IP Performance Measurement
Internet-Draft
Intended status: Informational
Expires: 16 November 2026

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

15 May 2026

Quality of Outcome (QoO)
draft-ietf-ippm-qoo-10

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 users, application developers, and network operators.

Conceptually based on the Quality Attenuation metric, QoO provides a method for defining and evaluating application-specific, quality-focused network performance requirements to enable insights for network troubleshooting and 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

Copyright Notice

Table of Contents

1. Introduction

This document introduces the Quality of Outcome (QoO) network quality score and the corresponding QoO framework.

QoO scores convey how well applications are expected to perform on assessed networks, with higher scores indicating that applications are more likely to perform well. To that end, QoO scores express measured network conditions as a percentage on a linear scale bounded by two application-specific thresholds: one for unacceptable performance (0%) and one for optimal performance (100%). This allows network quality to be communicated in easily understood terms such as "This network provides 94% of optimal conditions for video conferencing (relative to the threshold for unacceptable performance)" while remaining objective and adaptable to different network quality needs.

The QoO framework defines guidelines for conducting network performance measurements, how stakeholders specify the quality-focused network performance requirements (regarding latency, packet loss, and throughput) at the two quality thresholds, and how the user-facing QoO score is calculated based on such performance requirements and network performance measurements.

This document and the QoO framework assume that it is sufficient to assess network quality in terms of a minimum required throughput, a set of latency percentiles, and packet loss ratios, with the expectation that these dimensions will ultimately also capture the effects of additional factors. Hence, similar to Quality Attenuation [TR-452.1], the QoO framework assesses the network state based on latency distributions and packet loss probabilities and additionally considers throughput. This design ensures spatial composability [RFC6049], enabling network operators to achieve fault isolation (Section 5.4.4 of [I-D.ietf-opsawg-rfc5706bis]), advanced root-cause analyses from within the network (Section 5.4.3 of [I-D.ietf-opsawg-rfc5706bis]), and network planning while supporting comprehensive end-to-end tests.

1.1. What’s In? What’s Out?

This document defines a minimum viable QoO framework consisting of:

* Guidelines for conducting network performance measurements (Section 3.1)

* Guidelines for specifying quality-focused network performance requirements (Section 3.2)

* Calculation formulas for computing QoO scores (Section 3.3)

The document also discusses operational considerations (Section 4) and known weaknesses and open questions (Section 5). The appendix provides additional context on fundamental design considerations for QoO (Appendix A), a comparison of QoO with existing quality metrics (Appendix B), and preliminary insights from a small-scale user testing campaign (Appendix C).

The document intentionally leaves certain aspects of the QoO framework unspecified to allow for broad applicability across different deployment contexts and to enable the gathering of operational experience that can inform future, more prescriptive documents. The following items are out of scope for this document and may be addressed in future work:

* 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

* What levels of precision are considered acceptable

1.2. 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 network 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 application-specific, quality-focused network performance requirements for optimal and unacceptable application performance, typically expressed as a percentage.

Optimal performance: A level of performance beyond which further improvements in network conditions do not result in perceptible improvements in application performance or user experience.

Requirements for Optimal Performance (ROP): The network performance characteristics at which an application achieves optimal performance. 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].

2. Background and Design Considerations

This section provides concise background information on Quality Attenuation (Section 2.1) and a short summary of key design considerations for QoO (Section 2.2) with further elaborations in Appendix A.

2.1. Background on Quality Attenuation

Quality Attenuation is defined in the Broadband Forum standard Quality of Experience Delivered (QED) [TR-452.1] and characterizes network quality based on measurements of latency distributions and packet loss probabilities.

Using latency distributions to measure network quality has been proposed by various researchers and practitioners (e.g., [Kelly], [RFC6049], and [RFC8239]). Quality Attenuation uses a latency distribution as the basis for an Improper Random Variable (IRV). The cumulative distribution function of the IRV captures the likelihood that a packet "completes" within any given time, e.g., that it is received at the destination when one-way latency is assessed. The IRV incorporates packet loss by treating lost packets as infinite (or too late to be of use, i.e., not arriving within an application-specific time threshold) latency, similar to the One-Way Loss Metric for IP Performance Metrics (IPPM) [RFC7680], which defines packet loss as packets that fail to arrive within a specified time threshold. The "intangible mass" of the IRV represents the probability that a packet never completes within any useful time (i.e., is lost or arrives too late).

Quality Attenuation enables spatial composition [RFC6049] of network segments as two distributions can be composed using convolution. Measurements from different network segments can be combined to derive end-to-end quality assessments, or end-to-end measurements can be decomposed to isolate the contribution of individual segments. This composability enables network operators to perform fault isolation and root cause analysis by identifying which portions of a network path contribute most to performance degradation.

2.2. QoO Design Considerations

Quality Attenuation provides a mathematically rigorous foundation for network quality assessment and it can capture the ability of a network to satisfy a variety of application needs. However, interpreting its raw distributional outputs and component decompositions can be difficult, especially for application developers and end-users who might be primarily interested in understanding whether specific applications will perform adequately.

The QoO framework is specifically designed to address this limitation by translating the results of the underlying network performance measurements, i.e., latency distributions and packet loss ratios, into intuitive percentage scores that directly relate the measured network conditions to application-specific network performance requirements in an understandable and unambiguous way. To that end, the design of the QoO framework is motivated by the needs of three distinct stakeholder groups -- end-users, application developers, and network operators -- and bridges the gap between the technical aspects of network performance and the practical needs of those who depend on it.

End-users need network quality metrics that are understandable and that relate as directly as possible to application performance, such as video smoothness, web page load times, or gaming responsiveness. The QoO framework addresses this need by basing the QoO score on objective QoS measurements while communicating network quality in intuitive terms, thereby creating a middle ground between QoS and QoE metrics and allowing end-users to understand if a network is a likely source of impairment for the performance of applications.

Application developers need the ability to express quality-focused network performance requirements for their applications across all relevant dimensions of network quality (latency, packet loss, throughput) in order to test or state their network requirements. The QoO framework addresses this need by enabling both simple and complex requirement specifications, accommodating developers with varying levels of networking expertise.

Network operators need tools for fault isolation, performance comparison, and bottleneck identification. The QoO framework addresses this need through its use of latency distributions and packet loss probabilities, whose spatial composability enables operators to measure network segments independently, combine results to understand end-to-end quality, or decompose measurements to isolate problem areas, enabling network analysis in general. Additionally, operators can use the underlying raw measurement results to derive Quality Attenuation measures if more fine-grained insight is needed (see Section 4.1).

Overall, the QoO framework design acknowledges that all stakeholders ultimately care about the performance of applications running over a network by

1. capturing network performance metrics that correlate with application performance as perceived by users,

2. supporting comparison against diverse application requirements, and

3. providing composability for spatial decomposition and root cause analysis.

Additional elaboration on these three core properties is provided in Appendix A.

3. The QoO Framework

The QoO framework builds on the conceptual foundation of Quality Attenuation [TR-452.1]. Similar to Quality Attenuation, QoO evaluates network conditions using latency distributions and packet loss probabilities under the assumption that other factors which could in principle be measured but are not explicitly captured by QoO will inevitably influence the observed latency and packet loss behavior, so that QoO indirectly accounts for their effects. The QoO framework additionally includes throughput, i.e., the available network capacity, as applications usually have minimum throughput requirements below which they do not work at all. The measured conditions are compared against application-specific, quality-focused network performance requirements. Latency requirements are specified along multiple dimensions (such as 90th or 95th latency percentiles) while packet loss requirements specify mean packet loss ratios. Both latency and packet loss requirements are specified at two thresholds:

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

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

For throughput, requirements specify only a global minimum required value.

When the measured network conditions fall between the defined thresholds for any of the assessed performance dimensions for latency and packet loss, the QoO framework calculates a score for each dimension by expressing the current network quality as a relative position (percentage) on the linear scale from 0 to 100 between the corresponding CPUP (0) and ROP (100) thresholds. The minimum score across all dimensions serves as the overall QoO score for the assessed network based on the rationale that the most degraded performance dimension is likely to determine the application's perceived quality. If the throughput requirement is not met, the QoO score is always 0.

The remainder of this section describes how network conditions can be measured (Section 3.1), how QoO defines application-specific network performance requirements (Section 3.2), and how QoO scores are calculated (Section 3.3) with an example provided in Section 3.4.

3.1. Measuring Network Performance

The QoO framework relies on information on the latency and packet loss conditions and the available capacity of the network to be assessed. 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]. Similar considerations apply to packet loss measurements while throughput measurements usually involve active testing.

In addition to measurement approaches standardized in the QED framework [TR-452.1], 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]

* On-path telemetry methods (IOAM [RFC9197], AltMark [RFC9341])

* Latency tests under loaded network conditions [I-D.ietf-ippm-responsiveness]

* Throughput tests with included latency measures [Throughputtest]

* DNS response latency measurements (Section 2.8 of [RFC8517], [RFC8912])

* Passive TCP / QUIC handshake measurements [TCPHandshake][RFC9312]

* Continuous passive TCP / QUIC measurements [TCPContinuous][RFC9312]

* Simulating or estimating real traffic [LatencyEstimation]

3.1.1. Measurement Considerations

The QoO framework does not mandate the use of specific measurement techniques, measurement configurations, or measurement conditions. For example, it is agnostic to traffic direction, does not prescribe specific conditions for sampling, such as fixed time intervals or defined network load levels, and it does not enforce a minimum sample count, even though distributions formed from small numbers of samples (e.g., 10) are clearly insufficient for statistical confidence despite being technically valid.

Instead, the chosen measurement methodology must be able to capture characteristics of applications to be studied with sufficient resolution as different applications and application classes fail under different network conditions. For example, downloads are generally more tolerant of latency than real-time applications. Video conferences are often sensitive to high 90th percentile latency and to the difference between the 90th and 99th percentiles. Online gaming typically has a low tolerance for high 99th percentile latency. Similar considerations apply for a variety of other factors. For example, applications usually require some minimum level of throughput and tolerate some maximum packet loss ratio. The intent of this underspecification is to balance rigor with practicality, recognizing that constraints vary across devices, applications, and deployment environments.

Another dimension to this is the modelling of full latency distributions, which 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, trading off composability, which is only possible

with distributions, for an easier deployment. 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 choice of measurement methodology also needs to account for network conditions and their variability. Idle-state measurements capture baseline characteristics unaffected by competing traffic, whereas measurements taken under load reflect conditions that are more likely to challenge application performance, such as elevated latencies and queuing. Both active and passive methods can capture either state, although with different degrees of control. Passive monitoring of production traffic usually reflects actual network load but may not always allow capturing heavily loaded conditions. Active measurements can target heavily loaded conditions by generating synthetic traffic equivalent to the application load alongside the probes but capturing the actual or idle network conditions may not be possible depending on the footprint of the measurement method.

Performing active measurements or generating artificial load can degrade the network under test or inadvertently enable denial-of-service (DoS) abuse [RFC2330]. Hence, corresponding methodology needs to be designed to avoid significant impact, e.g., by only permitting authenticated measurements (Section 6.2 of [RFC4656]) or including rate limits and other safeguards against self-induced congestion (Section 4.2 of [RFC9946]). Section 6.3 provides additional mitigation strategies, mostly focusing on DoS.

Internet forwarding paths can also shift on a variety of timescales due to routing changes, load balancing, or traffic engineering, meaning a measurement reflects the network’s state only during the sampling period. Such factors need to be considered when conducting performance measurements. See Section 4.4 for a discussion of the operational implications, and Section 5.1 for the more severe case of volatile environments such as mobile cellular networks.

3.1.2. Reporting Measurement Results

This document defines a minimum viable framework, and often trades precision for simplicity to facilitate adoption and usability in many different contexts. To assess the corresponding loss of precision and account for the underspecification of the measurement methodology, each measurement must be accompanied by the following metadata in order to support reproducibility and enable confidence analysis, even across QoO deployments:

* Description of the measurement method, including:

- Standard name (if applicable) or reference
- Measurement class (Active, Passive, or Hybrid) [RFC7799]
- Protocol layer used for measurements (ICMP, TCP, UDP, ...)

* Measurement configuration, including:
  - Probe packet size (if applicable)
  - Traffic class of probed traffic
  - Sampling method, including but not limited to
    - 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)
* Description of the measured path, including:
  - Endpoints (source and destination)
  - Network segments traversed
  - Measurement points (if applicable)
  - Direction (two-way, one-way source-to-destination, one-way destination-to-source)
* Description of the measurement duration, including:
  - Timestamp of first sample (e.g., in the format used in TWAMP [RFC5357][RFC8877])
  - Total duration of the sampling period (in milliseconds)
  - Number of samples collected

These metadata elements are required for interpreting the precision and reliability of the measurements. As demonstrated in [QoOSimStudy] and discussed in Section 4.8, 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, which can also be affected by path variation across multiple protocol layers (see Section 4.4)

* 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.

3.2. Describing Network Performance Requirements

The goal of the QoO framework is to establish a quantifiable distance between unacceptable and optimal network conditions, thereby enabling an objective assessment of relative quality, i.e., how close some measured network conditions are to the optimal conditions specified by corresponding requirements. Matching the scope of the network performance measurements, corresponding network performance requirement specifications cover three dimensions: latency, expressed as a set of percentile-latency tuples; packet loss, expressed as a single ratio; and throughput, expressed as a minimum required value. For latency and packet loss, these specifications define both the Requirements for Optimal Performance (ROP), and the Conditions at the Point of Unacceptable Performance (CPUP). There is only one global minimum throughput requirement as insufficient network capacity may give unacceptable application outcomes without necessarily inducing a lot of latency or packet loss.

Figure 1 illustrates one example requirement specification. For optimal performance, the example application requires that 99% of packets need to arrive within 100ms and 99.9% within 200ms while tolerating at most 0.1% packet loss (ROP). The perceived service becomes unacceptable if 99% of packets have not arrived within 200ms, or 99.9% within 300ms, or if there is more than 2% packet loss (CPUP). The underlying minimum throughput requirement is 4Mbps.

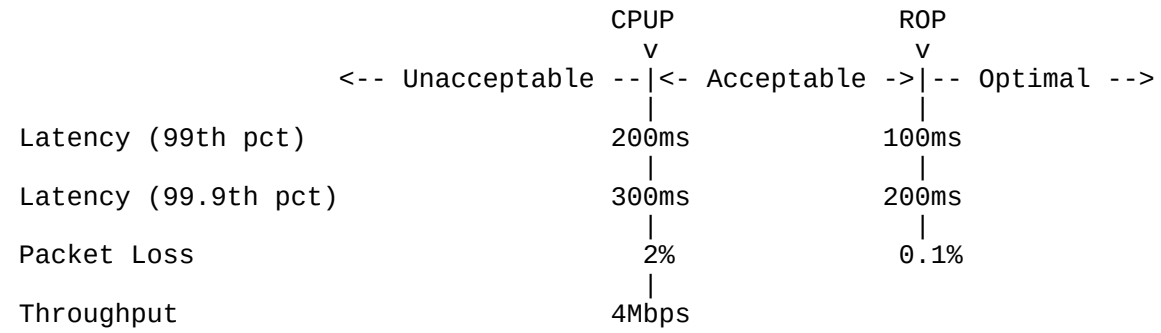

Figure 1: Example Network Performance Requirement Specification

3.2.1. Specification Guidelines

The QoO framework provides the general structure for network performance requirement specifications, enabling stakeholders to specify 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.

The QoO framework does not mandate the use of specific latency percentiles and it does not define standardized network performance requirement specifications for specific applications or application classes. Packet loss and throughput requirements can be arbitrary non-negative values while latency requirements are specified as sets of non-negative percentile-latency tuples. The set of included percentiles can be minimal (e.g., 100% within 200ms) or extended as needed and different percentiles may be used to characterize different applications.

For ease of operation, this document does mandate that latency percentiles specified in network performance requirement specifications must match the information available in the network performance measurements. This means that when the measurements report full latency distributions, requirements can use arbitrary percentiles. If the simplified latency reporting described in Section 3.1.1 is used, the requirement specification must use percentiles that are included in the reported measurements, i.e., one or more of the [0th, 10th, 25th, 50th, 75th, 90th, 95th, 99th, 99.9th, 100th] percentiles if the [BITAG] recommendation is followed.

For simplicity, the document further mandates that latency percentiles used in the ROP must also be used in the CPUP, and vice versa. For example, if the CPUP uses the 99.9th percentile then the ROP must also include the 99.9th percentile, and if the ROP uses the 99th percentile then the CPUP must also include the 99th percentile.

Finally, the network performance requirement specification must specify if the requirements are one-way or two-way. If the requirement is one-way, the direction between the endpoints of the assessed path, i.e., source-to-destination or destination-to-source, must be specified (see Section 3.1.2). In case of a two-way requirement, a decomposition into source-to-destination and destination-to-source components may be specified.

3.2.2. Creating a Network Performance Requirement Specification

This document does not define a standardized approach for creating quality-focused network performance requirement specifications. Instead, this section provides general considerations for deriving requirement specifications.

Deriving consistent and reproducible thresholds for ROP and CPUP specifications requires standardized testing conditions. Application developers should therefore publish their testing methodologies, including the network conditions, hardware configurations, and measurement procedures used to establish these thresholds, so that results can be independently validated and compared across implementations. 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.

To illustrate the large design-space for such testing, Section 4.9 sketches an arguably subjective approach to identifying thresholds for ROP and CPUP specifications, which should not be used in deployments due to its subjectiveness. Future documents may define new methods for deriving appropriate network performance requirements for QoO and could also recommend a fixed set of latency percentiles to be used, either for all applications or on a per-application / per-application-class basis to make QoO measurements between different networks or providers more comparable.

3.3. Calculating QoO

The QoO score compares network performance measurements to application-specific, quality-focused network performance requirements by assessing how close the measured network performance is to the network conditions needed for optimal application performance. The QoO score reflects the directionality (one-way or two-way) used in the measurements and the requirements; all need to use the same directionality and, for one-way assessments, the same

direction (source-to-destination or destination-to-source). There are three key scenarios:

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

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

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

3.3.1. Overall QoO Calculation

The overall QoO score is the minimum of a latency (QoO_latency), a packet loss (QoO_loss), and a throughput (QoO_throughput) score:

QoO = min(QoO_latency, QoO_loss, QoO_throughput)

with QoO_latency, QoO_loss, and QoO_throughput as defined in the following and illustrated in Figure 2.

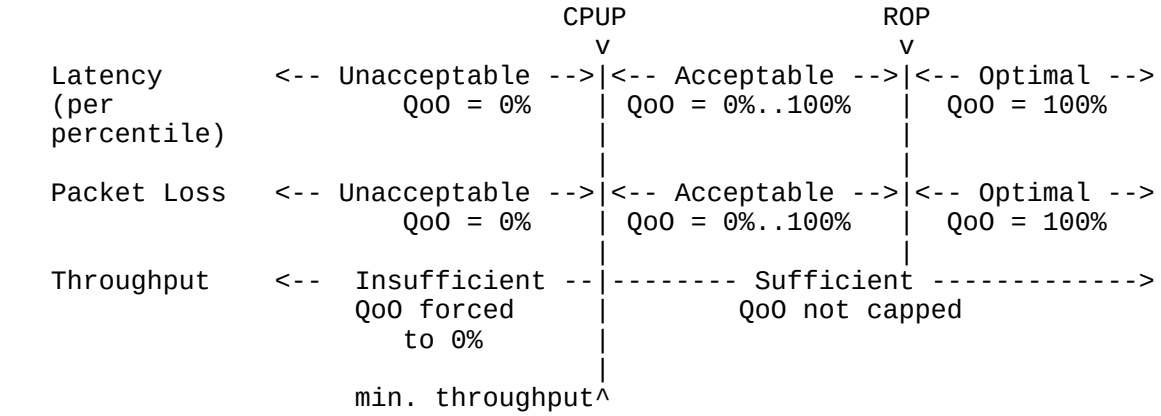

Figure 2: The three dimensions of QoO.

3.3.2. Latency Component

The QoO latency score is based on linear interpolations of the latency values at all latency percentiles defined in ROP / CPUP and represents the minimum value for all percentiles:

```
for i in latency_percentiles:
  m = 1 - ((ML[i] - ROP[i]) / (CPUP[i] - ROP[i]))
  metrics[i] = clamp(0, m, 1)
QoO_latency = find_min(metrics) * 100
```

Where:

* latency_percentiles are the latency percentiles contained in the ROP and CPUP definitions.

* ML[i] is the measured latency at percentile latency_percentiles[i].

* ROP[i] is the latency indicated in ROP at percentile latency_percentiles[i].

* CPUP[i] is the latency indicated in CPUP at percentile latency_percentiles[i].

3.3.3. 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 measured mean packet loss ratio (MLoss) and the packet loss thresholds defined in the ROP and CPUP:

```
m = 1 - ((M_Loss - ROP_Loss) / (CPUP_Loss - ROP_Loss))
QoO_loss = clamp(0, m, 1) * 100
```

Where:

* M_Loss is the measured mean packet loss ratio.

* ROP_Loss is the packet loss ratio indicated in ROP.

* CPUP_Loss is the packet loss ratio indicated in CPUP.

3.3.4. Throughput Component

In contrast to the latency and packet loss components, throughput uses a binary threshold:

```
QoO_throughput = (M_Throughput >= R_Throughput) ? 100 : 0
```

Where:

* M_Throughput is the measured throughput.

* R_Throughput is the minimum required throughput.

3.4. Example

The following example illustrates the QoO calculations.

Example requirements:

* Minimum throughput: 4Mbps

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

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

Example measured conditions:

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

* Measured mean packet loss ratio: 2%

* Measured throughput: 28Mbps

Latency component:

```
m1 = 1 - ((350ms - 200ms) / (500ms - 200ms)) = 1 - 0.5 = 0.5
m2 = 1 - ((375ms - 300ms) / (600ms - 300ms)) = 1 - 0.25 = 0.75
metrics = [clamp(0, m1, 1), clamp(0, m2, 1)] = [0.5, 0.75]
QoO_latency = find_min(metrics) * 100 = 50
```

Packet loss component:

```
m = 1 - ((2% - 1%) / (5% - 1%)) = 1 - 0.25 = 0.75
QoO_loss = clamp(0, m, 1) * 100 = 75
```

Throughput component:

```
28Mbps > 4Mbps
-> QoO_throughput = 100
```

Overall QoO score:

```
QoO = min(QoO_latency, QoO_loss, QoO_throughput) = min(50, 75, 100) = 50
```

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 conditions and considering latency, packet loss, and throughput. The score implies that the latency impact dominates the packet loss and throughput impacts and that the network overall provides conditions at the midway point of the performance range.

4. Operational Considerations

Aiming to ensure broad and easy applicability of the QoO framework across diverse use cases, the document imposes only a few strict mandates. Instead, this section provides general guidance concerning the operation of the QoO framework based on intuitions and assumptions that guided the development of the framework. Future documents are expected to capture and refine best practices once more operational experience has been gathered.

Some preliminary insights from a small-scale user testing campaign are provided in Appendix C. More comprehensive and large-scale testing are needed to assess the QoO framework.

4.1. QoO in the Quality Assessment Landscape

QoS, QoO, and QoE, as well as Quality Attenuation occupy different positions on a spectrum from raw network characterization to subjective user experience. QoS characterizes raw network behavior (latency, packet loss, throughput), usually without direct reference to any particular application or user. QoE, on the other hand, focuses on the subjective user perception directly.

QoO, by design, measures network service quality, not subjective user experience. However, as QoO scores are anchored to application-defined thresholds, they are expected to correlate with QoE metrics, such as Mean Opinion Score (MOS) [P.800.1], positioning QoO between QoS and QoE. The QoO framework itself does not define where QoO scores fall on this spectrum. Instead, the exact position primarily depends 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 closely approximate QoE metrics, while still maintaining the objectivity and composability benefits of QoS metrics.

Quality Attenuation is complementary to QoO in that it also aims to provide QoE-oriented QoS assessments. Both draw on the same latency distributions and packet loss probabilities, but differ in how those measurements are transformed: Quality Attenuation preserves the full distributional detail needed for convolution and per-hop decomposition, while QoO trades that detail for an application-

anchored score that is immediately actionable. Since both rely on the same underlying data, switching between Quality Attenuation and QoO requires no additional measurements, so operators can use QoO to produce a score that is immediately meaningful to all stakeholders and Quality Attenuation if they need more detailed root-cause analysis, capacity planning, or segment comparisons.

4.2. Composability, Flexibility, and Use Cases

One of the key strengths of the QoO framework is the mathematical composability of the underlying latency distributions and packet loss probabilities (see Appendix B.10), which allows measurements from different network segments to be combined or decomposed to isolate per-segment contributions. The composability also enables flexible deployment scopes as QoO scores may be computed for the complete end-to-end path (e.g., from application clients to servers), or focused on specific network segments, such as the customer-facing access network, intermediate transit networks, or server-side infrastructure. The network performance requirement specifications provide another dimension of flexibility as specifications can have different scopes, such as per-application, per application-type, or per-Service Level Agreement (SLA).

A holistic use of QoO with a fine-grained attribution of per-segment contributions requires sharing the measured distributions and probabilities for the involved segments among all relevant stakeholders, which can be challenging across different operators or networks. However, even without sharing raw data across all networks of an end-to-end path, QoO remains valuable for analyzing and troubleshooting individual network segments. Operators can use QoO to assess specific segments within their own networks, and end-users can gain insights into their own connectivity as long as their network providers support QoO. Hence, QoO is well-suited for incremental deployment (Section 2.1.2 of [RFC5218]).

4.3. Aligning Measurements with Service Levels

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 QoS 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.

4.4. Path Stability and Temporal Validity

Even when measurements are correctly aligned with the intended service delivery level, network behavior can vary within that service level over time.

Network conditions along a given path can fluctuate with varying traffic load: congestion, for example, can cause latency to increase and packet loss to rise transiently. The multi-percentile representation of latency in the QoO requirements specifications naturally captures such fluctuations within a measurement window, although the shape of the distribution depends on the load conditions present during that window.

Beyond load-driven fluctuation, forwarding paths themselves can also shift on a variety of timescales: routing changes, load balancing decisions, and traffic engineering policies may cause packets or flows to traverse different physical paths, each with potentially different latency, loss, and capacity characteristics. Load balancing, such as Equal-Cost Multi-Path (ECMP), can even mean that measurements represent a mixture of the characteristics across all active routes rather than those of a single coherent path.

Together, congestion-driven variation and path diversity mean that a QoO assessment captures a snapshot of network behavior during the sampling period, and conditions may differ significantly at other times. Operators should therefore repeat measurements periodically, and interpret individual QoO scores in light of when and under what load conditions they were obtained. Implementers also need to decide whether measurement traffic is steered consistently (e.g., by tuning flow tuples to follow specific ECMP paths) or deliberately varied to sample full path diversity, and document which approach was used in the measurement metadata.

These challenges are even more pronounced for mobile cellular networks, where path and capacity can change by an order of magnitude within seconds (see also Section 5.1).

4.5. Multipath Protocols

A related challenge arises when the application itself uses a transport protocol that exploits multiple paths concurrently, such as Multipath TCP [RFC8684] or Multipath QUIC [I-D.ietf-quic-multipath]. In such cases, the scheduling of traffic across paths is application-dependent and governed by many possible policies, including preferring a specific path, minimizing latency, maximizing aggregate capacity, distributing traffic equally, or minimizing path churn.

Single-flow measurements necessarily only follow one of the available paths at a time and may therefore fail to represent the full distribution of latency and loss conditions that the application actually experiences across its concurrent paths. Measuring across multiple flow tuples can provide a more comprehensive view, but the correct weighting of per-path measurements cannot be determined without knowing how the application distributes its traffic.

Passive monitoring of the actual application traffic is the most reliable approach for capturing the effect of multipath scheduling, as it directly observes which paths and proportions the application uses, but it may require visibility at multiple vantage points and may not always be feasible.

Even when the scheduling policy and per-path conditions are known, aggregating per-path scores into a single application-level QoO score requires further assumptions about how each path’s conditions affect the overall experience. A simplified, policy-agnostic approach is to use the worst score of available paths as the overall score. This approach is conservative and does not depend on assumptions about traffic distribution, although it may underestimate overall quality when the application avoids the worst path.

As with path diversity and load-driven variation, a QoO score reflects only the conditions observable on the measured path subset during the sampling period.

4.6. 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 available network capacity 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 quality.

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.

4.7. Continuous Measurements

The QoO framework does not define measurement periods: it can be applied to measurements taken over a specific, bounded interval (e.g., when conducting a scheduled test run) as well as to continuous measurements collected from live traffic over an extended period. Deployments may use either mode depending on the use case and available infrastructure [RFC6703].

When measurements are collected continuously, implementations must decide how to window or aggregate samples into the latency distribution and packet loss estimate used to compute a QoO score. Several approaches are possible, and each involves trade-offs:

Fixed time windows (e.g., last hour, last day, or last week) are simple to implement and compare across operators. Longer windows smooth out short-term anomalies but may obscure recent degradation; shorter windows are more responsive but less stable.

Rolling or sliding windows of the most recent N samples or the most recent T seconds provide a continuously updated view of network quality, balancing responsiveness with stability.

Measurements can also be grouped around specific events, such as user sessions or application usage periods. This approach can improve relevance for end-user-facing scores but may introduce selection bias.

The choice of windowing strategy affects which percentiles are observed and therefore the resulting QoO score. Implementations should document the windowing strategy used alongside the reported QoO scores and the measurement approach to ensure results are interpretable and comparable. Standardization of specific windowing approaches is considered out of scope for this document and left for future work.

4.8. Sensitivity 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 overly pessimistic QoO scores.

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, the following guidelines for practical application are deduced:

* 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 ROP and CPUP thresholds so that the resulting QoO scores accurately reflect application performance.

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

4.9. A Subjective Approach to 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 due to its inherently subjective nature.

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.

5. Known Weaknesses and Open Questions

Network performance measurements can be interpreted in different ways to derive quality assessments. QoO does so by comparing measured conditions against application-specific, quality-focused network performance requirements to produce a percentage-based score, which introduces several artifacts whose significance may vary depending on the context. The following section discusses some known limitations. A general assumption underlying the framework is that factors not explicitly captured by QoO (such as temporal packet ordering, fine-grained throughput variations, or the full shape of the latency distribution) will inevitably influence the observed latency and packet loss behavior, so that QoO indirectly accounts for their effects.

5.1. Volatile Networks

Volatile networks pose a challenge for network quality prediction, with the level of assurance of the prediction likely to decrease as session duration increases [QoSPrediction]. Where the volatile network is a segment forming part of an Internet path then it will typically form the bottleneck of that path.

Radio access networks are prone to volatility: the available capacity for a given user device can change by an order of magnitude within seconds due to physical radio factors. This is especially the case

in mobile cellular networks ([CellularPredictability]) due to the increased transmission distance and the difficulty of adding new access points or repeaters (in comparison to e.g. Wi-Fi). Other factors in cellular volatility include whether the user device is at the edge of a radio cell or undergoing cell handover, the radio interference and fading from the local environment, and any switch between radio bearers with differing capacity and transmission-time intervals (e.g., 3GPP 4G and 5G). This volatility applies on both dowlink (cell tower to device) and uplink (device to cell tower), albeit uplink incurs an additional factor of the reduced transmit power available to a user's battery-powered device.

The above suggests a requirement for measuring network quality to and from an individual device connected via the radio access network, as that can account for the factors described above. How that facility is provisioned onto individual devices and how device-hosted applications can trigger a network quality query, is an open question.

5.2. 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.

5.3. 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.

5.4. 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 in the QoO framework 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).

5.5. Binary Throughput Threshold

Choosing a binary throughput 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.

5.6. 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.

6. 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. In addition to the considerations discussed below, implementers are also encouraged to consider the security considerations outlined in [RFC7594].

6.1. 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.

6.2. 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.

6.3. 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 [RFC2330].

To mitigate these risks, the following is recommended:

* Implement rate-limiting and access control for active measurement tools, e.g., similar to recommendations for the One-way Active Measurement Protocol [RFC4656].

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

* Monitor for abnormal measurement patterns that may indicate abuse.

6.4. 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.

7. Privacy Considerations

QoO measurements may involve collecting detailed performance data from end-user devices or applications, especially in the context of passive measurements [RFC2330]. 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)).

8. IANA Considerations

This document has no IANA actions.

9. 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".

9.1. 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

9.2. 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

10. References

10.1. 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>.

10.2. 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>.

[CellularPredictability]
Basit, O., Dinh, P., Khan, I., Kong, Z., Hu, Y., Koutsonikolas, D., Lee, M., and C. Liu, "On the Predictability of Fine-Grained Cellular Network Throughput Using Machine Learning Models", IEEE, 2024 IEEE 21st International Conference on Mobile Ad-Hoc and Smart Systems (MASS) pp. 47-56, DOI 10.1109/mass62177.2024.00018, September 2024, <https://doi.org/10.1109/mass62177.2024.00018>.

[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-ippm-responsiveness]
Paasch, C., Meyer, R., Cheshire, S., and W. Hawkins, "Responsiveness under Working Conditions", Work in Progress, Internet-Draft, draft-ietf-ippm-responsiveness-08, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-ippm-responsiveness-08>.

[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-04, 15 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-opsawg-rfc5706bis-04>.

[I-D.ietf-quic-multipath]
Liu, Y., Ma, Y., De Coninck, Q., Bonaventure, O., Huitema, C., and M. Kテシhlewind, "Managing multiple paths for a QUIC connection", Work in Progress, Internet-Draft, draft-ietf-quic-multipath-21, 17 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-quic-multipath-21>.

[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>.

[LatencyEstimation]
Li, C., Nasr-Esfahany, A., Zhao, K., Noorbakhsh, K., Goyal, P., Alizadeh, M., and T. Anderson, "m3: Accurate Flow-Level Performance Estimation using Machine Learning", ACM, Proceedings of the ACM SIGCOMM 2024 Conference pp. 813-827, DOI 10.1145/3651890.3672243, August 2024, <https://doi.org/10.1145/3651890.3672243>.

[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>.

[QoSPrediction] Moreno, N., Dsouza, F., Kassler, A., Pullem, F., Xu, B., Amend, M., and C. Choi, "QoS and Capacity Prediction for 5G Network Slicing", IEEE, 2025 21st International Conference on Network and Service Management (CNSM) pp. 1-5, DOI 10.23919/cnsm67658.2025.11297458, October 2025, <https://doi.org/10.23919/cnsm67658.2025.11297458>.

[RFC2330] Paxson, V., Almes, G., Mahdavi, J., and M. Mathis, "Framework for IP Performance Metrics", RFC 2330, DOI 10.17487/RFC2330, May 1998, <https://www.rfc-editor.org/rfc/rfc2330>.

[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>.

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, DOI 10.17487/RFC4656, September 2006, <https://www.rfc-editor.org/rfc/rfc4656>.

[RFC5218] Thaler, D. and B. Aboba, "What Makes for a Successful Protocol?", RFC 5218, DOI 10.17487/RFC5218, July 2008, <https://www.rfc-editor.org/rfc/rfc5218>.

[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>.

[RFC6703] Morton, A., Ramachandran, G., and G. Maguluri, "Reporting IP Network Performance Metrics: Different Points of View", RFC 6703, DOI 10.17487/RFC6703, August 2012, <https://www.rfc-editor.org/rfc/rfc6703>.

[RFC7594] Eardley, P., Morton, A., Bagnulo, M., Burbridge, T., Aitken, P., and A. Akhter, "A Framework for Large-Scale Measurement of Broadband Performance (LMAP)", RFC 7594, DOI 10.17487/RFC7594, September 2015, <https://www.rfc-editor.org/rfc/rfc7594>.

[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>.

[RFC8684] Ford, A., Raiciu, C., Handley, M., Bonaventure, O., and C. Paasch, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 8684, DOI 10.17487/RFC8684, March 2020, <https://www.rfc-editor.org/rfc/rfc8684>.

[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>.

[RFC8877] Mizrahi, T., Fabini, J., and A. Morton, "Guidelines for Defining Packet Timestamps", RFC 8877, DOI 10.17487/RFC8877, September 2020, <https://www.rfc-editor.org/rfc/rfc8877>.

[RFC8912] Morton, A., Bagnulo, M., Eardley, P., and K. D'Souza, "Initial Performance Metrics Registry Entries", RFC 8912, DOI 10.17487/RFC8912, November 2021, <https://www.rfc-editor.org/rfc/rfc8912>.

[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>.

[RFC9946] Morton, A., Ciavattone, L., and R. Geib, Ed., "The UDP Speed Test Protocol (UDPSTP) for One-Way IP Capacity Metric Measurement", RFC 9946, DOI 10.17487/RFC9946, April 2026, <https://www.rfc-editor.org/rfc/rfc9946>.

[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/>.

[TCPContinuous] Sengupta, S., Kim, H., and J. Rexford, "Continuous in-network round-trip time monitoring", ACM, Proceedings of the ACM SIGCOMM 2022 Conference pp. 473-485, DOI 10.1145/3544216.3544222, August 2022, <https://doi.org/10.1145/3544216.3544222>.

[TCPHandshake] Apostolaki, M., Singla, A., and L. Vanbever, "Performance-Driven Internet Path Selection", ACM, Proceedings of the ACM SIGCOMM Symposium on SDN Research (SOSR) pp. 41-53, DOI 10.1145/3482898.3483366, October 2021, <https://doi.org/10.1145/3482898.3483366>.

[Throughputtest] MacMillan, K., Mangla, T., Saxon, J., Marwell, N., and N. Feamster, "A Comparative Analysis of Ookla Speedtest and Measurement Labs Network Diagnostic Test (NDT7)", Association for Computing Machinery (ACM), Proceedings of the ACM on Measurement and Analysis of Computing Systems vol. 7, no. 1, pp. 1-26, DOI 10.1145/3579448, February 2023, <https://doi.org/10.1145/3579448>.

[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>.

Appendix A. QoO Framework Design Considerations

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.

Network capacity 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 available network capacity alone. [RPM], 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 the available network capacity. 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.

A.1. General 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 general requirements for such a framework while the sections therafter describe requirements from the perspective of each of the target groups: end-users (Appendix A.2), application developers (Appendix A.3), and network operators (Appendix A.4).

In general, all stakeholders ultimately care about the performance of applications running over a network. Application performance does not only depend on the available network capacity 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 4.3). 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 presents requirements from the perspective of each of the three target groups: end-users (Appendix A.2), application developers (Appendix A.3), and network operators (Appendix A.4).

A.2. Requirements from 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 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)

These numbers only serve as examples and their exact value depends on the specific application and the test methodology used to derive them, such that they are not to be interpreted as universally applicable (see also Section 3.2.2 and Section 4.9). Instead, additional standardization efforts are needed to derive more universally applicable thresholds for different classes of applications.

A.3. 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.

A.4. Requirements from 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.

Appendix B. Comparison To 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 in comparison to QoO.

Each metric is evaluated against the three criteria established in Appendix A.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
Quality of Outcome	Yes	Yes	Yes (Through Quality Attenuation)

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 spatial composition as described in Appendix A.4 and [RFC6049]: the ability to combine measurements from individual path segments to derive end-to-end properties, or to decompose end-to-end measurements to isolate per-segment contributions.

B.1. Throughput

Throughput relates to user-observable application outcomes as acceptable performance is impossible when throughput falls below an application's minimum requirement. Above that minimum threshold, the relationship weakens and additional capacity above a certain threshold will, at best, yield diminishing returns (and any returns are often due to reduced latency). While throughput can be compared to a variety of application requirements, it is not generally possible to conclude from sufficient throughput alone that an application will work well.

Throughput is not composable in the spatial sense. While the throughput of a composed path a-b-c equals the minimum of the two individual segment throughputs, the bottleneck segment cannot be identified from the composed value: if b-c is the bottleneck, then the throughput of a-b-c equals the throughput of b-c, and the higher capacity of segment a-b is not recoverable from the combined measurement.

B.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. Since this composition is additive, it is also invertible: knowing the end-to-end mean latency of a-b-c and the mean latency of one segment is sufficient to recover the mean latency of the other segment.

B.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 as the Nth percentile of a composed distribution cannot in general be derived from the Nth percentile of its constituent distributions without access to the full distributions.

B.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. This composition is also invertible for independent segments, enabling decomposition: knowing the end-to-end variance and the variance of one segment is sufficient to recover the variance of the other segment.

B.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.

B.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.

B.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.

B.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.

B.9. Quality Attenuation

Quality Attenuation is a network quality metric that combines dedicated latency distributions and packet loss probabilities 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 via convolution of the underlying distributions, which 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].

B.10. Quality of Outcome

Quality of Outcome (QoO) builds upon Quality Attenuation by adding application-specific, dual-threshold network performance requirements (ROP and CPUP) and translating the comparison between measured network conditions and these requirements into a percentage-based score. By incorporating latency distributions, packet loss ratios, and throughput measurements, QoO can assess how well a wide range of applications are expected to perform under given network conditions.

The underlying Quality Attenuation measurements used in QoO are mathematically composable via convolution [TR-452.1]. This composability extends to QoO in the sense that operators can measure individual network segments, compose the underlying Quality Attenuation distributions, and then compute QoO scores from the composed result. This composability requires the full distributional representation. When QoO score calculation is based only on scalar percentile summaries (see Section 3.1.1), this composability is not available.

Appendix C. Preliminary Insights From a Small-Scale User Testing Campaign

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.

Acknowledgments

The authors would like to thank Karl Magnus Kalvik, Olav Nedrelid, and Knut Joar Strテクmmen for their conceptual and technical contributions to the QoO framework.

The authors would like to thank Gavin Young, Brendan Black, Kevin Smith, Gino Dion, Mayur Sarode, Greg Mirsky, Wim De Ketelaere, Peter Thompson, and Neil Davies for their contributions to the initial version of this document.

The authors would like to thank Gorry Fairhurst, Jテカrg Ott, Paul Aitken, Mohamed Boucadair, Stuart Cheshire, Neil Davies, Guiseppe Fioccola, Ruediger Geib, Will Hawkins, Marcus Ihlar, Mehmet ナ楹シkrテシ Kuran, Paul Kyzivat, Jason Livingood, Greg Mirsky, Tal Mizrahi, Luis Miguel Contreras Murillo, Tommy Pauly, Alexander Raake, Werner Robitza, Kevin Smith, Martin Thomson, and Michael Welzl for their feedback and input to this document throughout the IETF process.

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