EONA: Experience-Oriented Network Architecture

Junchen Jiang†, Xi Liu*, Vyas Sekar†, Ion Stoica+◦, Hui Zhang†*
†CMU *Conviva +UC Berkeley ◦Databricks

ABSTRACT

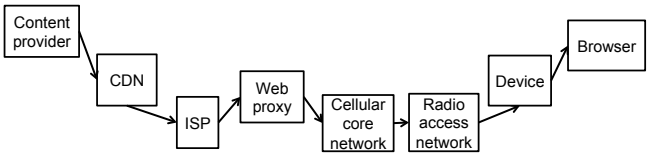
There is a growing recognition among researchers, industry practitioners, and service providers of the need to optimize user-perceived application experience. Network infrastructure owners (i.e., ISPs) have traditionally been left out of this equation, leading to repeated tussles between content providers and ISPs. In parallel, application providers have to deploy complex workarounds that reverse engineer the network's impact on application-level metrics. In this work, we make the case for EONA, a new network paradigm where application providers and network providers can collaborate meaningfully to improve application experience. We observe a confluence of technology trends that are enablers for EONA: the ability to collect large volumes of client-side application measurements, the emergence of novel "big data" platforms for real-time analytics, and new control plane capabilities for ISPs (e.g., SDN, IXPs, NFV). We highlight the challenges and opportunities in designing suitable EONA interfaces between infrastructure and application providers and EONA-enhanced control loops that leverage these interfaces to optimize user experience.

Categories and Subject Descriptors
C.2.4 [**Computer-Communication Networks**]: Distributed systems—*Distributed applications*
General Terms Design, Management, Measurement, Performance

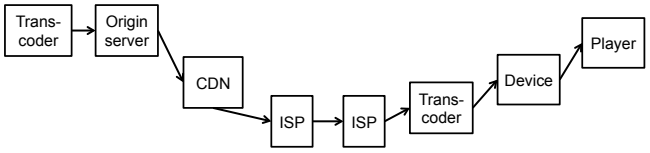
1 Introduction

The Internet is an *eyeball economy* that is driven by application experience [33, 2, 17]. There is an increasing realization of this in the networking community as evidenced by the numerous papers in recent networking conferences (e.g., [20, 36, 17, 47]), recent workshops explicitly aimed at moving up the stack and focusing human-centric experience [10, 9], and many parallel industry efforts [25, 33, 12].

Ensuring good application experience, however, is difficult in today's application delivery ecosystem with multiple independent subsystems (entities) logically owning and controlling different pieces of the delivery chain. Figure 1 illustrates the complex delivery chain for two popular use cases: video streaming and web over cellular networks. Note that each delivery ecosystem involves multiple independent subsystems spanning both application (*AppPs*) and infrastructure providers (*InfPs*). In this paper, we take a broader view of what constitutes an InfP; e.g., CDNs have been traditionally viewed as application-level overlays in the networking community but in reality they are in the middle of the delivery infrastructure today. In fact, the line between ISPs and CDNs might be blurring as ISPs enter the CDN market and with emerging content-centric architectures.



(a) Web delivery over cellular network



(b) Video streaming delivery system

Figure 1: An abstract view of web and video streaming delivery systems that consists of multiple application service providers as well as infrastructure providers. The end-to-end application quality (user experience) can be impacted by any service provider.

AppPs today rely on complex and inefficient techniques to work around the decisions of the InfPs.[1] InfPs such as ISPs have traditionally been left out of this application value chain and their control loops are largely agnostic to actual application experience. Even though recent evidence suggest efforts by InfPs to account for user experience, they do not have the necessary visibility into application performance. The status quo is a deadlock that leads to undesirable outcomes for everyone involved as each subsystem in the delivery chain undercuts or works around the other, leading to significant inefficiencies, power struggles, and fingerpointing [4, 11].

Our overarching goal is to improve user-perceived application experience because that is the fundamental driving

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
Hotnets '14, October 27–28, 2014, Los Angeles, CA, USA.
Copyright 2014 ACM 978-1-4503-3256-9 ...$15.00.

[1] We do admit there are cases where a single administrative entity serves an AppP in one context and as a InfP in another; e.g., Amazon/Google is an AppP as a web application and is a InfP w.r.t. its cloud service offerings.

force for the Internet economy. While we can envision an alternative universe where a single provider owns the entire delivery ecosystem to optimize application experience, this is fundamentally at odds with the practical reality that different entities own and control a piece of the delivery ecosystem as seen in Figure 1.

To this end, we make the case for EONA, a network architecture that enables AppPs and InfPs to work together within the realities of a federated and autonomous operational model to optimize application experience. We take the stance that both AppPs and InfPs need to react to this trend of application-experience as the driving force to remain relevant (and profitable) and that doing so results in positive synergies for both.[2] While the goals of AppPs and InfPs may not be perfectly aligned, they can both benefit from EONA-based approaches to optimize shared goals of customer satisfaction and minimizing negative effects due to independent operation without information sharing. We also recognize that the business and incentive structures for adopting EONA might be very complex as it involves entities with different business models. As such, our focus in this *position paper* is on articulating a possible technical basis for this architecture rather than advocating specific economic settlements.

In outlining the EONA vision, we observe and leverage a confluence of favorable technology trends: (1) AppPs today have extensive and widely deployed client-side instrumentation to collect application-level quality measurements [33, 20, 34]; (2) Recent network technology trends such as software-defined networking [32] and network functions virtualization [6], coupled with deployment of new exchange points[13], empower InfPs with novel control capabilities; (3) Many individual subsystems have already built or starting to build their own control plane platforms (e.g., [36]).

As shown in Figure 2, the EONA architecture envisions new *interfaces* between AppPs and InfPs. AppPs export high-level measurements of client-side application quality to collaborating InfPs (via new EONA-A2I interfaces), while InfPs export hints on their internal control decisions and network state to AppPs (via EONA-I2A interfaces). The control loops of AppPs and InfPs will now leverage the information exchanged through these new EONA interfaces to provide a tighter integration to optimize application's quality of experience (QoE).

At a high level, the idea of cross-provider information sharing is not new and we are indeed inspired by past efforts in networking research [48, 1, 24]. That said, we believe there are three distinguishing features of EONA. First, it is driven by application experience as opposed to network-level metrics (e.g., bandwidth [48]). Second, EONA envisions a two-way interface as opposed to prior work that envisioned an one-way exchange mostly from ISPs to applications [48, 1]. Third, we envision the control loops of both

[2]At least for InfPs, history suggests that if they don't respond, they run the risk of being bypassed altogether.

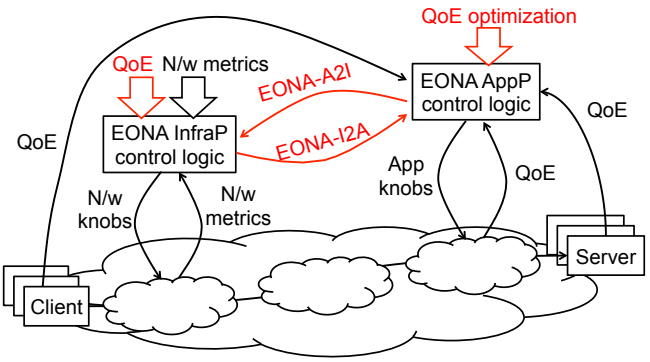


Figure 2: Architecture of EONA and its contrast to current world today. The A2I and I2A allow the InfP and AppP to exchange information that can inform each others' control loops to act in a concerted manner.

AppPs and InfPs actively working to improve application experience rather than prior efforts where the InfPs is largely passive (e.g., [24]). Finally, we note that in terms of timing EONA is better positioned and aligned with industry trends. In some sense, these prior architectures were ahead of their time and lacked either the necessary use case "pulls" (i.e., application experience as a key driver) and the technology "pushes" (e.g., SDN or client-side measurements) [23].

2 Motivating Scenarios

We begin with motivating scenarios to show how infrastructure providers (AppPs) could benefit from information from infrastructure providers (InfPs) and vice versa. These scenarios are inspired by real-world anecdotes from a large-scale application delivery optimization service.

Inefficiencies in application-level optimizations: Today, AppPs lack sufficient information about the state of the network infrastructure. Thus, they rely on complex reverse engineering, inference and diagnostics, and coarse trial-and-error processes to optimize application experience. This complicates the applications and leads to suboptimal experience.

- *Coarse control:* Today if a video player detects an issue with a particular server within a CDN, it has no choice but to switch to an alternative CDN [36, 35]. The granularity of this switch, however, is quite coarse at a *CDN granularity* and thus may disrupt experience; e.g., if the alternative CDN does not have the content in its cache yet. In this case, if the CDN can provide hints on alternative servers, the video player can reconnect to a different server and continue to play the video. By retaining the traffic the CDN can retain its share of revenue and by exploiting intra-CDN caching the application will experience less disruption.
- *Lack of visibility:* Figure 3 illustrates a flash-crowd scenario within ISP [36]. In this case, the application-level control loop (i.e., HTTP adaptive player control logic) first tried to switch across multiple CDNs but clients still see very high buffering (i.e., bad user experience). In fact, if the AppP could have known explicit congestion

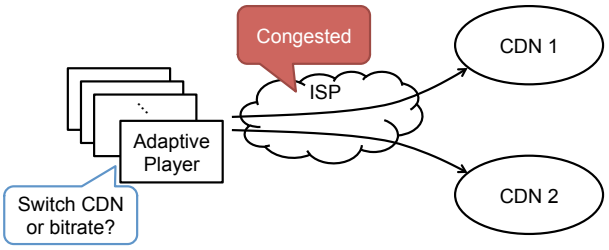


Figure 3: Example that shows lack of visibility – HTTP adaptive players switch CDNs and the access ISP is congested, while a better solution is to switch down bitrate.

signals from the ISP, it should have adapted the video bitrate to make the ISP less congested and avoid buffering.

Infrastructure providers need application visibility: InfPs today have little insights into application performance, which makes it challenging to understand the relationship between infrastructure-level controls and end-user experience. In general, application-level measurements can help InfPs understand how their infrastructure is working for different applications, how configuration changes impact user experience, and how they can better serve its customers' demands.

- *Reverse engineering application experience:* Recent work suggests that many ISPs are trying to infer application-level experience using network-level measurements as shown in Figure 4. This includes methods to understand the relationship between web experience metrics and radio network characteristics [16, 45, 14] or using coarse network-level behaviors such as time-to-first-byte in HTTP to approximate web experience [27]. While such efforts are useful, they are stop-gap solutions. InfPs can be empowered if they have direct application measurements to avoid inference, which can be inaccurate and require expensive deep inspection capabilities (e.g., [19]).

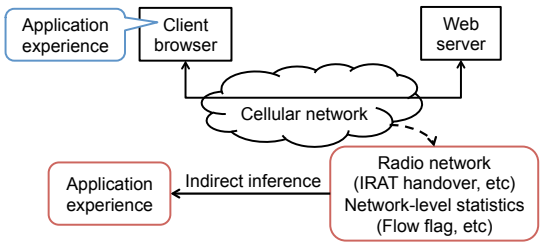


Figure 4: Cellular networks inferring application experience based on radio network characteristics or network-level behaviors, while application experience is available from clients.

- *Impacts of configuration changes:* InfPs would like to understand how configuration changes or software updates affect system performance [37, 38]. This is a common theme that has emerged in our discussions with several service providers. For example, they may want shut down some servers to save energy during off-peak hours. However, they are often too conservative or too aggressive in the decisions because they cannot observe how these decisions impact user applications.

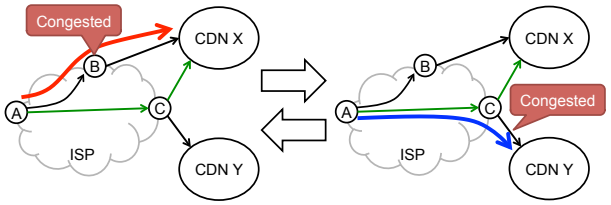


Figure 5: Oscillation can happen if control loops are independent.

Interactions between control loops: With independent control loops running by AppPs and InfPs, there can be control oscillations leading to instability. Consider the scenario of Figure 5 where an AppP has two CDNs (X and Y) to choose from, and the ISP peers with CDN X and Y at a public IXP C and it also peers with CDN X in a local peering point B. Suppose the AppP uses CDN X by default, and the ISP prefers to egress traffic for CDN X at B (the path is labeled by red line in Figure 5). In this scenario, the egress traffic increases congestion at B and results in poor user experience. Now, ISP and AppP react with ISP changing its peering point for CDN X to C, and AppP switching from CDN X to Y (blue line in Figure 5). Then, because peering point B is now not congested, the ISP changes its peering point with CDN X back to B. Meanwhile, because CDN Y does not have the capacity to handle all traffic, AppP switches CDN from Y back to X, which sets the situation back to the same as beginning, creating an (infinite) oscillating loop in both AppP and InfP. However, if the egress traffic goes to CDN X via peering point C (green line in Figure 5), there will not be any congestion, but without AppP knowing the capacity of B and ISP's current decision, this path will never be used.

3 EONA Overview

As the motivating scenarios showed, today each AppP and InfP runs separate control loops, with each having limited visibility into a piece of the complex application ecosystem. Each control logic has a set of "knobs" that it tunes to optimize some objective function. For instance, ISPs observe and optimize network-level metrics such as delay or throughput. Similarly, AppPs control the end points (clients and servers) and can observe end-to-end application experience metrics and can control the choice of server or CDN or adapt other application-level knobs (e.g., video bitrate).

As we saw in the previous section, there are several issues with such an architecture. First, InfPs have very little insight into application-level experience measures and may end up optimizing for measures that may have little, if any, impact on application experience. Second, AppPs have to use inefficient trial-and-error techniques to optimize quality, and they may hit fundamental bottlenecks as we saw in Figure 3 as they have little visibility into lower-level network states. Furthermore, there may be inherent instability/conflicts between the control loops as in Figure 5.

We could consider extreme design points that can address these issues; e.g., if the AppP owns and control every single knob in this delivery chain. Another possibility is to re-architect existing control loops; e.g., the InfP relinquishes some or all of the knobs to the AppP or vice versa. Rather than completely replace the existing control/ownership of the individual subsystems or resort to monopolistic outcomes, EONA takes an explicit stance to respect the semi-autonomous structure of the Internet application ecosystem and assumes that there will be independent entities that own, control, and monitor different pieces of the delivery ecosystem.

To enable the different providers to work in *concert* with the goal of improving application-level experience, EONA introduces new *interfaces* between the AppPs and InfPs. Note that EONA does not require changes to the data plane of AppPs and InfPs. They continue to run independent control loops and they do not need to relinquish control over their "knobs" to other providers. EONA merely adds two new *information sharing* interfaces that can better guide the control logic to improve experience-centric (or experience-friendly) objectives. For instance, InfPs and AppPs can establish "looking glass"-like servers that can be queried to implement the respective interfaces.

Using the EONA-A2I interface, we envision AppPs exporting critical application-centric experience measures collected from client-side measurements together with relevant attributes (e.g., the client ISP, and the server location). The InfPs' control loops use this information to incorporate such application experience measures into their control optimizations. As argued elsewhere, as providers seek to adopt new SDN technologies, they are still in search of suitable "use pulls" [23]. EONA's focus on application experience may offer such a concrete "use pull" to inform SDN use cases.

Similarly, InfPs export the EONA-I2A to provide additional visibility into infrastructure details. For instance, InfPs may provide hints on their peering policies or the shortest paths to CDN servers or local congestion information. Such information can better guide the AppPs' control loops that manage client- and server-side knobs.

We note that participation in EONA is optional; InfPs and AppPs can opt-in and also choose the subset of collaborators to export EONA interfaces. (We assume some suitable access control mechanism over the EONA-query servers.) We believe that the policy and economic considerations are best left to market forces and thus we do not mandate any specific economic basis or settlements and focus on the architectural enablers.

We take the stance that both InfPs and AppPs have strong incentives to adopt EONA with the goal of optimizing user experience. We already see positive evidence of this with cooperation across providers in specific instances [3]. We also see negative evidence that the lack of EONA-like solutions (e.g., recent public spats between AppPs and InfPs [4, 11]) might put us on an unfortunate trajectory with no imminent solution to optimize user experience.

Given this high-level view of EONA, there are two natural issues that remain:

- **Interface design:** What should these new EONA-specific interfaces capture? How do we design an interface that is narrow enough to allow different providers to retain independent control but at the same expressive enough to improve user experience?
- **Control logic:** How should the control logic in the AppP and InfP change in a EONA-enhanced world? How can the EONA interfaces address the conflicts and inefficiencies we outlined in the previous section? Could EONA introduce new types of oscillation problems by introducing new points of synchronization or coupling between control loops?

We present preliminary attempts to address these in the next two sections.

4 EONA Interface Design

We envision that different applications may require different EONA interfaces since their requirements are fundamentally different; e.g., web apps may be latency sensitive while video apps may be throughput sensitive. Furthermore, even within a single application such as video streaming, there are various segments where the requirements are different, e.g., short-form news clips and long-form movies. Instead of proposing complete interface for various applications, we propose a high-level framework for interface design and illustrate it through an example scenario.

Recipe for interface design: Our general goal is to design an interface that is both useful (i.e., help to improve performance) and yet narrow (i.e., expose minimum information). We suggest the following high-level recipe for this process:

1. We envision AppPs and InfPs enumerating a suite of interesting use case scenarios such as those highlighted in §2.
2. For each scenario, we first consider how a hypothetical *global controller* that can utilize all available *data* (e.g., application-level and network-level measurements) to control different *knobs* in the ecosystem (e.g., CDN, bitrate, paths, peering points) to optimize the application experience.
3. We map the data and knobs back to their natural owner; e.g., AppPs control knobs such as choice of CDN/bitrate while InfPs control knobs such as the paths or peering points. Similarly, AppPs might observe application experience while InfPs can observe network congestion.

   If an optimization made by global controller involves knobs owned by one party and data owned by another party, this indicates this information need to be shared to the owner of the knob. At the end of this step, we define a "wide" interface that essentially exposes *all* information of both *control decisions* (i.e., values of the knobs) and the *internal data* (i.e., attributes of application session or network status).
4. Now, for an architecture like EONA to be deployable, the

interfaces should be *narrow* – it must hide certain private information or implementation details and must be minimal enough for widespread adoption by both InfPs and AppPs. Thus, we identify the most *critical* information. That is, we share a small subset of control decisions and internal data attributes that a InfP (AppP) needs (and is willing) to expose to better inform the AppP (InfP) control logic, such that the application quality is still close to that of the global controller.

Note that the control is still independent and AppP and InfP are not relinquishing the knobs; they are merely exposing the information of values of the decisions associated with their knobs.

Illustrative example: Consider our oscillation example from Figure 5. We address this problem by following the recipe.

First, we can consider a hypothetical global controller that will optimally set the knob values: the CDN for each client, the bitrate for each client, and the traffic splits across the peering points for each CDN. In choosing these knobs, it will utilize the data such as the choices of CDNs, choices of bitrates, the peering choices between the ISP and the CDNs, and the load/capacity available at each peering point. In reality, however, there is a natural ownership and control over the different knobs. AppPs control knobs such as choice of CDN/bitrate while InfPs control knobs such as the paths or peering points for each CDN.

Second, each provider may not be willing to share all the data and control decisions with the others. Thus, we need to identify the *critical* knob decisions and data attributes that can help each local control solve the oscillation problem.

In this case, we identify the critical information that the providers must share:

- A2I: The AppPs can provide direct measurements of application experience that the InfP's clients observe with each CDN. A2I also provides an estimate of the total volume of traffic intended to different CDNs so that the InfP can decide a suitable traffic split across peering points.
- I2A: In this case, the InfP might inform the AppPs of its multiple peering points for the different CDNs and the congestion level on each peering point. This can help the AppP choose a better load balancing strategy across CDNs and also attribute problems to the peering point rather than the CDN, and avoid doing a wholesale shift of clients between CDNs.

Note that we are not exposing private information pertinent to specific users or devices or the network topology or the specific traffic engineering strategy.

Open questions: There are three open questions for interface design:

- *Identifying useful knobs and data:* We currently rely on domain knowledge to enumerate the relevant knob and data attributes. However, as past work as shown it may not be trivial to identify which knobs or data have significant impact on experience as there might be several confounding factors and relationships across quality measures [17]. In this case, we might need some type of feature selection techniques (e.g., information gain [21]) to identify the relevant attributes.
- *Balancing effectiveness vs. minimality:* AppPs and InfPs must be able to specify what can or cannot be shared through A2I and I2A. In order that necessary information is shared while preserving privacy concerns, one can think of using standard techniques such as aggregation or other types of "blinding" techniques (e.g., [40]).
- *Standardizing formats:* There are many low-level details we do not discuss here; measurement methodology to obtain the data, units and precision, time intervals of aggregation, and so forth. We believe that some standard body (e.g., IETF) will precisely define these semantics of the data exchange (e.g., [7]).

5 EONA-Enhanced Control

This section discusses new opportunities enabled by EONA and challenges on control logics of AppPs and InfPs and their interactions. As in the earlier section, we use the video use cases to make the discussion concrete.

AppP control logic: EONA can reduce the complexity of the current "blackbox" inference and trial-and-error methods that AppPs use. Let us consider the example of AppPs making CDN selection (in §2). If CDN shares server load information, and provides hints on alternative server IPs via EONA-I2A, then the AppP can first try intra-CDN server switching before switching CDNs to exploit intra-CDN cache locality. Similarly, prior work has shown that current HTTP adaptation players, which can select both CDN and bitrate, perform poorly due to biased interactions with other players' control loops at bottleneck links (e.g., [36, 28]). In Figure 3, if the AppP has additional information from ISPs that attributes bottlenecks to the ISP rather than CDN, the adaptation logic can react to ISP congestion, by switching down bitrate to make the ISP less congested.

InfP control logic: We envision different ways in which InfP's can update their control logic. First, they can use the application experience *directly* by updating their optimization objectives. For example, previous work has shown that optimizing application-level quality metrics can simplify traffic engineering (e.g., [29]). Second, they can use the user experience estimates and use reactive measures if they observe quality degradations. For instance, in the server energy-saving example in §2, the InfP can model how the server capacity impacts quality of experience and redeploy servers if the quality degrades significantly. Finally, the A2I interface enables InfPs to better diagnose performance problem, which might not directly manifest in terms of low-level network performance metrics. For instance, switching from a path with video content cached on-path to another path that has no cache but better network-level performance (e.g., loss rate) can hurt the experience of users.

Control conflicts and instabilities: One natural concern with any system that has independent controllers is the issue of control instabilities and conflicts as we saw in Figure 5.

Note that this problem is not unique to EONA, but EONA might change the problem on several fronts. First, unlike previous work, both AppP and InfP are working on similar experience-driven measures rather than conflicting objectives. Second, EONA's control loops are more aware of others' decisions. For instance, in the example of Figure 5, the oscillation can be avoided if the AppP switches CDN based on peering points' capacity and ISP's peering point selection – if AppP knows that ISP changes its peering point with CDN X to B, which has enough capacity, the AppP would not have switched to CDN Y.

That said, we do acknowledge that it is possible that by introducing tighter coupling between the control loops, a EONA-enabled world might introduce new types of control stability issues. Specifically, today the InfPs are AppPs are operating on very different timescales; e.g., ISP traffic engineering operates on the scales of tens of minutes if not more, while video players react on the timescales of a few seconds. With a EONA world where both the ISP and video player are operating in synchrony, we could introduce new types of instabilities or oscillation problems.

Open challenges: We highlight several open challenges of control logics design under EONA.

- *Search space exploration:* Both AppPs and InfPs are deploying new capabilities that give them more control knobs. With more knobs, however, the search space of options grows combinatorially. A natural question is if and how EONA interfaces can simplify this exploration process.
- *Dealing with staleness:* Because we envision independent and reactive control loops,the data exported by the EONA interfaces may have some inherent delay. Thus, the control logics must also be designed to be robust against such staleness or inaccuracies.
- *Scalability:* Sharing information between AppPs and InfPs may increase the sophistication of the control logics. For example, a typical AppP can collect user experience for tens millions of sessions each day [20], and such large volumes of data can cause serious scalability challenges for the control logic of InfPs, to which recent advances in big data platforms (e.g., [8]) may provide an approach.
- *Oscillations:* An interesting direction for future work is to formally understand if and how EONA can exacerbate control instabilities. We speculate that some sort of dampening or backoff algorithms can help here.
- *Fairness and trust:* There are other natural concerns, such as fairness when an InfP serves multiple AppPs and mutual trust between InfP and AppPs. Here, we assume that external market forces can help; e.g., if the InfP is misbehaving then the AppP can switch providers or can hurt the InfP's reputation. Alternatively, we can envision third-party/neutral validation services that can serve a role in an EONA-enabled architecture.

6 Related Work

Metrics of user experience: Previous work has attempted to derive quantitative relationships between application metrics observed by AppPs and user engagement (e.g., [20, 33]) and attempted to provide unified experience models (e.g., [17]). These works inform the types of experience metrics EONA helps to optimize. Recent work suggests ISPs are inferring user experience from network-layer metrics (e.g., [16, 45, 14, 27, 38]). Since AppPs are in a better position than InfPs to measure the experience, EONA argues that they should directly export these rather than rely on inaccurate inference.

User experience optimization: There are many concurrent efforts to optimize experiences at every piece of the delivery ecosystem including ISPs (e.g., [44, 15, 30]), TCP enhancements (e.g., [25]), client-side (e.g., [28]), and via overlays (e.g., [31, 41]).

Cross-provider sharing and optimization: EONA shares its motivation with prior work on cross-provider sharing across ISPs and P2P providers such as P4P [48] and ALTO [1] and across ISPs and CDNs (e.g., [29, 24, 42, 46]. While EONA is inspired by these efforts, there are key differences: (1) EONA is explicitly driven by and designed for user experience; (2) Information sharing in EONA is bidirectional; and (3) EONA envisions both InfPs and AppPs' control loops working separately to improve user experience.

New knobs for providers: EONA is well positioned to leverage recent advances in software-define networks (e.g., [39]), network function virtualization (e.g., [6]), and new peering capabilities (e.g., [22]). While these offer new control capabilities, EONA provides a new and necessary use case of optimizing user experience.

7 Conclusions

EONA takes a stance that application experience is the key driver for both InfPs and AppPs and that they should cooperate to optimize experience-centric measures together through EONA interfaces. Seen in this light, EONA may seem a radical departure from long-standing networking principles (e.g., end-to-end principles [43]) and beliefs (e.g., net neutrality [26]). On the other hand, however, we see that this future is inevitable and it is already beginning to take root in an ad hoc manner; e.g., Akamai and ISP collaborating [24], Comcast and Netflix reaching out-of-band settlements [3], Netflix and Google reaching out to ISPs [5, 18]. We believe that it behooves the network research community to act soon and inform this future before it unravels.

While EONA is not the only (or optimal) architecture, and it is equally possible that there are other reasons to not deploy EONA; e.g., preferences for infrastructure "neutrality". We do not want to dogmatically assume that one architecture is inherently the right or only choice. Instead, we hope that our arguments and the technical basis of EONA outlined in this position paper marks the beginning of a meaningful debate.

Acknowledgments: This work was funded in part by NSF under award number CNS-1345305.

8 References

[1] Application-layer traffic optimization (alto) problem statement. http://tools.ietf.org/html/rfc5693.

[2] Cisco study. http://goo.gl/tMRwM.

[3] Comcast and netflix reach deal on service. http://goo.gl/BuUhKA.

[4] Netflix is shaming verizon for its slow internet. http://www.businessinsider.com/netflix-slow-verizon-streaming-warning-2014-6.

[5] Netflix open connect content delivery network. https://www.netflix.com/openconnect.

[6] Network functions virtualisation. http://portal.etsi.org/nfv/nfv_white_paper.pdf.

[7] Overview of mpeg-dash standard. http://dashif.org/mpeg-dash/.

[8] Spark. http://spark.incubator.apache.org/.

[9] Workshop on future human-centric multimedia networking.

[10] Workshop on measurements up and down the stack (w-must). http://conferences.sigcomm.org/sigcomm/2012/wmust.php.

[11] Youtube, following netflix, is now publicly shaming internet providers for slow video. http://goo.gl/Buou15.

[12] I. Sodagar. The MPEG-DASH Standard for Multimedia Streaming Over the Internet. *IEEE Multimedia*, 2011.

[13] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, S. Uhlig, and W. Willinger. Anatomy of a large european ixp. In *SIGCOMM '12*.

[14] V. Aggarwal, E. Halepovic, J. Pang, S. Venkataraman, and H. Yan. Prometheus: Toward quality-of-experience estimation for mobile apps from passive network measurements. HotMobile '14.

[15] A. Akella, B. Maggs, S. Seshan, and A. Shaikh. On the performance benefits of multihoming route control. *IEEE/ACM Transactions on Networking*, 16(1):91–104, 2008.

[16] A. Balachandran, V. Aggarwal, E. Halepovic, J. Pang, S. Seshan, S. Venkataraman, and H. Yan. Modeling web quality-of-experience on cellular networks. MobiCom '14.

[17] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang. Developing a predictive model of quality of experience for internet video. In *ACM SIGCOMM '13*.

[18] M. Calder, X. Fan, Z. Hu, E. Katz-Bassett, J. Heidemann, and R. Govindan. Mapping the expansion of google's serving infrastructure. In *ACM IMC '13*.

[19] C. Cranor, T. Johnson, O. Spataschek, and V. Shkapenyuk. Gigascope: a stream database for network applications. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 647–651. ACM, 2003.

[20] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. A. Joseph, A. Ganjam, J. Zhan, and H. Zhang. Understanding the impact of video quality on user engagement. In *Proc. SIGCOMM*, 2011.

[21] P. Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 2012.

[22] N. Feamster, J. Rexford, S. Shenker, R. Clark, R. Hutchins, D. Levin, and J. Bailey. Sdx: A software-defined internet exchange. *Open Networking Summit*, 2013.

[23] N. Feamster, J. Rexford, and E. Zegura. The road to sdn: an intellectual history of programmable networks. *ACM SIGCOMM Computer Communication Review*, 44(2):87–98, 2014.

[24] B. Frank, I. Poese, Y. Lin, G. Smaragdakis, A. Feldmann, B. Maggs, J. Rake, S. Uhlig, and R. Weber. Pushing cdn-isp collaboration to the limit. *ACM SIGCOMM CCR*, 43(3), 2013.

[25] M. Ghobadi, Y. Cheng, A. Jain, and M. Mathis. Trickle: Rate limiting youtube video streaming. In *Proceedings of the USENIX Annual Technical Conference (ATC)*, page 6, 2012.

[26] R. W. Hahn and S. Wallsten. The economics of net neutrality. *The Economists' Voice*, 3(6), 2006.

[27] E. Halepovic, J. Pang, and O. Spatscheck. Can you get me now?: Estimating the time-to-first-byte of http transactions with passive measurements. IMC '12.

[28] J. Jiang, V. Sekar, and H. Zhang. Improving Fairness, Efficiency, and Stability in HTTP-Based Adaptive Streaming with Festive . In *ACM CoNext*, 2012.

[29] W. Jiang, R. Zhang-Shen, J. Rexford, and M. Chiang. Cooperative content distribution and traffic engineering in an isp network. In *ACM SIGMETRICS Performance Evaluation Review*, volume 37, pages 239–250. ACM, 2009.

[30] R. Keralapura, N. Taft, C.-N. Chuah, and G. Iannaccone. Can isps take the heat from overlay networks. In *ACM HotNets '14*.

[31] L. Kontothanassis, R. Sitaraman, J. Wein, D. Hong, R. Kleinberg, B. Mancuso, D. Shaw, and D. Stodolsky. A transport layer for live streaming in a content delivery network. *Proceedings of the IEEE*, 92(9):1408–1419, 2004.

[32] T. Koponen, M. Casado, N. Gude, J. Stribling, L. Poutievski, M. Zhu, R. Ramanathan, Y. Iwata, H. Inoue, T. Hama, et al. Onix: A distributed control platform for large-scale production networks. In *OSDI*, volume 10, pages 1–6, 2010.

[33] S. S. Krishnan and R. K. Sitaraman. Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs. In *IMC*, 2012.

[34] Z. Li, J. Lin, M.-I. Akodjenou, G. Xie, M. A. Kaafar, Y. Jin, and G. Peng. Watching videos from everywhere: a study of the pptv mobile vod system. IMC '12.

[35] H. H. Liu, Y. Wang, Y. R. Yang, H. Wang, and C. Tian. Optimizing cost and performance for content multihoming. *ACM SIGCOMM Computer Communication Review*, 42(4):371–382, 2012.

[36] X. Liu, F. Dobrian, H. Milner, J. Jiang, V. Sekar, I. Stoica, and H. Zhang. A Case for a Coordinated Internet Video Control Plane. In *SIGCOMM*, 2012.

[37] A. Mahimkar, Z. Ge, J. Yates, C. Hristov, V. Cordaro, S. Smith, J. Xu, and M. Stockert. Robust assessment of changes in cellular networks. CoNEXT '13.

[38] A. A. Mahimkar, H. H. Song, Z. Ge, A. Shaikh, J. Wang, J. Yates, Y. Zhang, and J. Emmons. Detecting the performance impact of upgrades in large operational networks. SIGCOMM '10.

[39] 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 CCR*, 38(2):69–74, 2008.

[40] F. McSherry and R. Mahajan. Differentially-private network trace analysis. *ACM SIGCOMM Computer Communication Review*, 41(4):123–134, 2011.

[41] E. Nygren, R. K. Sitaraman, and J. Sun. The akamai network: a platform for high-performance internet applications. *ACM SIGOPS Operating Systems Review*, 44(3):2–19, 2010.

[42] I. Poese, B. Frank, G. Smaragdakis, S. Uhlig, A. Feldmann, and B. Maggs. Enabling content-aware traffic engineering. *ACM SIGCOMM Computer Communication Review*, 42(5):21–28, 2012.

[43] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end arguments in system design. *ACM Transactions on Computer Systems (TOCS)*, 2(4):277–288, 1984.

[44] M. Schapira, Y. Zhu, and J. Rexford. Putting bgp on the right path: A case for next-hop routing. In *ACM HotNets '10*.

[45] M. Z. Shafiq, J. Erman, L. Ji, A. X. Liu, J. Pang, and J. Wang. Understanding the impact of network dynamics on mobile video user engagement. SIGMETRICS '14.

[46] A. Sharma, A. Venkataramani, and R. K. Sitaraman. Distributing content simplifies isp traffic engineering. In *Proceedings of the ACM SIGMETRICS/international conference on Measurement and modeling of computer systems*, pages 229–242. ACM, 2013.

[47] X. S. Wang, A. Balasubramanian, A. Krishnamurthy, and D. Wetherall. How speedy is spdy. In *NSDI'14*.

[48] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. G. Liu, and A. Silberschatz. P4p: Provider portal for applications. In *ACM SIGCOMM Computer Communication Review*, volume 38, pages 351–362. ACM, 2008.