] Malekimajd, M., Movaghar, A. and Hosseinimotlagh, S., 2015. Minimizing latency in geo-distributed clouds. *The Journal of Supercomputing*, *71*(12), pp.4423-4445.

[6] Gu, L., Zeng, D., Barnawi, A., Guo, S. and Stojmenovic, I., 2015. Optimal task placement with QoS constraints in geo-distributed data centers using DVFS. *IEEE Transactions on Computers*, *64*(7), pp.2049-2059.

[7] Zhang, Q., Zhu, Q., Zhani, M.F., Boutaba, R. and Hellerstein, J.L., 2013. Dynamic service placement in geographically distributed clouds. *IEEE Journal on Selected Areas in Communications*, *31*(12), pp.762-772.
[8] Boniface, M., et al. "Platform-as-a-service architecture for real-time quality of service management in clouds." *Internet and Web Applications and Services (ICIW), 2010 Fifth International Conference on*. IEEE, 2010.
[9] S.-I. Lee et al, "NGSON: features, state of the art, and realization", *IEEE Communications Magazine",* vol. 50(1), 2012
[10] R. Landa, et al., "The large-scale geography of internet round trip times," in IFIP Networking Conference, 2013, 2013, pp. 1–9.
[11] G. Fettweis et al., "5G: Personal mobile internet beyond what cellular did to telephony", *IEEE Comm. Mag.*, vol. 52(2), 2014.
[12] D. Jayasinghe, et al, "Variations in Performance and Scalability: An Experimental Study in IaaS Clouds Using Multi-Tier Workloads", Services Computing, IEEE Transactions on, Volume 7, Issue 2, June 2014.
[13] T. K. Phan et al. "Utility-maximizing server selection", Proc. Of the IFIP Networking Conference, 2016.
[14] OASIS, "TOSCA Simple Profile in YAML Version" http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.0/TOSCA-Simple-Profile-YAML-v1.0.html *Accessed on Sept 19,2016*
[15] RFC7285, Application-Layer Traffic Optimization Protocol

Pieter Simoens received his Ph.D. degree in 2011 from Ghent University and is now assistant professor at the same institute. His research interests include distributed real-time systems, with at a specific focus on the delivery of advanced services through distributed edge clouds. He has (co-) authored more than 70 articles in journals and conference proceedings.



David Griffin is a Principal Research Associate in the Department of Electronic and Electrical Engineering, University College London. He has a BSc from Loughborough University and a PhD from UCL, both in Electronic and Electrical Engineering. His research interests include planning, management and dynamic control for providing QoS in multiservice networks and novel routing paradigms for the future Internet.

Elisa Maini is a Research Associate in the Department of Electronic and Electrical Engineering, University College London. She received her Ph.D. in Computer and Automation Engineering from the University of Naples Federico II. Her current research interests include network optimisation and modelling, software-defined networking, and network function virtualisation.

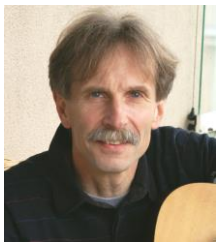
Truong Khoa Phan received his PhD degree from INRIA/I3S, Sophia, France. He is currently a Research Associate the Department of Electronic and Electrical Engineering, University College London. His research interests include network optimisation, cloud computing, multicast and P2P.

Miguel Rio is a Senior Lecturer in the Department of Electronic and Electrical Engineering, University College London where he researches and lectures on Internet technologies. His research interests include on real-time overlay streaming, network support for interactive applications, Quality of Service routing and network monitoring and measurement.

Luc Vermoesen is a research engineer in the IP Platforms Research Program in Bell Labs in Antwerp, Belgium. He graduated in engineering in 1989 and studied computer science in 1995. He joined Alcatel-Lucent back in 2000 where he worked on projects involving 3G Mobile, VDSL prototyping, Asynchronous Access Multiplexer and IP Service routing and switching. In 2007, he joined the Bell Labs Fixed Access team where he was involved in Home Networking research and contributed to the Broadband Forum standardization activities. In 2009, he started working on multimedia-related research topics like novel graphical user interfaces for IPTV and network-based rendering techniques using dedicated HW acceleration. From 2011 onwards, he is involved in cloud computing research with specific interest in virtualization and performance, as well as the applicability of heterogeneous hardware in the cloud. He currently holds over a dozen patents.

Frederik Vandeputte received his Ph.D. degree in 2008 from Ghent University and is now research engineer at Nokia Bell Labs. His research interests include software parallelization on heterogeneous architectures, heterogeneous cloud systems, network functions virtualization and performance optimization. He has (co-)authored over a dozen articles in journals and conference proceedings.

Folker Marten Schamel is founder and managing director of Spinor GmbH, provider of the Shark 3D software for creating interactive virtual worlds in the gaming, movie and broadcasting industries. He is credited for contributing to the specification of the OpenGL standard. He has a Diploma in theoretical physics with mathematics as minor.



Dariusz Bursztynowski received his Ph.D. in Telecommunications from Warsaw University of Technology in 1992. His research interests include network architecture, traffic engineering, network performance modelling and evaluation. He has been involved in a number of Orange activities related to network planning, network management, and traffic engineering. He is currently working at Orange on future network architectures in the field of naming, routing, and autonomic resource management mechanisms.