IP Performance Measurement
Internet-Draft
Intended status: Informational
Expires: 3 August 2026

B. I. T. Monclair
M. Olden
I. Kunze, Ed.
CUJO AI

30 January 2026

Quality of Outcome (QoO)
draft-ietf-ippm-qoo-06

Abstract

This document introduces the Quality of Outcome (QoO) network quality score and the corresponding QoO framework as an approach to network quality assessment designed to align with the needs of application developers, users, and operators.

By leveraging the Quality Attenuation metric, QoO provides a method for defining and evaluating application-specific, quality-focused network performance requirements to enable insights for network optimization and simple Quality of Service scores for end-users.

About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-ippm-qoo/.

Discussion of this document takes place on the IP Performance Measurement Working Group mailing list (mailto:ippm@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/ippm/. Subscribe at https://www.ietf.org/mailman/listinfo/ippm/.

Source for this draft and an issue tracker can be found at https://github.com/getCUJO/QoOID.

Status of This Memo

Table of Contents

1. Introduction

This document introduces the Quality of Outcome (QoO) network quality score. It is designed to be easy to understand, while at the same time being objective, adaptable to different network quality needs, and allowing advanced analyses to identify the root cause of network problems (Section 5.4.3 of [I-D.ietf-opsawg-rfc5706bis]). Centered around the QoO score, this document defines the QoO framework which allows application developers to specify quality-focused network performance requirements (for example, regarding latency, throughput, and packet loss) and provides a way to derive the user-facing QoO score by comparing the quality-focused performance requirements of applications to measurements of network performance.

The QoO framework builds on Quality Attenuation [TR-452.1], a network quality metric that enables network operators to achieve fault isolation (Section 5.4.4 of [I-D.ietf-opsawg-rfc5706bis]) and effective network planning through composability [RFC6049]. Quality Attenuation meets most of the requirements for a meaningful network quality score set out in Section 4.1: it is composable, captures the ability of a network to satisfy application requirements, and can be compared to a variety of application needs. However, interpreting raw Quality Attenuation values is difficult for end-users and application developers. The challenge is simplifying the entailed information to present results in an understandable and unambiguous way without losing too much precision and accuracy.

The QoO framework takes a probabilistic approach to this challenge because network stacks and applications adapt dynamically to changing network conditions. Also, applications and underlying networking protocols make separate optimizations based on the perceived network quality over time, making absolute certainty about outcomes practically impossible. However, educated assessments of expected outcomes remain achievable for which the QoO framework uses a per-application, per application-type, or per-Service Level Agreement (SLA) granularity.

This document assumes that network quality can be represented by a minimum required throughput and a set of latency percentiles with corresponding packet loss thresholds. Application developers, regulatory bodies, and other interested parties can then use this representation to describe quality-focused network performance requirements. The QoO framework gives structure to this approach by defining two network quality thresholds: one for optimal application performance and one for unacceptable application performance. The QoO score serves as a linear distance measure between the two distinct thresholds and allows network conditions to be expressed in easily understood terms such as "This network provides 94% of optimal conditions for video conferencing (relative to the threshold for unacceptable performance)" while supporting both comprehensive end-to-end tests and analyses from within the network.

The framework is designed to be flexible in its application scope. QoO measurements may be performed across the complete end-to-end path (from application client to server), or focused on specific network segments such as the customer-facing access network, intermediate transit networks, or server-side infrastructure. Through its composability properties, measurements from different segments can be combined or decomposed to isolate performance issues regardless of where they occur in the network path.

This document defines a minimum viable framework, and often trades precision for simplicity to facilitate adoption and usability in many different contexts, such as active testing from applications and monitoring from network equipment. Assessing the corresponding loss of precision is important and requires combining measurement results with a description of the measurement approach; Section 6.1 provides corresponding guidelines.

Note that this document focuses specifically on the general framework of using quality-focused network performance requirements and corresponding network performance measurements to calculate QoO network quality scores. How applications define and share their network performance requirements, which format is used to publish such requirement information, how operators retrieve such data from applications or services, how the precision of the resulting QoO scores is assessed, and what levels of precision are considered acceptable is out of the scope of this document.

2. Terminology

This document uses the following terminology:

Network: The communication infrastructure that facilitates data transmission between endpoints, including all intermediate devices, links, and protocols that affect the transmission of data. This encompasses both the physical infrastructure and the logical protocols that govern data transmission. The network may support various communication patterns and may span multiple administrative domains.

Network Segment: A portion of the complete end-to-end network path between application endpoints. A network segment may represent a specific administrative domain (e.g., access network, transit network, or server-side infrastructure), a particular technology domain (e.g., Wi-Fi or cellular), or any subset of the path for which independent quality measurements and analysis are desired.

Quality Attenuation: A network quality metric defined in [TR-452.1] that combines latency and packet loss distributions in a unified approach to jointly assess latency and loss characteristics of network performance.

Quality of Experience (QoE): The degree of delight or annoyance of the user of an application or service. See also [P.10].

Quality of Service (QoS): The totality of characteristics of a telecommunications service that bear on its ability to satisfy stated and implied needs of the user of the service. See also [P.10].

Quality of Outcome (QoO): A network quality framework and metric that evaluates network quality based on how closely measured network conditions meet application-specific, quality-focused performance requirements. QoO is a QoS indicator that may be related to, but cannot be considered the same as, the actual QoE of end-users.

QoO Score: A numerical value that represents the distance-based assessment of network quality relative to network performance requirements for optimal and unacceptable application performance on a given network for a specific application, typically expressed as a percentage.

Requirements for Optimal Performance (ROP): The network performance characteristics at which an application achieves optimal performance and quality, beyond which further improvements in network conditions do not result in perceptible improvements in application performance or user experience. When network performance exceeds ROP thresholds, any sub-optimal user experience can be assumed not to be caused by the part of the network path that has been measured for QoO calculations.

Conditions at the Point of Unacceptable Performance (CPUP): The network performance threshold below which an application fails to provide acceptable user experience. Note that 'unacceptable' in this context refers to degraded performance quality rather than complete technical failure of the application. There is no universally strict threshold defining when network conditions become unacceptable for applications.

Composability: The mathematical property that allows network quality measurements to be combined across different network segments or decomposed to isolate specific network components for analysis and troubleshooting.

Accuracy and Precision: "Accuracy" refers to how close measurements are to the value that reflects the real conditions. "Precision" refers to the consistency and repeatability of measurements. These terms are used with their standard statistical meanings and are not interchangeable [ISO5725-1].

3. Overview

The QoO framework produces simple percentage scores that express the network quality in relation to pre-defined network performance requirements of applications. For example: "This network provides 94% of optimal conditions for video conferencing". This way, QoO conveys an intuition for how well an application is expected to perform in the described network with higher scores intended to convey that applications are more likely to have optimal performance.

The QoO framework compares measured network performance against application-specific, quality-focused network performance requirements. Applications define two thresholds:

* Optimal performance (ROP): Network conditions where application performance becomes optimal

* Unacceptable performance (CPUP): Network conditions where application performance becomes unacceptable

The framework calculates QoO scores when measured network performance falls between these thresholds, expressing the network quality as a relative position (percentage) on the linear scale between the thresholds.

The key innovation is using dual thresholds rather than binary pass/fail criteria, providing meaningful scores even when networks are not perfect while accounting for different application sensitivities.

QoO calculations are mathematically composable, enabling network operators to isolate performance bottlenecks across different network segments for precise fault diagnosis and network planning.

QoO scores are expected to correlate with QoE metrics, such as Mean Opinion Score (MOS) [P.800.1], but they are not designed to deliver MOS or QoE scores directly. QoO measures network service quality, not subjective user experience.

It is important to note that the QoO framework itself does not define where QoO scores fall on the spectrum between QoS and QoE metrics. The position on this spectrum depends primarily on how the ROP and CPUP thresholds are chosen. With appropriate threshold selection based on user-acceptance testing and application performance analysis, QoO scores can likely be tuned to be close to (if not identical to) QoE metrics, while still maintaining the objectivity and composability benefits of QoS metrics.

The remainder of this document explains the detailed requirements, mathematical foundations, and implementation considerations for the QoO framework.

4. Motivation

This section describes the features and attributes that a network quality framework must have to be useful for different stakeholders: application developers, end-users, and network operators.

At a high level, end-users need an understandable network quality metric. Application developers require a network quality metric that allows them to evaluate how well their application is likely to perform given the measured network performance. Network operators need a metric that facilitates troubleshooting and optimization of their networks. Existing network quality metrics and frameworks address the needs of one or two of these stakeholders, but there is none that bridges the needs of all three. Examples include throughput metrics that operators use to prove regulatory compliance but with little relevance to application performance, or subjective QoE metrics that are understandable to users but difficult for operators to collect at meaningful scale.

A key motivation for the QoO framework is to bridge the gap between the technical aspects of network performance and the practical needs of those who depend on it. For example, while solutions exist for many of the problems causing high and unstable latency in the Internet, such as bufferbloat mitigation techniques [RFC8290] and improved congestion control algorithms [RFC8033], the incentives to deploy them have remained relatively weak. A unifying framework for assessing network quality can serve to strengthen these incentives significantly.

Bandwidth alone is necessary but not sufficient for high-quality modern network experiences. High idle and working latencies, large delay variations, and unmitigated packet loss are major causes of poor application outcomes. The impact of latency is widely recognized in network engineering circles [BITAG], but benchmarking the quality of network transport remains complex. Most end-users are unable to relate to metrics other than Mbps, which they have long been conditioned to think of as the only dimension of network quality.

Real Time Response under load tests [RRUL] and Responsiveness tests [RPM] make significant strides toward creating a network quality metric that is intended to be closer to application outcomes than bandwidth alone. The latter, in particular, is successful at being relatively relatable and understandable to end-users. However, as noted in [RPM], "Our networks remain unresponsive, not from a lack of technical solutions, but rather a lack of awareness of the problem". This lack of awareness means that some operators might have little incentive to improve network quality beyond increasing bandwidth. For example, despite the availability of open-source solutions such as FQ_CoDel [RFC8290], which has been available for over a decade, vendors rarely implement them in widely deployed equipment (e.g., Wi-Fi routers still commonly exhibit bufferbloat). A universally accepted network quality framework that successfully captures the degree to which networks provide the quality required by applications may help to increase the willingness of vendors to implement such solutions.

The IAB workshop on measuring Internet quality for end-users identified a key insight: users care primarily about application performance rather than network performance. Among the conclusions was the statement, "A really meaningful metric for users is whether their application will work properly or fail because of a lack of a network with sufficient characteristics" [RFC9318]. Therefore, one critical requirement for a meaningful framework is its ability to answer the following question: "Will networking conditions prevent an application from working as intended?".

Answering this question requires several considerations. First, the Internet is inherently stochastic from the perspective of any given client, so absolute certainty is unattainable. Second, different applications have different needs and adapt differently to network conditions. A framework aiming to answer the stated question must accommodate such diverse application requirements. Third, end-users have individual tolerances for degradation in network conditions and the resulting effects on application experience. These variations must be factored into the design of a suitable network quality framework.

4.1. Requirements

This section describes the requirements for an objective network quality framework and metric that is useful for end-users, application developers, and network operators/vendors alike. Specifically, this section outlines the three main requirements and their motivation.

In general, all stakeholders ultimately care about the performance of applications running over a network. Application performance does not only depend on bandwidth but also on the delay and delay variation of network links and computational steps involved in making the application function. These delays depend on how the application places load on the network, how the network is affected by environmental conditions, and the behavior of other users and applications sharing the network resources. Likewise, packet loss (e.g., caused by congestion) can also negatively impact application performance in different ways depending on the class of application.

Different applications may have different needs from a network and may impose different patterns of load. To determine whether an application will likely work well or fail, a network quality framework must compare measurements of network performance to a wide variety of application requirements. It is important that these measurements reflect the actual network service configuration that will handle the application flows, including any traffic prioritization, network slicing, VPN services, or other differentiated service mechanisms (see Section 8.1). Flexibility in describing application requirements and the ability to capture the delay and loss characteristics of a network with sufficient accuracy and precision are necessary to compute a meaningful QoO network quality score that can be used to better estimate application performance.

The framework must also support spatial composition [RFC6049], [RFC6390] to enable operators to take actions when measurements show that applications fail too often. In particular, spatial composition allows results to be divided into sub-results, each measuring the performance of a required sub-milestone that must be reached in time for the application to perform adequately.

To summarize, the QoO framework and the corresponding QoO score should have the following properties in order to be meaningful:

1. Capture a set of network performance metrics which provably correlate to the application performance of a set of different applications as perceived by users.

2. Compare meaningfully to different application requirements.

3. Be composable so operators can isolate and quantify the contributions of different sub-outcomes and sub-paths of the network.

Next, the document focuses on the requirements of each of the mentioned target groups.

4.1.1. Requirements for End-Users

The QoO framework should facilitate a metric that is based on objective QoS measurements (such as throughput [RFC6349], packet loss [RFC6673][RFC7680], delays [RFC2681][RFC7679], and (one-way) delay variations [RFC3393]), correlated to application performance, and relatively understandable for end-users, similar to QoE metrics, such as Mean Opinion Score (MOS) [P.800.1].

If these requirements are met, QoO is a middle ground between QoS and QoE metrics and allows end-users to understand if a network is a likely source of impairment for what they care about: the outcomes of applications. Examples are how quickly a web page loads, the smoothness of a video conference, or whether or not a video game has any perceptible lag.

End-users may have individual tolerances of session quality (i.e., the quality experienced during a single application usage period, such as a video call or gaming session), below which their quality of experience becomes personally unacceptable. However, it may not be feasible to capture and represent these tolerances per user as the user group scales. A compromise is for the QoO framework to place the responsibility for sourcing and representing end-user requirements onto the application developer. Application developers are expected to perform user-acceptance testing (UAT) of their application across a range of users, terminals, and network conditions to determine the terminal and network requirements that will meet the acceptability thresholds for a representative subset of their end-users. Performing UAT helps developers estimate what QoE new end-users are likely to experience based on the application’s network performance requirements. These requirements can evolve and improve based on feedback from end-users, and in turn better inform the application’s requirements towards the network. Some real world examples where ’acceptable levels’ have been derived by application developers include:

* Remote music collaboration: <20ms one-way latency [JamKazam]

* Cloud gaming: >15Mbps downlink throughput and 80ms round-trip time (RTT) [XboxNetReqs] (specific requirements vary by game and platform; see [CSGO] for an example study on the impact of latency on Counter Strike: Global Offensive)

* Virtual reality (VR): <20ms RTT from head motion to rendered update in VR ([RFC9817]; see [G.1051] for latency measurement and interactivity scoring)

Note that developers of similar applications may have arrived at different figures.

4.1.2. Requirements from Application and Platform Developers

The QoO framework needs to provide developers the ability to describe the quality-focused network performance requirements of their applications. The network performance requirements must include all relevant dimensions of network quality so that applications sensitive to different network quality dimensions can all evaluate the network accurately. Not all developers have network expertise, so to make it easy for developers to use the framework, developers must be able to specify network performance requirements approximately. Therefore, it must be possible to describe both simple and complex network performance requirements. The framework also needs to be flexible so that it can be used with different kinds of traffic and that extreme network performance requirements which far exceed the needs of today’s applications can also be articulated.

If these requirements are met, developers of applications and platforms can state or test their network requirements and evaluate whether the network is sufficient for an optimal application outcome. Both the application developers with networking expertise and those without can use the framework.

4.1.3. Requirements for Network Operators and Network Solution Vendors

From an operator perspective, the key is to have a framework that allows finding the network quality bottlenecks and objectively comparing different networks, network configurations, and technologies. To achieve this goal, the framework must support mathematically sound compositionality (’addition’ and ’subtraction’) as network operators rarely manage network traffic end-to-end. If a test is purely end-to-end, the ability to find bottlenecks may be gone. If, however, measurements can be taken in both end-to-end (e.g., a-b-c-d-e) and partial (e.g., a-b-c) fashion, the results can be decomposed to isolate the areas outside the influence of a network operator. In other words, the network quality of a-b-c and d-e can be separated. Compositionality is essential for fault detection and accountability.

By having mathematically correct composition, a network operator can measure two segments separately, perhaps even with different approaches, and combine or correlate the results to understand the end-to-end network quality.

Another example where composition is useful is troubleshooting a typical web page load sequence over TCP. If web page load times are too slow, DNS resolution time, TCP RTT, and the time it takes to establish TLS connections can be measured separately to get a better idea of where the problem is. A network quality framework should support this kind of analysis to be maximally useful for operators.

The framework must be applicable in both lab testing and monitoring of production networks. It must be useful on different time scales, and it cannot have a dependency on network technology or OSI layers.

If these requirements are met, network operators can monitor and test their network and understand where the true bottlenecks are, regardless of network technology.

5. Background

The foundation of the QoO framework is Quality Attenuation [TR-452.1]. This work will not go into detail about how to measure Quality Attenuation, but some relevant techniques are:

* Active probing with the Two-Way Active Measurement Protocol (TWAMP) Light [RFC5357], the Simple Two-Way Active Measurement Protocol (STAMP) [RFC8762], or the Isochronous Round-Trip Tester (IRTT) [IRTT]

* Latency Under Load Tests

* Speed Tests with latency measures

* Simulating real traffic

* End-to-end measurements of real traffic

* TCP SYN ACK or DNS Lookup RTT Capture

* On-Path Telemetry methods (IOAM [RFC9197], AltMark [RFC9341])

* Estimation

Quality Attenuation represents quality measurements as distributions. Using latency distributions to measure network quality has been proposed by various researchers and practitioners (e.g., [Kelly], [RFC8239], and [RFC6049]). Quality Attenuation additionally uses a packet loss distribution for which it views packet loss as infinite (or too late to be of use, e.g., > 5 seconds) latency [TR-452.1], similar to the One-Way Loss Metric for IP Performance Metrics (IPPM) [RFC7680], which defines packet loss in terms of packets that fail to arrive within a specified time threshold. The novelty of [TR-452.1] lies in its unified treatment of latency and loss within a single distributional framework, enabling mathematical composition of network segments.

Latency distributions can be gathered via both passive monitoring and active testing [RFC7799]. The active testing can use any type of traffic, such as connection-oriented TCP and QUIC or connectionless UDP. It can be applied across different layers of the protocol stack and is network technology independent, meaning it can be gathered in an end-user application, within some network equipment, or anywhere in between. Passive methods rely on observing and time-stamping packets traversing the network. Examples of this include TCP SYN and SYN/ACK packets (Section 2.2 of [RFC8517]) and the QUIC spin bit [RFC9000][RFC9312].

A key assumption behind the choice of latency distribution is that different applications and application categories fail at different points of the latency distribution. Some applications, such as downloads, have lenient latency requirements when compared to real-time applications. Video conferences are typically sensitive to high 90th percentile latency and to the difference between the 90th and the 99th percentile. Online gaming typically has a low tolerance for high 99th percentile latency. All applications require a minimum level of throughput and a maximum packet loss rate. A network quality metric that aims to generalize network quality must take the latency distribution, throughput, and packet loss into consideration.

Two distributions can be composed using convolution [TR-452.1].

5.1. Discussion of Other Network Quality Metrics

Numerous network quality metrics and associated frameworks have been proposed, adopted, and, at times, misapplied over the years. The following is a brief overview of several key network quality metrics.

Each metric is evaluated against the three criteria established in Section 4.1. Table 1 summarizes the properties of each of the surveyed metrics.

Metric	Can Assess How Well Applications Are Expected to Work	Easy to Articulate Application Requirements	Composable
Throughput	Yes for some applications	Yes	No
Mean latency	Yes for some applications	Yes	Yes
99th Percentile of Latency	No	No	No
Variance of latency	No	No	Yes
IPDV	Yes for some applications	No	No
PDV	Yes for some applications	No	No
Trimmed mean of latency	Yes for some applications	Yes	No
Round Trips Per Minute	Yes for some applications	Yes	No
Quality Attenuation	Yes	No	Yes

Table 1: Summary of Performance Metrics Properties

The column "Can Assess How Well Applications Are Expected to Work" indicates whether a metric can, in principle, capture relevant information to assess application performance, assuming that measurements cover the properties of the end-to-end network path that the application uses. "Easy to Articulate Application Requirements" refers to the ease with which application-specific requirements can be expressed using the respective metric. "Composable" indicates whether the metric supports mathematical composition to enable detailed network analysis.

5.1.1. Throughput

Throughput is related to user-observable application outcomes because there must be enough bandwidth available. Adding extra bandwidth above a certain threshold will, at best, receive diminishing returns (and any returns are often due to reduced latency). It is not possible to assess optimal or unacceptable application performance based on throughput alone for most applications. Throughput can be compared to a variety of application requirements, but since there is no direct correlation between throughput and application performance, it is not possible to conclude that an application will work well even if it is known that enough throughput is available.

Throughput cannot be composed.

5.1.2. Mean Latency

Mean latency relates to user-observable application outcomes in the sense that the mean latency must be low enough to support a good experience. However, it is not possible to conclude that a general application will work well if the mean latency is good enough [BITAG].

Mean latency can be composed. For example, if the mean latency values of links a-b and b-c are known, then the mean latency of the composition a-b-c is the sum of a-b and b-c.

5.1.3. 99th Percentile of Latency

The 99th percentile of latency relates to user-observable application outcomes because it captures some information about how bad the tail latency is. If an application can handle 1% of packets being too late, for instance by maintaining a playback buffer, then the 99th percentile can be a good metric for measuring application performance. It does not work as well for applications that are very sensitive to overly delayed packets because the 99th percentile disregards all information about the delays of the worst 1% of packets.

It is not possible to compose 99th-percentile values.

5.1.4. Variance of Latency

The variance of latency can be calculated from any collection of samples, but network latency is not necessarily normally distributed. As such, it can be difficult to extrapolate from a measure of the variance of latency to how well specific applications will work.

The variance of latency can be composed. For example, if the variance values of links a-b and b-c are known, then the variance of the composition a-b-c is the sum of the variances a-b and b-c.

5.1.5. Inter-Packet Delay Variation (IPDV)

The most common definition of IPDV [RFC5481] measures the difference in one-way delay between subsequent packets. Some applications are very sensitive to this performance characteristic because of time-outs that cause later-than-usual packets to be discarded. For some applications, IPDV can be useful in assessing application performance, especially when it is combined with other latency metrics. IPDV does not contain enough information to assess how well a wide range of applications will work.

IPDV cannot be composed.

5.1.6. Packet Delay Variation (PDV)

The most common definition of PDV [RFC5481] measures the difference in one-way delay between the smallest recorded latency and each value in a sample.

PDV cannot be composed.

5.1.7. Trimmed Mean of Latency

The trimmed mean of latency is the mean computed after the worst x percent of samples have been removed. Trimmed means are typically used in cases where there is a known rate of measurement errors that should be filtered out before computing results.

In the case where the trimmed mean simply removes measurement errors, the result can be composed in the same way as the mean latency. In cases where the trimmed mean removes real measurements, the trimming operation introduces errors that may compound when composed.

5.1.8. Round-trips Per Minute

Round-trips per minute [RPM] is a metric and test procedure specifically designed to measure delays as experienced by application-layer protocol procedures, such as HTTP GET, establishing a TLS connection, and DNS lookups. Hence, it measures something very close to the user-perceived application performance of HTTP-based applications. RPM loads the network before conducting latency measurements and is, therefore, a measure of loaded latency (also known as working latency), and well-suited to detecting bufferbloat [Bufferbloat].

RPM is not composable.

5.1.9. Quality Attenuation

Quality Attenuation is a network quality metric that combines dedicated latency and packet loss distributions into a single variable [TR-452.1]. It relates to user-observable outcomes in the sense that they can be measured using the Quality Attenuation metric directly, or the Quality Attenuation value describing the time-to-completion of a user-observable outcome can be computed if the Quality Attenuation of each sub-goal required to reach the desired outcome is known [Haeri22].

Quality Attenuation is composable because the convolution of Quality Attenuation values allows computing the time it takes to reach specific outcomes given the Quality Attenuation of each sub-goal and the causal dependency conditions between them [Haeri22].

6. Sampling and Network Requirements

This section presents considerations for collecting network performance measurements and specifying network performance requirements, together enabling the QoO framework and calculating QoO scores.

6.1. Sampling Requirements

To ensure broad applicability across diverse use cases, the QoO framework deliberately avoids prescribing specific conditions for sampling, such as fixed time intervals or defined network load levels. This flexibility enables deployment in both controlled and production environments.

For a complete assessment, the QoO framework requires a latency distribution, packet loss measurements, and throughput estimates as described in Section 6.2. When measurements are taken during periods of network load, the result naturally includes latency under load. In scenarios such as passive monitoring of production traffic, capturing artificially loaded conditions may not always be feasible, whereas passively observing the actual network load may be possible.

Modeling the full latency distribution may be too complex to allow for easy adoption of the framework. Instead, reporting latency at selected percentiles offers a practical compromise between accuracy and deployment considerations. A commonly accepted set of percentiles spanning from the 0th to the 100th in a logarithmic-like progression has been suggested by others [BITAG] and is recommended here: [0th, 10th, 25th, 50th, 75th, 90th, 95th, 99th, 99.9th, 100th].

The framework is agnostic to traffic direction but mandates that measurements specify whether latency is one-way or round-trip.

Importantly, the framework does not enforce a minimum sample count. This means that even a small number of samples (e.g., 10) could technically constitute a distribution but such cases are clearly insufficient for statistical confidence. The intent is to balance rigor with practicality, recognizing that constraints vary across devices, applications, and deployment environments.

To support reproducibility and enable confidence analysis, each measurement must be accompanied by the following metadata:

* Description of the measurement path, including the endpoints (source and destination), network segments traversed, measurement points (if applicable), and direction (uplink, downlink, or bidirectional)

* Timestamp of first sample

* Total duration of the sampling period

* Number of samples collected

* Sampling method, including:

  - Cyclic: One sample every N milliseconds (specify N)

  - Burst: X samples every N milliseconds (specify X and N)

  - Passive: Opportunistic sampling of live traffic (non-uniform intervals)

These metadata elements are essential for interpreting the precision and reliability of the measurements. As demonstrated in [QoOSimStudy], low sampling frequencies and short measurement durations can lead to misleadingly optimistic or imprecise QoO scores.

To assess the precision of network performance measurements, implementers should consider:

* The repeatability of measurements under similar network conditions

* The impact of sampling frequency and duration on percentile estimates, particularly for high percentiles (e.g., 99th, 99.9th)

* The measurement uncertainty introduced by hardware/software timing jitter, clock synchronization errors, and other system-level noise sources

* The statistical confidence intervals for percentile estimates based on sample size

Acceptable levels of precision depend on the use case. Implementers should document their precision assessment methodology and report precision metrics alongside QoO scores when precision is critical for the use case.

6.2. Describing Network Performance Requirements

The QoO framework builds upon the work already proposed in the Broadband Forum standard called Quality of Experience Delivered (QED) [TR-452.1], which defines the Quality Attenuation metric. Correspondingly, QoO expresses network performance requirements as a set of percentile-latency tuples with corresponding packet loss thresholds and a minimum required throughput. For example, a requirement might state: at 4Mbps, 90% of packets must arrive within 100ms, and 100% within 200ms, implying 0% packet loss. This list can be minimal (e.g., 100% within 200ms) or extended as needed and different percentiles may be used to characterize different applications. Still, it might be beneficial for future standardization activities to converge on a fixed set of general percentiles or for specific applications/ application classes to make QoO measurements between different providers more comparable. For the sake of simplicity, this document only states that the latency percentiles in the requirements must match one or more of the percentiles defined in the measurements, i.e., one can set requirements at the [0th, 10th, 25th, 50th, 75th, 90th, 95th, 99th, 99.9th, 100th] percentiles. Packet loss rates and bandwidth must be reported as separate values.

Applications do of course have throughput requirements, and thus a complete framework for application-level network quality must also take capacity into account. Insufficient bandwidth may give unacceptable application outcomes without necessarily inducing a lot of latency or packet loss. Therefore, the network requirements must include a minimum throughput requirement. A fully specified requirement can be thought of as specifying the latency and loss requirements to be met while the end-to-end network path is loaded in a way that is at least as demanding of the network as the application itself. This may be achieved by running the actual application and measuring delay and loss alongside it, or by generating artificial traffic to a level at least equivalent to the application traffic load.

Whether the requirements are one-way or two-way must be specified. Where the requirement is one-way, the direction (user-to-network or network-to-user) must be specified. In case of a two-way requirement, a decomposition into uplink and downlink components may be specified.

Network performance requirements and measurements are already standardized in the QED framework [TR-452.1]. This document extends the QED framework with a method that translates the network performance requirements and measurements into a network quality score that quantifies how close the provided network conditions are to the optimal conditions specified by the requirements.

To that aim, first recall the key design goal of establishing a quantifiable distance between optimal and unacceptable network conditions, thereby enabling an objective assessment of relative quality. Accordingly, the requirements specification is extended to define both the network performance required for achieving optimal application performance and the lower network performance threshold below which the application performance is considered unacceptable.

The two ends of the distance measure correspond to the Requirements for Optimal Performance (ROP) and the Conditions at the Point of Unacceptable Performance (CPUP). For example, ROP could be defined as: at 4Mbps, 99% of packets need to arrive within 100ms, 99.9% within 200ms, and 0.1% packet loss is acceptable for the outcome to be as intended. Similarly, CPUP could be defined as: if 99% of the packets have not arrived after 200ms, or 99.9% within 300ms, the perceived service will be unacceptable.

If a latency percentile is included in the ROP, it must also be defined in the CPUP, and vice versa, i.e., neither specification should define a percentile that is not present in the other. For example, if the 99.9th percentile is part of the CPUP then the ROP must also include the 99.9th percentile.

The derivation of ROP and CPUP values requires standardized testing conditions to ensure consistency and accuracy. Application developers should publish their testing methodologies, including the network conditions, hardware configurations, and measurement procedures used to establish these thresholds. Without such standardization, the overall accuracy and precision of QoO scores may be reduced due to variations in testing approaches across different applications and developers.

Developers are encouraged to follow relevant standards for testing methodologies, such as ITU-T P-series recommendations for subjective quality assessment ([P.800], [P.910], [P.1401]) and IETF IPPM standards for network performance measurement ([RFC7679], [RFC7680], [RFC6673]). These standards provide guidance on test design, measurement procedures, and statistical analysis that can help ensure consistent and reproducible threshold definitions.

6.3. Creating Network Performance Requirement Specifications

This document does not define a standardized approach for creating a quality-focused network performance requirement specification. Instead, this section provides general guidance on and a rough outline for deriving an admittingly subjective requirement specification, aiming to create a basis for future standardization efforts focusing on developing a standardized, objective requirement creation framework. Additional information is provided in [QoOAppQualityReqs]. Direct use of the approach described below in production scenarios is discouraged.

When determining quality-focused network performance requirements for an application, the goal is to identify the network conditions where application performance is optimal and where it becomes unacceptable. There is no universally strict threshold at which network conditions render an application unacceptable. For optimal performance, some applications may have clear definitions, but for others, such as web browsing and gaming, lower latency and loss is always preferable.

One approach for deriving possible thresholds is to run the application over a controlled network segment with adjustable quality and then vary the network conditions while continuously observing the resulting application-level performance. The latter can be assessed manually by the entity performing the testing or using automated methods, such as recording video stall duration within a video player. Additionally, application developers could set thresholds for acceptable fps, animation fluidity, i/o latency (voice, video, actions), or other metrics that directly affect the user experience and measure these user-facing metrics during tests to correlate the metrics with the network conditions.

Using this scenario, one can first establish a baseline under excellent network conditions. Network conditions can then be gradually worsened by adding delay or packet loss or decreasing network capacity until the application no longer performs optimally. The corresponding network conditions identify the minimal requirements for optimal performance (ROP). Continuing to worsen the network conditions until the application fails completely eventually yields the network conditions at the point of unacceptable performance (CPUP).

Note that different users may have different tolerance levels for application degradation. Hence, tests conducted by a single entity likely result in highly subjective thresholds. The thresholds established should represent acceptable performance for the target user base, which may require user studies or market research to determine appropriate values.

As stated at the beginning of this section, this document does not define a standardized approach for creating a quality-focused network performance requirement specification and directly using the approach described above is discouraged.

7. Calculating QoO

The QoO score assesses how close the measured network performance is to the network conditions needed for optimal application performance, incorporating both latency and packet loss. There are three key scenarios:

* The network meets all requirements for optimal performance (ROP). QoO Score: 100%.

* The network fails one or more criteria for conditions at the point of unacceptable performance (CPUP). QoO Score: 0%.

* The network performance falls between optimal and unacceptable. In this case, a continuous QoO score between 0% and 100% is computed by taking the worst score derived from latency and packet loss.

Note that the QoO score should reflect the directionality of the measurements (one-way or round-trip) as specified in the network performance requirements. When comparing measurements to requirements, both must use the same directionality and, for one-way measurements, the same direction (uplink or downlink).

7.1. Calculation

7.1.1. Latency Component

The latency-based QoO score is computed as follows:

```
QoO_latency = min_{i}(min(max((1 - ((ML_i - ROP_i) / (CPUP_i -
ROP_i))) * 100, 0), 100))
```

Where:

* ML_i is the Measured Latency at percentile i.

* ROP_i is the latency as indicated in the Requirement for Optimal Performance at percentile i.

* CPUP_i is the latency as indicated in the Condition at the Point of Unacceptable Performance at percentile i.

7.1.2. Packet Loss Component

Packet loss is considered as a separate, single measurement that applies across the entire traffic sample, not at each percentile. The packet loss score is calculated using a similar interpolation formula, but based on the total measured packet loss (MLoss) and the packet loss thresholds defined in the ROP and CPUP:

```
QoO_loss = min(max((1 - ((MLoss - ROP_Loss) / (CPUP_Loss -
ROP_Loss))) * 100, 0), 100)
```

Where:

* MLoss is the Measured Packet Loss.

* ROP_Loss is the acceptable packet loss for optimal performance.

* CPUP_Loss is the packet loss threshold beyond which the application becomes unacceptable.

7.1.3. Overall QoO Calculation

The overall QoO score is the minimum of the latency and packet loss scores:

```
QoO = min(QoO_latency, QoO_loss)
```

7.2. Example

The following example illustrates the QoO calculations.

Example requirements and measured data:

* ROP: 4Mbps {99%, 200ms}, {99.9%, 300ms} 1% loss

* CPUP: {99%, 500ms}, {99.9%, 600ms} 5% loss

* Measured Latency: 99% = 350ms, 99.9% = 375ms

* Measured Packet Loss: 2%

* Measured Minimum Bandwidth: 32Mbps / 28Mbps

QoO_latency = min(min(max((1 - (350ms - 200ms) / (500ms - 200ms)) * 100, 0), 100), min(max((1 - (375ms - 300ms) / (600ms - 300ms)) * 100, 0), 100)) = min(50.00, 75.00) = 50.00

QoO_loss = min(max((1 - (2% - 1%) / (5% - 1%)) * 100, 0), 100) = 75.00

QoO = min(QoO_latency, QoO_loss) = min(50.00, 75.00) = 50.00

In this example, the network scores 50% on the QoO assessment range between unacceptable and optimal for the given application when using the measured network and considering both latency and packet loss. The score implies that the latency impact dominates the packet loss impact and that the network overall provides conditions at the midway point of the performance range.

8. Operational Considerations

This section discusses general operational considerations concerning the use of the QoO framework.

8.1. Deployment Considerations

The QoO framework assumes that measurements reflect the actual connectivity service that will be provided to application flows. However, networks may offer multiple connectivity service levels (e.g., VPN services [RFC2764], corporate customer tiers, and network slicing configurations [RFC9543]). In such deployments, it is important to ensure that:

* Measurements are taken using the same connectivity service level that will be used by the application

* The measurement methodology accounts for any traffic prioritization, differentiated services, or quality-of-service mechanisms that may affect application performance

* Network configurations and policies that will apply to application traffic are reflected in the measurement conditions

Failing to align measurements with the actual service delivery may result in QoO scores that do not accurately reflect the application’s expected performance.

8.2. Adaptive Applications

Many modern applications are adaptive, meaning they can adjust their behavior based on network conditions. For example, video streaming applications may reduce bit rate when bandwidth is limited, or increase buffer size when latency is high.

For adaptive applications, there are typically different levels of optimal performance rather than a single absolute threshold. For example, a video streaming application might provide different available video resolutions, ranging from 4K to 480p resolution. Combined with different transmission latencies, each of these resolutions can induce varying levels of perceived usability.

The QoO framework can accommodate such applications by defining multiple ROP/CPUP thresholds corresponding to different quality levels. The framework can then assess how well the application will achieve each quality level, providing a more nuanced view of application performance than a simple binary pass/fail metric. Another, less complex approach at the cost of reduced fidelity in the QoO score, is to set the threshold for optimal performance at the highest rendition available for the video stream, and the threshold for unacceptability where the lowest rendition cannot be delivered without resulting in stalling events.

Application developers implementing adaptive applications should consider publishing quality profiles that define network performance requirements for different adaptation levels, enabling more accurate QoO assessment.

8.3. Sensitity to Sampling Accuracy

While the QoO framework itself places no strict requirement on sampling patterns or measurement technology, a simulation study [QoOSimStudy] conducted to inform the creation of this document examined the metric’s real-world applicability under varying conditions and made the following conclusions:

1. Sampling Frequency: Slow sampling rates (e.g., <1Hz) risk missing rare, short-lived latency spikes, resulting in overly optimistic QoO scores.

2. Measurement Noise: Measurement errors on the same scale as the thresholds (ROP, CPUP) can distort high-percentile latencies and cause artificially lower QoO.

3. Requirement Specification: Slightly adjusting the latency thresholds or target percentiles can cause significant changes in QoO, especially when the measurement distribution is near a threshold.

4. Measurement Duration: Shorter tests with sparse sampling tend to underestimate worst-case behavior for heavy-tailed latency distributions, biasing QoO in a positive direction.

In summary, overly noisy or inaccurate latency samples can artificially inflate worst-case percentiles, thereby driving QoO scores lower than actual network conditions would warrant. Conversely, coarse measurement intervals can miss short-lived spikes entirely, resulting in an inflated QoO.

From these findings, we deduce the following guidelines for practical application:

* Calibrate the combination of sampling rate and total measurement period to capture fat-tailed distributions of latency with sufficient accuracy.

* Avoid or account for significant measurement noise where possible (e.g., by calibrating time sources, accounting for clock drift, considering hardware/software measurement jitter).

* Thoroughly test application requirement thresholds so that the resulting QoO scores accurately reflect application performance.

These guidelines are non-normative but reflect empirical evidence on how QoO performs.

8.4. Insights From User Testing

While subjective QoE testing as specified in the ITU-T P-series recommendations ([P.800], [P.910], and [P.1401]) is out of scope of this document, a study involving 25 participants tested the QoO framework in real-world settings [QoOUserStudy]. Participants used specially equipped routers in their homes for ten days, providing both network performance data and feedback through pre- and post-trial surveys.

Participants found QoO scores more intuitive and actionable than traditional metrics (e.g., speed tests). QoO directly aligned with their self-reported experiences, increasing trust and engagement.

These results indicate that users find it easier to correlate QoO scores with real-world application performance than, for example, a speed test. As such, QoO is expected to help bridge technical metrics with application performance. However, the specific impact of QoO should be studied further, for example, via comparative studies with blinded methodologies that compare QoO to other QoS-type approaches or application-provided QoE ratings as the mentioned study’s design might have introduced different forms of bias.

9. Known Weaknesses and Open Questions

The described QoO framework simplifies the comparison between network performance requirements from applications and Quality Attenuation measurements. This simplification introduces several artifacts, the significance of which may vary depending on the context. The following section discusses some known limitations.

9.1. Volatile Networks

Volatile networks - in particular, mobile cellular networks - pose a challenge for network quality prediction, with the level of assurance of the prediction likely to decrease as session duration increases. Historic network conditions for a given cell may help indicate times of network load or reduced transmission power, and their effect on throughput/latency/loss. However, as terminals are mobile, the signal bandwidth available to a given terminal can change by an order of magnitude within seconds due to physical radio factors. These include whether the terminal is at the edge of a cell for a radio network, or undergoing cell handover, the radio interference and fading from the local environment, and any switch between radio bearers with differing signal bandwidth and transmission-time intervals (e.g., 3GPP 4G and 5G). This suggests a requirement for measuring Quality Attenuation to and from an individual terminal, as that can account for the factors described above. How that facility is provisioned onto individual terminals and how terminal-hosted applications can trigger a Quality Attenuation query, is an open question.

9.2. Missing Temporal Information in Distributions.

The two latency series (1,200,1,200,1,200,1,200,1,200) and (1,1,1,1,1,200,200,200,200,200) have identical distributions, but may have different application performance. Ignoring this information is a tradeoff between simplicity and precision. To capture all information necessary to adequately capture outcomes quickly gets into extreme levels of overhead and high computational complexity. An application’s performance depends on reactions to varying network conditions, meaning nearly all different series of latencies may have different application outcomes.

9.3. Subsampling the Real Distribution

Additionally, it is not feasible to capture latency for every packet transmitted. Probing and sampling can be performed, but some aspects will always remain unknown. This introduces an element of uncertainty and perfect predictions cannot be achieved; rather than disregarding this reality, it is more practical to acknowledge it. Therefore, discussing the assessment of outcomes provides a more accurate and meaningful approach.

9.4. Assuming Linear Relationship Between Optimal Performance and Unusable

It has been shown that, for example, interactivity cannot be modeled by a linear scale [G.1051]. Thus, the linear modeling proposed here adds an error in estimating the perceived performance of interactive applications.

One can conjure up scenarios where 50ms latency is actually worse than 51ms latency as developers may have chosen 50ms as the threshold for changing quality, and the threshold may be imperfect. Taking these scenarios into account would add another magnitude of complexity to determining network performance requirements and finding a distance measure (between requirement and actual measured capability).

9.5. Binary Bandwidth Threshold

Choosing a binary bandwidth threshold is to reduce complexity, but it must be acknowledged that many applications are not that simple. Network requirements can be set up per quality level (resolution, frames per-second, etc.) for the application if necessary.

9.6. Arbitrary Selection of Percentiles

A selection of percentiles is necessary for simplicity, because more complex methods may slow adoption of the framework. The 0th (minimum) and 50th (median) percentiles are commonly used for their inherent significance. According to [BITAG], the 90th, 98th, and 99th percentiles are particularly important for certain applications. Generally, higher percentiles provide more insight for interactive applications, but only up to a certain threshold beyond which applications may treat excessive delays as packet loss and adapt accordingly. The choice between percentiles such as the 95th, 96th, 96.5th, or 97th is not universally prescribed and may vary between application types. Therefore, percentiles must be selected arbitrarily, based on the best available knowledge and the intended use case.

10. Security Considerations

The QoO framework introduces a method for assessing network quality based on probabilistic outcomes derived from latency, packet loss, and throughput measurements. While the framework itself is primarily analytical and does not define a new protocol, some security considerations arise from its deployment and use.

10.1. Measurement Integrity and Authenticity

QoO relies upon accurate and trustworthy measurements of network performance. If an attacker can manipulate these measurements, either by injecting falsified data or tampering with the measurement process, they could distort the resulting QoO scores. This could mislead users, operators, or regulators into making incorrect assessments of network quality.

To mitigate this risk:

* Measurement agents have to authenticate with the systems collecting or analyzing QoO data.

* Measurement data has to be transmitted over secure channels (e.g., (D)TLS) to ensure confidentiality and integrity.

* Digital signatures may be used to verify the authenticity of measurement reports.

10.2. Risk of Misuse and Gaming

As QoO scores may influence regulatory decisions, SLAs, or user trust, there is a risk that network operators or application developers might attempt to "game" the system. For example, they might optimize performance only for known test conditions or falsify requirement thresholds to inflate QoO scores.

Mitigations include:

* Independent verification of application requirements and measurement methodologies.

* Use of randomized testing procedures.

* Transparency in how QoO scores are derived and what assumptions are made.

10.3. Denial-of-Service (DoS) Risks

Active measurement techniques used to gather QoO data (e.g., TWAMP, STAMP, and synthetic traffic generation) can place additional load on a network. If not properly rate-limited, this may inadvertently degrade services offered by a network or be exploited by malicious actors to launch DoS attacks.

To mitigate these risks, the following is recommended:

* Implement rate-limiting and access control for active measurement tools.

* Ensure that measurement traffic does not interfere with critical services.

* Monitor for abnormal measurement patterns that may indicate abuse.

10.4. Trust in Application Requirements

QoO depends on application developers to define ROP and CPUP. If these are defined inaccurately-either unintentionally or maliciously-the resulting QoO scores may be misleading.

To address such risks, the following recommendations are made:

* Encourage peer review and publication of application requirement profiles.

* Where QoO is used for regulatory or SLA enforcement, require independent validation of requirement definitions.

11. Privacy Considerations

QoO measurements may involve collecting detailed performance data from end-user devices or applications. Depending on the deployment model, this includes metadata such as IP addresses, timestamps, or application usage patterns.

To protect user privacy:

* Data collection should be subject to user consent prior to collecting data.

* Data collection should follow the principle of data minimization, only collecting what is strictly necessary.

* Privacy-sensitive information (e.g., Personally Identifiable Information (PII)) should be anonymized or pseudonymized where possible.

* Users should be informed about what data is collected and how it is used, in accordance with applicable privacy regulations (e.g., General Data Protection Regulation (GDPR)).

12. IANA Considerations

This document has no IANA actions.

13. Implementation status

Note to RFC Editor: This section must be removed before publication of the document.

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in [RFC7942]. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to [RFC7942], "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

13.1. qoo-c

* Link to the open-source repository:

  https://github.com/getCUJO/qoo-c

* The organization responsible for the implementation:

  CUJO AI

* A brief general description:

A C library for calculating Quality of Outcome

* The implementation’s level of maturity:

  A complete implementation of the specification described in this document

* Coverage:

  The library is tested with unit tests

* Licensing:

  MIT

* Implementation experience:

  Tested by the author. Needs additional testing by third parties.

* Contact information:

  Bj淡rn Ivar Teigen Monclair: bjorn.monclair@cujo.com

* The date when information about this particular implementation was last updated:

  27th of May 2025

13.2. goresponsiveness

* Link to the open-source repository:

  https://github.com/network-quality/goresponsiveness

  The specific pull-request: https://github.com/network-quality/goresponsiveness/pull/56

* The organization responsible for the implementation:

  University of Cincinatti for goresponsiveness as a whole, Domos for the QoO part.

* A brief general description:

  A network quality test written in Go. Capable of measuring RPM and QoO.

* The implementation’s level of maturity:

Under active development; partial QoO support integrated.

* Coverage:

  The QoO part is tested with unit tests

* Licensing:

  GPL 2.0

* Implementation experience:

  Needs testing by third parties

* Contact information:

  Bj淡rn Ivar Teigen Monclair: bjorn.monclair@cujo.com

  William Hawkins III: hawkinwh@ucmail.uc.edu

* The date when information about this particular implementation was last updated:

  10th of January 2024

14. References

14.1. Normative References

[RFC6049] Morton, A. and E. Stephan, "Spatial Composition of Metrics", RFC 6049, DOI 10.17487/RFC6049, January 2011, <https://www.rfc-editor.org/rfc/rfc6049>.

[RFC6390] Clark, A. and B. Claise, "Guidelines for Considering New Performance Metric Development", BCP 170, RFC 6390, DOI 10.17487/RFC6390, October 2011, <https://www.rfc-editor.org/rfc/rfc6390>.

[TR-452.1] Broadband Forum, "TR-452.1: Quality Attenuation Measurement Architecture and Requirements", September 2020, <https://www.broadband-forum.org/download/TR-452.1.pdf>.

14.2. Informative References

[BITAG] BITAG, "Latency Explained", October 2022, <https://www.bitag.org/documents/BITAG_latency_explained.pdf>.

[Bufferbloat]
"Bufferbloat: Dark buffers in the Internet", n.d., <https://queue.acm.org/detail.cfm?id=2071893>.

[CSGO] Xu, X., Liu, S., and M. Claypool, "The Effects of Network Latency on Counter-strike: Global Offensive Players", IEEE, 2022 14th International Conference on Quality of Multimedia Experience (QoMEX) pp. 1-6, DOI 10.1109/qomex55416.2022.9900915, September 2022, <https://doi.org/10.1109/qomex55416.2022.9900915>.

[G.1051] ITU-T, "Latency measurement and interactivity scoring under real application data traffic patterns", ITU-T G.1051, March 2023, <https://www.itu.int/rec/T-REC-G.1051>.

[Haeri22] "Mind Your Outcomes: The ホ撚SD Paradigm for Quality-Centric Systems Development and Its Application to a Blockchain Case Study", n.d., <https://www.mdpi.com/2073-431X/11/3/45>.

[I-D.ietf-opsawg-rfc5706bis]
Claise, B., Clarke, J., Farrel, A., Barguil, S., Pignataro, C., and R. Chen, "Guidelines for Considering Operations and Management in IETF Specifications", Work in Progress, Internet-Draft, draft-ietf-opsawg-rfc5706bis-01, 17 December 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-opsawg-rfc5706bis-01>.

[IRTT] "Isochronous Round-Trip Tester", n.d., <https://github.com/heistp/irtt>.

[ISO5725-1]
ISO, "Accuracy (trueness and precision) of measurement methods and results Part 1: General principles and definitions", ISO 5725-1:2023, July 2022, <https://www.iso.org/standard/69418.html>.

[JamKazam] Wilson, D., "What is Latency and Why does it matter?", n.d., <https://jamkazam.freshdesk.com/support/solutions/articles/66000122532-what-is-latency-why-does-it-matter-?>.

[Kelly] Kelly, F. P., "Networks of Queues", n.d., <https://www.cambridge.org/core/journals/advances-in-applied-probability/article/abs/networks-of-queues/38A1EA868A62B09C77A073BECA1A1B56>.

[P.10] ITU-T, "Vocabulary for performance, quality of service and quality of experience", ITU-T P.10/G.100, November 2017, <https://www.itu.int/rec/T-REC-P.10>.

[P.1401] ITU-T, "Methods, metrics and procedures for statistical evaluation, qualification and comparison of objective quality prediction models", ITU-T P.1401, January 2020, <https://www.itu.int/rec/T-REC-P.1401>.

[P.800] ITU-T, "Methods for subjective determination of transmission quality", ITU-T P.800, August 1996, <https://www.itu.int/rec/T-REC-P.800>.

[P.800.1] ITU-T, "Mean opinion score (MOS) terminology", ITU-T P.800.1, July 2016, <https://www.itu.int/rec/T-REC-P.800.1>.

[P.910] ITU-T, "Subjective video quality assessment methods for multimedia applications", ITU-T P.910, October 2023, <https://www.itu.int/rec/T-REC-P.910>.

[QoOAppQualityReqs]
テ⊠tensen, T., "Performance Measurement of Web Applications", n.d., <https://domos.ai/storage/U6TlxIlbcl1dQfcNhnCleziJWF23P5w0xWzOARh8-published.pdf>.

[QoOSimStudy]
Monclair, B. I. T., "Quality of Outcome Simulation Study", n.d., <https://github.com/getCUJO/qoosim>.

[QoOUserStudy]
Monclair, B. I. T., "Application Outcome Aware Root Cause Analysis", n.d., <https://domos.ai/storage/LaiW4tJQ2kj4OOTiZbnf48MbS22rQHcZQmCriih9-published.pdf>.

[RFC2681] Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, DOI 10.17487/RFC2681, September 1999, <https://www.rfc-editor.org/rfc/rfc2681>.

[RFC2764] Gleeson, B., Lin, A., Heinanen, J., Armitage, G., and A. Malis, "A Framework for IP Based Virtual Private Networks", RFC 2764, DOI 10.17487/RFC2764, February 2000, <https://www.rfc-editor.org/rfc/rfc2764>.

[RFC3393] Demichelis, C. and P. Chimento, "IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)", RFC 3393, DOI 10.17487/RFC3393, November 2002, <https://www.rfc-editor.org/rfc/rfc3393>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <https://www.rfc-editor.org/rfc/rfc5357>.

[RFC5481] Morton, A. and B. Claise, "Packet Delay Variation Applicability Statement", RFC 5481, DOI 10.17487/RFC5481, March 2009, <https://www.rfc-editor.org/rfc/rfc5481>.

[RFC6349] Constantine, B., Forget, G., Geib, R., and R. Schrage, "Framework for TCP Throughput Testing", RFC 6349, DOI 10.17487/RFC6349, August 2011, <https://www.rfc-editor.org/rfc/rfc6349>.

[RFC6673] Morton, A., "Round-Trip Packet Loss Metrics", RFC 6673, DOI 10.17487/RFC6673, August 2012, <https://www.rfc-editor.org/rfc/rfc6673>.

[RFC7679] Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Delay Metric for IP Performance Metrics (IPPM)", STD 81, RFC 7679, DOI 10.17487/RFC7679, January 2016, <https://www.rfc-editor.org/rfc/rfc7679>.

[RFC7680] Almes, G., Kalidindi, S., Zekauskas, M., and A. Morton, Ed., "A One-Way Loss Metric for IP Performance Metrics (IPPM)", STD 82, RFC 7680, DOI 10.17487/RFC7680, January 2016, <https://www.rfc-editor.org/rfc/rfc7680>.

[RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <https://www.rfc-editor.org/rfc/rfc7799>.

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/rfc/rfc7942>.

[RFC8033] Pan, R., Natarajan, P., Baker, F., and G. White, "Proportional Integral Controller Enhanced (PIE): A Lightweight Control Scheme to Address the Bufferbloat Problem", RFC 8033, DOI 10.17487/RFC8033, February 2017, <https://www.rfc-editor.org/rfc/rfc8033>.

[RFC8239] Avramov, L. and J. Rapp, "Data Center Benchmarking Methodology", RFC 8239, DOI 10.17487/RFC8239, August 2017, <https://www.rfc-editor.org/rfc/rfc8239>.

[RFC8290] Hoeiland-Joergensen, T., McKenney, P., Taht, D., Gettys, J., and E. Dumazet, "The Flow Queue CoDel Packet Scheduler and Active Queue Management Algorithm", RFC 8290, DOI 10.17487/RFC8290, January 2018, <https://www.rfc-editor.org/rfc/rfc8290>.

[RFC8517] Dolson, D., Ed., Snellman, J., Boucadair, M., Ed., and C. Jacquenet, "An Inventory of Transport-Centric Functions Provided by Middleboxes: An Operator Perspective", RFC 8517, DOI 10.17487/RFC8517, February 2019, <https://www.rfc-editor.org/rfc/rfc8517>.

[RFC8762] Mirsky, G., Jun, G., Nydell, H., and R. Foote, "Simple Two-Way Active Measurement Protocol", RFC 8762, DOI 10.17487/RFC8762, March 2020, <https://www.rfc-editor.org/rfc/rfc8762>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/rfc/rfc9000>.

[RFC9197] Brockners, F., Ed., Bhandari, S., Ed., and T. Mizrahi, Ed., "Data Fields for In Situ Operations, Administration, and Maintenance (IOAM)", RFC 9197, DOI 10.17487/RFC9197, May 2022, <https://www.rfc-editor.org/rfc/rfc9197>.

[RFC9312] Kテシhlewind, M. and B. Trammell, "Manageability of the QUIC Transport Protocol", RFC 9312, DOI 10.17487/RFC9312, September 2022, <https://www.rfc-editor.org/rfc/rfc9312>.

[RFC9318] Hardaker, W. and O. Shapira, "IAB Workshop Report: Measuring Network Quality for End-Users", RFC 9318, DOI 10.17487/RFC9318, October 2022, <https://www.rfc-editor.org/rfc/rfc9318>.

[RFC9341] Fioccola, G., Ed., Cociglio, M., Mirsky, G., Mizrahi, T., and T. Zhou, "Alternate-Marking Method", RFC 9341, DOI 10.17487/RFC9341, December 2022, <https://www.rfc-editor.org/rfc/rfc9341>.

[RFC9543] Farrel, A., Ed., Drake, J., Ed., Rokui, R., Homma, S., Makhijani, K., Contreras, L., and J. Tantsura, "A Framework for Network Slices in Networks Built from IETF Technologies", RFC 9543, DOI 10.17487/RFC9543, March 2024, <https://www.rfc-editor.org/rfc/rfc9543>.

[RFC9817] Kunze, I., Wehrle, K., Trossen, D., Montpetit, M., de Foy, X., Griffin, D., and M. Rio, "Use Cases for In-Network Computing", RFC 9817, DOI 10.17487/RFC9817, August 2025, <https://www.rfc-editor.org/rfc/rfc9817>.

[RPM] "Responsiveness under Working Conditions", July 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-ippm-responsiveness>.

[RRUL] "Real-time response under load test specification", n.d., <https://www.bufferbloat.net/projects/bloat/wiki/RRUL_Spec/>.

[XboxNetReqs] Microsoft, "Understanding your remote play setup test results", n.d., <https://support.xbox.com/en-US/help/hardware-network/connect-network/console-streaming-test-results>.

Acknowledgments

The authors would like to thank Will Hawkins, Stuart Cheshire, Jason Livingood, Olav Nedrelid, Greg Mirsky, Tommy Pauly, Marcus Ihlar, Tal Mizrahi, Ruediger Geib, Mehmet ナ樞シkrテシ Kuran, Michael Welzl, Kevin Smith, Luis Miguel Contreras Murillo, Guiseppe Fioccola, Neil Davies, Paul Aitken, Werner Robitza, and Alexander Raake for their feedback and input to this document.

Authors’ Addresses

Bjテクrn Ivar Teigen Monclair
CUJO AI
Gaustadallテゥen 21
0349
Norway
Email: bjorn.monclair@cujo.com

Magnus Olden
CUJO AI
Gaustadallテゥen 21
0349
Norway
Email: magnus.olden@cujo.com

Ike Kunze (editor)
CUJO AI
Gaustadallテゥen 21
0349
Norway
Email: ike.kunze@cujo.com