

# From Draft to Standard: An Expert-Validated Catalog of API Gateway Software Metrics

Eder dos Santos<sup>\*</sup>, Sandra Casas<sup>\*</sup>

<sup>\*</sup>Corresponding author: [esantos@uarg.unpa.edu.ar](mailto:esantos@uarg.unpa.edu.ar)

## Article info

Dataset link: <http://hdl.handle.net/11336/264879>

### Keywords:

Software quality  
Software measurement and metrics  
Internet software systems development  
Empirical and experimental studies in software engineering

Submitted: 17 Feb. 2026

Revised: 6 May 2026

Accepted: 6 May 2026

Available online: 12 May 2026

## Abstract

**Context:** In the fast-growing API Economy scenario, API Gateways became an essential tool for microservice reliability and scalability, yet lack standardized quality models and measures to assess these tools in an objective fashion.

**Objective:** This study quantitatively and qualitatively assessed the accuracy and relevance of a set of 59 metrics, gathered from prior analysis of 68 mainstream API Gateway software offerings.

**Method:** An expert judgment evaluation study was conducted with seven domain specialists ( $N = 7$ ) from both academia and industry, who assessed each metric's accuracy and relevance using 5-point Likert scales. A multi-statistical method was performed to assess ratings, inter-observer consistency and reliability.

**Results:** Although ratings were generally neutral to positive, statistical analysis revealed low consensus across experts. A subset of 24 core metrics with strong agreement and high scores was identified, alongside moderate- and low-agreement sets. These items cover key API management concerns, including traffic monitoring, error handling, latency, and resource usage.

**Conclusions:** By combining practical definitions, expert judgment, and a tiered classification mechanism, this work established a broad and comprehensive set of metrics that will help to bridge the gap between metric adoption in real-world platforms and their theoretical grounding in software quality models.

## 1. Introduction

### Problem statement

As organizations increasingly adopted microservices architectures in their software ecosystems, the role of API Gateways became pivotal in ensuring system performance, scalability, reliability, security and maintainability [1–3].

While existing literature addresses various aspects of API Management software quality such as performance, security, and usability [4,5], there is a notable absence of standardized, validated frameworks to assess the quality of API Gateways [6]. This gap impedes the development of consistent evaluation methodologies and benchmarks for API Gateway implementations.

## Study context

In contemporary software engineering, Application Programming Interfaces (API) serve as fundamental building blocks, enabling interoperability and integration across diverse systems and platforms [1, 7]. The proliferation of digital services has underscored the necessity for effective API management, with the API Gateway emerging as a critical component in orchestrating and securing API traffic [5].

The growing adoption of APIs has led to the proliferation of various API Gateway products. Current industry “state of the market” reports indicate that there are 65+ API Gateway offerings available nowadays<sup>1,2,3</sup>.

In such a competitive landscape, it is essential to understand how software quality can be defined, measured and assessed, and what these products provide to satisfy the stakeholders’ requirements. However, there is a fragmented research landscape characterized by technical implementations often lacking grounding in established software quality models, and a notable absence of standardized, validated frameworks for API Gateway quality assessment [6].

To address this gap comprehensively, our research has pursued a multi-phase strategy integrating multiple evidence sources:

1. Understanding the research landscape [6]: identified quality attributes, measurement and evaluation methods, and the absence of formal frameworks in API management research.
2. Understanding practitioners’ priorities [3]: a practitioner-oriented survey explored how professionals perceive API management quality requirements and mapped these to ISO/IEC 25010 quality attributes [8], revealing that functional suitability, performance efficiency, reliability, and security are consistently prioritized.
3. Capturing operational reality [9]: the analysis of 68 mainstream API Gateway platforms documented the operational metrics actually deployed in production systems.
4. Expert validation (current study): Rigorous evaluation of practice-based metrics by domain experts to establish a validated, standardized foundation.

## Research objective

The main goal of this work was to quantitatively and qualitatively validate the accuracy and relevance of a preliminary set of 59 API Gateway software metrics gathered from a previous analysis of current mainstream API Gateway software offerings, as detailed in [9]. The validation was carried out using an expert judgment approach, following the guidelines proposed in [10, 11].

## Contribution statement

This study contributes to the software engineering and API management literature by providing a validated and empirically grounded set of API Gateway quality metrics. Main contributions include:

- The empirical evaluation of accuracy and relevance of a large set of API Gateway software metrics, performed by a cohort of seven internationally recognized specialists from both academia and industry.

---

<sup>1</sup><https://www.gartner.com/en/documents/4829031>

<sup>2</sup><https://www.postman.com/state-of-api/>

<sup>3</sup><https://apilandscape.apiscene.io/>

- A structured statistical analysis of the agreement level among specialists, based on inter-observer reliability and internal consistency, according to [12].
- A tier-based classification scheme based on expert ratings and statistical agreement between the specialists.
- A subset of 24 core metrics with strong agreement and high scores that can support standardized quality assessment in the API management domain, which can be establish a ready-to-use artifact to measure, assess and improve API Gateway software quality.

By combining systematic artifact analysis, expert validation, and quantitative rigor, this research effort addresses the absence of standardized quality measures for API Gateways and offers a robust foundation for benchmarking, product evaluation, and integration with established software quality frameworks. Ultimately, it represents a significant step toward defining a tailored API Gateway software product quality model, serving as a foundation for more consistent assessment, informed decision-making, and future standardization efforts in the domain.

## Research significance

This work addresses a critical gap in API management research and practice at a pivotal moment in the field’s evolution. The significance of this work operates on multiple levels:

**Theoretical contribution:** By validating operational metrics through expert judgment and grounding them in established software quality frameworks, this research bridges the gap between ad-hoc industry practices and formal quality assessment methodologies. The resulting catalog provides the empirical foundation necessary for developing a comprehensive API Gateway quality model aligned with existing standards such as [13].

**Practical impact:** The 24 validated core metrics offer immediately deployable quality indicators for organizations managing API infrastructure. Unlike existing platform-specific metrics, this validated set provides a vendor-neutral, expert-endorsed foundation for benchmarking, monitoring, and evaluating API Gateway implementations across diverse organizational contexts.

**Methodological implication:** The four-step validation strategy – integrating systematic literature analysis, practitioner survey data, operational platform analysis, and expert judgment – establishes a rigorous methodological template for validating practice-derived software metrics. This approach demonstrates how industry practices can be systematically elevated to theoretically grounded standards.

**Timing and context:** With 65+ API Gateway products currently available and the API Economy projected to grow substantially, the absence of standardized quality assessment frameworks represents a significant impediment to informed decision-making, systematic improvement, and vendor comparison. This work proposes a validated measurement foundation essential for the field’s maturation from fragmented implementations to standardized best practices.

### 1.1. Paper structure

The remainder of this paper is organized as follows. Section 2 explains the background knowledge. Section 3 reviews related work on API management and API Gateway quality, providing context for the research. Section 4 outlines the research methodology, detailing the sampling strategy, survey design, and multi-statistical approach used to validate expert ratings. In Section 5, we present the key findings from expert ratings and statistical analysis,

and include a tier-based classification of API Gateway measures. Section 6 discusses the implications of these findings, while Section 7 addresses the potential threats to the validity of the study. Finally, Section 8 concludes the paper and offers directions for future research.

## 2. Background

### 2.1. API, API management and API Gateways

In recent years, the distribution models for information systems have been moving towards XaaS [14] paradigms. This shift entailed organizations providing their digital assets as services to customers [15]. In general, such services are supported by API that typically adhere to REST [16] principles.

API present both a business side and a technical side [17]: they determine how the organizations want to use their assets to deliver value within the organization and to external third parties, and they are “a technical answer to a business problem” that allow to implement a set of requirements that govern how applications can interact and exchange data. In this sense, the term “API Economy” [18] reflects the ongoing trend across industries towards innovative approaches to enhancing their business frameworks.

The API Economy scenario exerts additional pressure in developing, deploying, and maintaining information systems. Since APIs have gained a critical aspect [19], organizations need to proactively address the risks of failure by enhancing API Management [5, 17] capabilities and managing their APIs through API Management platforms.

API Management Platforms provide the basic capabilities to create, analyze, and manage APIs in a secure and scalable environment such as providing helpful documentation, controlling access to the API, as well as monitoring and analyzing its usage [5]. At the core of digital integration strategies, these platforms facilitate the creation, analysis, and governance of APIs within a secure and flexible framework, streamlining API management processes [1, 5, 20].

At the core of API Management products lies the API Gateway. The API Gateway acts as a centrally managed, API-oriented control service that encapsulates, exposes, secures, and manages back-end data and services as RESTful APIs, which provides a framework for establishing a facade on top of these services [21]. API Gateway’s main features may include security, traffic management, service and data mapping, caching, load balancing, error reporting, and performance monitoring, among others. API Gateways provide a broad set of metrics that enables the production of business-value reports and user-friendly dashboards aiming at helping to understand who uses available APIs and how they are utilized [5].

### 2.2. API management capabilities and practices

In general, API Management Platform features are generally characterized in the form of practices and capabilities.

According to [5], API Management capabilities represent the features that enables developers to create, analyze, audit, log, analyze, and manage the lifecycle of API in a secure, reliable and flexible environment. Suggested capabilities are: developer enablement for API; API lifecycle management; secure, reliable and flexible communications; and API auditing, logging and analytics.

Additionally, a systematic mapping study (SMS) by [22] identified 114 API Management practices and 39 capabilities. In their framework, a practice was defined as “any activity with the explicit goal of improving, encouraging, or managing the usage of an API”, while a capability referred to “the ability to achieve a specific API Management objective through the coordinated execution of two or more interrelated practices”. Commonly reported capabilities included authentication, monitoring, security, analytics, catalog and documentation, API publication and deployment, monetization, and version management.

### 2.3. Software product quality and metrics

There is a wide range of standards applicable to various areas of knowledge in Software Engineering, with a common objective related to quality. In general, software quality can be appreciated from two perspectives: the first is based on the ISO 9001 standard [23], which focuses on quality policy across the entire organization. The terminology of this standard may be unfamiliar to software professionals, and quality management auditors may not be familiar with the software engineering vocabulary [10, 24]; the second seeks to decompose software quality into a set of desired characteristics of a software product and select measures and evaluation methods to determine if the desired level of those characteristics has been achieved. From this perspective, over time, various quality models have been developed, such as FURPS, Boehm, Dromey, and McCall, among others [25].

The terminology for quality attributes of software products differs from one model to another; each model has a different number of hierarchical levels, as well as a distinct number of characteristics [25]. A milestone in the definition of quality standards was the publication of the ISO/IEC 9126:1991 standard [26], which was subsequently replaced by the ISO/IEC 9126:2001 [27] and ISO/IEC 14598:1999 [28] standards. Currently, the international standards series ISO/IEC 25000 [29], known as SQuaRE (Software Product Quality Requirements and Evaluation), unifies and replaces the ISO/IEC 9126 and ISO/IEC 14598 series. SQuaRE is considered the current de facto standard series for software product quality, addressing aspects related to quality management, requirements, software and data quality models, as well as measurement and evaluation processes for quality.

The ISO/IEC SQuaRE series is structured into several divisions. Within the scope of this paper, the relevant divisions are outlined as follows:

The Quality Model Division (2501n) provides models for system and software product quality, quality in use, and data quality. The general software product quality model is described in [13].

The Quality Measurement Division (2502n) includes a system and software product quality measurement reference model, definitions of quality measures, and practical guidance for their implementation. The framework for quality measurement is illustrated in [30].

The Quality Evaluation Division (2504n) offers requirements, recommendations, and guidelines for system and software product evaluation. An overview of the evaluation process is presented in [31].

The concept of measurement was introduced by [32] as “the assignment of numerals to objects or events according to rules”. This foundational definition highlights that measurement is not merely about assigning numbers, but doing so systematically, in a way that reflects the nature of the attributes being observed. The scale of measurement used – such as nominal, ordinal, interval, or ratio – determines the types of analysis or comparisons that are meaningful.

In software engineering, measurement is essential for understanding, evaluating, and improving software products and processes [33]. Accordingly, metrics are proposed to quantify specific quality attributes and support decision-making.

The representational theory of measurement formalizes the relationship between observed attributes and the numerical values used to represent them. As noted by several authors [12, 33, 34], all measurement begins with the observer's intuition and subjective interpretation, which must then be systematically refined into reliable, objective measures.

## 2.4. Expert judgment in software engineering

In software engineering, judgment studies leverage structured expert evaluations to assess constructs that are difficult to measure directly, by gathering empirical data from a group of participants who are asked to judge or rate behaviors, to discuss a given topic of interest. As stated by [10], “the Judgment Study is a compromise between achieving a high level of precision and generalizability of findings”.

Judgment studies rely on systematic sampling and involve experts, appropriately informed to respond to a certain question. These studies commonly employ formal elicitation methods – like Delphi [35] panels, focus groups [36], or evaluation studies [10] – combined with inter-rater agreement metrics [12] to ensure reliability and validation of subjective assessments.

According to [33], validating a software metric is the process of ensuring that the measure is a proper characterization of a claimed attribute by showing that its representation condition is satisfied.

From a theoretical standpoint, the accuracy of a measure depends on the definition of the measurement, as stated by [33]. However, measures may not be likely to be accurate, either because the definition is imprecise or ambiguous, or because it depends on the subjective judgment of the person doing the measuring.

## 3. Related work

The quality of API management has been increasingly examined from both academic and industry perspectives, yet it remains loosely defined and lacking formal standardization. This section critically examines existing research across four dimensions: the API management quality research landscape, quality modeling approaches, validation methodologies in the API domain, and the specific metric validation gap that motivates this study.

### 3.1. API management quality research landscape

A recent SMS [6] identified a fragmented landscape of research problems and evaluation methods in the API management domain. The study analyzed contributions across multiple venues and revealed that most work is categorized as technical implementations or specific solutions [37], often without grounding in established software quality models. Among the most commonly addressed quality attributes were performance efficiency, reliability, functional suitability, and security. The study also highlighted that research generally did not explicitly utilize formal quality models. However, it underscored that common meta-model elements, such as features and metrics, were widely employed as evaluation mechanisms.

This fragmentation reflects a broader pattern in software engineering research where, as noted by [37], solution-oriented contributions often provide limited generalizability without grounding in theoretical frameworks. The API management domain exemplifies this pattern – considerable implementation innovation exists, alongside sparse validation of underlying quality constructs. While researchers and practitioners recognize that performance, reliability, and security matter, the absence of standardized measurement frameworks prevents systematic comparison, benchmarking, or cumulative knowledge building. This gap between recognized importance and measurement standardization represents a critical impediment to the field’s maturation.

### 3.2. Quality modeling approaches for APIs

Several researchers have attempted to formalize API quality assessment through structured models. [38] proposed an API management maturity model (API-m-FAMM) from an organizational perspective, establishing a set of focus areas for capability development across different maturity levels. The model provides a framework for organizations to assess and improve their API management practices systematically. [39] introduced a web API quality meta-model and defined a quality model with three attributes – functionality, reliability, and usability – demonstrating quantitative evaluation in real-world applications through a case study approach. More recently, [40] presented a catalog of metrics to assess API usability, developed using the Goal-Question-Metric (GQM) [41] approach, which provides a structured method for deriving metrics from quality goals.

These contributions advance quality conceptualization in important ways. The maturity model [38] offers organizational guidance for capability improvement, while the meta-model [39] demonstrates feasibility of quantitative quality assessment. The GQM-based usability metrics [40] provide rigorous derivation from explicit quality goals. However, these works share common limitations relevant to the current study’s focus. First, the maturity model addresses organizational capabilities rather than product quality metrics – it guides *how* to manage APIs but not *what* to measure in API products. Second, the meta-model’s three-attribute scope provides limited coverage of the comprehensive quality concerns identified in [6], particularly overlooking operational aspects like resource utilization and system health. Third, while the usability metrics are rigorously derived, they address only one quality characteristic. Critically, none of these works specifically target API Gateway quality as a distinct domain, nor do they validate metrics against operational implementations in production systems. The current study addresses these gaps by focusing specifically on API Gateway infrastructure and validating metrics already deployed in real-world platforms.

### 3.3. Validation methodologies in API research

The use of surveys and empirical studies has been employed to characterize API practices and validate artifacts in the domain. [42] presented the outcomes of the first survey specifically targeting API developers in Argentina, identifying usage patterns and challenges faced by software developers when consuming web APIs within that regional context. This work provided valuable insights into practitioner experiences and common pain points in API consumption.

Building on artifact validation approaches, [40] employed surveys directed to both API consumers and developers to evaluate the usefulness of their proposed usability metrics

set. In a related study, [43] conducted a survey with 36 practitioners – including both experts and non-experts – to assess perceptions on the influence of various usability factors. Interestingly, their findings indicated that ratings did not vary significantly between experts and non-experts in the context of usability evaluation.

The finding that expert and non-expert ratings converged in usability assessment [43] provides an important methodological insight that contrasts with the current study’s design. This convergence likely reflects the intuitive nature of usability concerns – factors like ease of use and clarity are readily assessed by diverse users. However, evaluating technical quality metrics requires different expertise: assessing metric *accuracy* (whether definitions are precise and unambiguous) and operational *feasibility* (whether metrics can be implemented reliably) demands deep domain knowledge. Our deliberate focus on recognized experts thus reflects distinct validation objectives appropriate for technical metric evaluation rather than user experience assessment.

### 3.4. Practitioner perspectives and requirements

Complementing academic research, practitioner-oriented studies have explored quality requirements from the perspective of those implementing and operating API management systems. [3] conducted a survey exploring how professionals in the field perceive the quality requirements of API management systems and mapped these traits to quality attributes established in the ISO/IEC 25010 standard [8]. Functional suitability, performance efficiency, reliability, and security were consistently highlighted as the most critical characteristics – reinforcing the overlap between academic research focus and practitioner needs, while simultaneously exposing the absence of systematic quality frameworks for assessment.

The remarkable convergence between academic research emphasis [6] and practitioner priorities [3] on performance, reliability, security, and functional suitability provides strong evidence for these attributes’ centrality to API Gateway quality. This alignment validates the relevance of academic research directions. However, this consensus also reveals a critical gap: while agreement exists on *what quality attributes matter*, no standardized, validated frameworks exist for systematically *measuring* these concerns in API Gateway contexts. Practitioners must either rely on vendor-specific metrics that vary across platforms or develop ad-hoc measurement approaches that lack empirical validation. This measurement gap motivated our previous work in cataloging operational metrics [9] and drives the validation study reported in this paper.

### 3.5. The metric validation gap

Despite growing recognition of API management quality’s importance, as evidenced by the studies reviewed above, standardized and empirically validated metrics for API Gateway assessment remain conspicuously absent. Existing research has identified quality attributes that matter [6], characterized practitioner priorities [3], and proposed quality models [38,39], yet none have established validated metrics specifically for API Gateway infrastructure.

Our previous work [9] began addressing this gap by systematically cataloging 59 metrics from 68 commercial and open-source API Gateway platforms. The catalog was developed following Design Science Research (DSR) principles [44] and organized according to established API management capability frameworks [5,22]. Quantitative analysis revealed that most metrics addressed latency, response time, error tracking, and traffic monitoring – operational concerns aligned with the practitioner perspectives discussed in [3]. This

work documented operational reality by capturing what practitioners actually measure in production systems. However, the catalog lacked external validation, particularly through systematic expert evaluation.

As emphasized by measurement theory [33], validating a software metric requires demonstrating that it properly characterizes the claimed attribute – ensuring that the representation condition is satisfied. Practice-derived metrics, while demonstrating operational feasibility, require expert validation to establish theoretical grounding and confirm accurate representation of quality constructs. Without such validation, the risk remains that operational metrics may be measuring convenient proxies rather than genuine quality attributes, or that metric definitions may be ambiguous or inconsistent across implementations. This validation gap – the absence of systematic expert assessment of practice-derived API Gateway metrics – directly motivated the current study.

### 3.6. Positioning the current work

This study uniquely addresses the identified gaps by integrating multiple evidence sources and validation approaches. Our work differs from existing research in several critical dimensions:

**Domain specificity:** Unlike general API quality models that address APIs broadly, this work specifically targets API Gateway infrastructure – the critical component that orchestrates, secures, and monitors API traffic in microservices architectures. API Gateways present distinct quality concerns related to traffic routing, load balancing, security enforcement, and operational monitoring that differ from application-level API quality.

**Practice-grounded validation:** Rather than proposing theoretically derived metrics, we validate measures already deployed in 68 production systems [9], ensuring practical feasibility and operational relevance. This approach acknowledges that metrics demonstrating real-world adoption likely address genuine operational concerns, though they require validation to establish theoretical soundness.

**Rigorous expert judgment methodology:** We apply established validation methodologies from software engineering research [10, 12], including purposive expert sampling, structured evaluation instruments, and multi-method statistical analysis of inter-rater agreement. This rigor distinguishes our validation effort from informal consensus or anecdotal acceptance.

**Multi-source triangulation:** The current validation study represents the fourth phase of a comprehensive research program that integrated: (1) systematic mapping of API management research [6], (2) practitioner survey of quality requirements [3], (3) systematic analysis of operational metrics across platforms [9], and (4) expert validation (current study). This four-step approach provides robust triangulation across literature, practitioner perspectives, operational reality, and expert judgment.

**Actionable classification framework:** Beyond binary validation (accept/reject), we provide a tier-based classification that translates mixed expert assessments into practical guidance. This framework identifies metrics ready for immediate adoption (Tier 1), those requiring refinement (Tier 2), and those needing substantial revision (Tier 3), supporting evidence-based decision-making by researchers and practitioners.

By systematically bridging the gap between operational practice and theoretical validation, this research establishes the empirical foundation necessary for developing standardized API Gateway quality assessment frameworks. The validated metric set addresses the fragmentation identified in [6], operationalizes the quality requirements articulated in [3],

and elevates the practice-derived metrics documented in [9] to theoretically grounded, expert-validated standards.

## 4. Method

### Research questions

The overarching goal of this study is to establish a validated foundation for API Gateway quality assessment by evaluating the accuracy and relevance of 59 practice-derived metrics through expert judgment. This validation serves multiple purposes: (1) establishing consensus on metric definitions, (2) identifying a core set of reliable measures, (3) mapping validated metrics to quality concerns, and (4) providing a classification framework to guide adoption and future standardization efforts. To operationalize this goal systematically, four complementary research questions that address different facets of the validation process were formulated.

**RQ1:** *To what extent do domain experts agree on the accuracy and relevance of existing API Gateway quality metrics?*

Understanding the level of agreement among experts helps determine the perceived validity and practical applicability of the proposed metrics. Low consensus may indicate ambiguity, redundancy, or contextual variability that needs to be addressed before broader adoption.

**RQ2:** *Which metrics demonstrate strong consensus and high scores among experts, and can thus be considered core indicators for API Gateway quality evaluation?*

Identifying a prioritized set of validated metrics allows the construction of a focused, reliable foundation for API Gateway quality assessment that aligns with expert expectations and practical concerns.

**RQ3:** *What categories of API management capabilities are mostly represented by the validated metrics?*

Mapping validated metrics to broader quality concerns clarifies coverage, highlights gaps, and informs the development of a structured quality model tailored to the specific needs of API Gateway evaluation.

**RQ4:** *How can API Gateway quality metrics be classified based on expert ratings to guide their adoption, review, or exclusion?*

A systematic classification framework is needed to translate expert judgments into actionable categories that support decision-making. By defining clear criteria for grouping metrics, this classification can prioritize metrics for inclusion in quality models, identify those requiring further review, and exclude unreliable or irrelevant measures.

Collectively, these four research questions address complementary aspects of metric validation: RQ1 establishes the reliability baseline through consensus measurement; RQ2 identifies the validated core suitable for adoption; RQ3 ensures domain coverage by mapping to established capability frameworks; and RQ4 provides a classification mechanism that translates empirical findings into practical guidance. Answering all four RQs enables us to achieve the overarching goal of establishing a robust, expert-validated foundation for API Gateway quality assessment.

#### 4.1. A practice-oriented approach

The deliberate focus on metrics from deployed systems introduced in this work was informed by findings from our prior SMS [6]. That study found that: most contributions were categorized as technical implementations or specific solutions, often without grounding in established software quality models; research generally did not explicitly utilize formal quality models; and a notable absence of standardized, validated frameworks for API Gateway quality assessment existed. Given this landscape, a practice-first approach was methodologically motivated by:

1. Complementary evidence: The SMS [6] captured what researchers propose; [3] revealed the practitioners perspective of quality requirements; and the platform analysis [9] captured what practitioners implement. Together, these provide comprehensive coverage of the domain.
2. Bridging the research-practice gap: validating operational metrics already deployed ensures to create a pathway for integrating industry practice with academic quality frameworks, addressing the gaps identified in [6].
3. Ecological validity: Metrics implemented in production platforms have demonstrated feasibility, measurability, and perceived utility by platform developers and users across diverse organizational contexts.
4. Operationalizability: Real-world metrics come with implicit or explicit measurement procedures, reducing ambiguity about measurement implementation. It helps to mitigate definitional inconsistencies.

#### 4.2. Origin and development of the preliminary metric set

The 59 API Gateway software metrics evaluated in this study were systematically derived through a comprehensive analysis of mainstream API Gateway platforms, as detailed in our prior work [9]. This subsection describes the methodological approach used to ensure the breadth and representativeness of the preliminary metric catalog.

The metric extraction process analyzed 68 API Gateway software offerings, selected to represent the current market landscape comprehensively. The platform sample included commercial enterprise solutions, open-source platforms, cloud-native gateways, and hybrid and specialized solutions across various deployment models.

Platform selection was guided by three industry reports that catalog active API management solutions: Gartner Magic Quadrant for API Management,<sup>1</sup> Postman State of API Report,<sup>2</sup> and Apidays global API Landscape<sup>3</sup>. This triangulated approach ensured coverage of both market leaders and emerging platforms, representing diverse architectural patterns, target audiences, and operational contexts.

##### 4.2.1. Metrics extraction and definition

Following the DSR [44] framework, the metric extraction process consisted of four systematic phases.

1. Documentation review: For each platform, official documentation was systematically reviewed, including product documentation and technical specifications, monitoring and observability guides, dashboard and reporting interface descriptions, API reference documentation, and administrative console specifications.

2. Metric identification: Measures explicitly exposed, tracked, or reported by each platform were cataloged. This included metrics available through built-in monitoring dashboards, exported telemetry data, and logging and analytics features.
3. Standardization: Metrics with similar semantic meaning but different naming conventions across platforms were consolidated.
4. Categorization: Metrics were organized according to established API management capability frameworks [5, 22]. This categorization ensured that the metric set was grounded in recognized API management concerns and aligned with both academic research and industry practice.

For each identified metric, a formal definition was developed including: a metric code and a human-readable, standardized identifier; a clear, operational description of what the metric measures; and an associated category, derived from a set of seven capability-based categories (API performance, API utilization, caching, errors, latency and response time, resource utilization, system health, and traffic monitoring).

### 4.3. Sampling technique and data manipulation

This work employed a survey methodology [11, 12, 45] aimed at validating a preliminary catalog of 59 API Gateway software quality metrics [9] through expert judgment [10]. The survey followed an exploratory approach based on the classification proposed by [46], which focuses on using surveys to gather insights and assess the quality characteristics of software management tools and practices. The objective of the survey was to evaluate the accuracy and relevance of each proposed metric for assessing API Gateway quality, as perceived by a select group of experts in the field.

This evaluation study [10] applied a purposive judgmental sampling scheme, specifically expert sampling [47, 48]. Expert sampling is a type of purposive sampling where participants are intentionally selected for their recognized expertise in the field. This technique ensures that the feedback is credible, informed by deep subject knowledge, and directly relevant to the study's objectives.

**Sample size:** Seven experts participated in this validation study ( $N = 7$ ). This sample size aligns with established practices for expert judgment studies in software engineering [46, 47], where small, purposively selected panels of 5–10 specialists are typical, providing a manageable yet diverse set of opinions.

The selection aimed to balance theoretical perspectives with industry experience, ensuring that the feedback would be rigorous and applicable to both academic research and real-world practices. Experts were selected based on the following criteria: (1) recognized contributions to API management research or practice (e.g., peer-reviewed publications, book authorship, or platform development leadership), (2) minimum of three years of professional experience with API technologies, and (3) current active engagement in API-related work.

The survey was implemented using an online questionnaire platform, facilitating efficient data capture and ensuring ease of access for the participants. The platform allowed participants to submit responses and additional comments, providing both quantitative and qualitative data. Then, survey responses were exported as CSV files.

Data preprocessing, statistical computation, and visualization, including the generation of bar plots and distribution charts, were performed using RStudio (version 2025.05.1+513) [49] and R (version 4.5.0) [50], leveraging the `ggplot2` package (version 3.5.2) [51] for

plotting. Figures were incorporated to enhance interpretability of expert rating distributions and consensus levels.

#### 4.4. Survey design

The survey instrument was designed to validate a catalog of 59 API Gateway quality metrics proposed in earlier work [9]. It consisted of four primary sections: introduction, demographic profiling, metric evaluation, and final feedback.

**Introduction:** As suggested by [12], this section provided context for the study, including an overview of the research group, a summary of the previous work on the preliminary catalog, and the objectives of the current validation effort. It also informed participants the estimated completion time (approximately 15–20 minutes), and the intended use of the responses in refining the catalog.

**Demographic information:** This section gathered background data on participants, including name and email (for contact if needed), current professional sphere, primary role, years of experience, exposure to API technologies, and involvement in API-related communities. This information supported the verification of participant expertise and relevance to the API management domain.

**Metric assessment:** The core of the survey consisted of an evaluation of 59 metrics, summarized in Appendix A. For each metric, respondents were shown the metric’s name, definition, and category based on API management capabilities, as outlined in [5, 22]. Participants rated each metric using a five-point Likert scale (1 = Strongly Disagree (SD) to 5 = Strongly Agree (SA)) across two dimensions: (i) *Accuracy* – the extent to which the metric definition was understandable, and (ii) *Relevance* – the perceived importance of the metric for assessing API Gateway quality. Radio buttons were used to encourage intentional responses and minimize accidental selections.

The question format is illustrated in Table 1 and an example question is provided in Table 2.

Table 1. Survey question structure

Metric Descriptor: Text. Contains the metric code and metric name.					
Definition: Text. Brief definition of the metric.					
Category: Text. Based on API Management capability.					
			Ratings		
Criteria	SD	D	NAD	A	SA
Accuracy: this metric is well-defined.					
Relevance: this metric is relevant.					

The complete survey instrument, including all instructions, demographic questions, and the 59 metric evaluation items, was implemented using Google Forms. A PDF export of the complete survey showing all questions, definitions, rating scales, and instructions exactly as presented to participants is publicly available (see Data Availability section).

**Final feedback:** The survey concluded with an open-ended question soliciting general comments, suggestions, or observations from participants. This provided an opportunity for qualitative insights regarding the catalog content, clarity of definitions, coverage breadth, and any potential omissions.

Table 2. Example metric evaluation question

Metric Descriptor: <b>AU.1.1 Active Inbound Data</b>					
Definition: The total volume of inbound data currently being processed by the API Gateway. Reflects the real-time data received from clients.					
Category: API Utilization Metrics (AU).					
	SD	D	NAD	A	SA
Accuracy: this metric is well-defined.					
Relevance: this metric is relevant.					

In total, each participant was asked to complete 126 responses. The survey was made available from May 31 to June 21, 2025, providing a three-week period for participation. This timeframe was selected to offer respondents sufficient flexibility and time to complete the questionnaire at their convenience.

#### 4.5. Statistical analysis of expert ratings

To assess the inter-reliability and internal consistency of expert evaluations, a multi-method statistical validation strategy was adopted, following the principles outlined in [12, 52], as detailed as follows.

**Descriptive analysis and visualization:** A preliminary analysis was performed to examine the distribution of responses, identify potential outliers, and detect patterns of agreement or divergence between accuracy and relevance assessments. Visualization tools supported the initial interpretation of score dispersion and item-level trends.

**Exact agreement count:** For each metric, the number of identical responses across raters was computed, providing a direct measure of consensus. This allowed metrics with high or low rating uniformity to be identified.

**Analysis of variance:** A one-way ANOVA [53] was applied to determine whether significant differences existed in ratings across metrics. This tested the null hypothesis that all items were evaluated similarly. Where significant results were observed, Tukey's Honestly Significant Difference (HSD) [54] post-hoc test was used to identify which pairs of metrics differed significantly.

**Kendall's coefficient of concordance (W) [55]:** This non-parametric statistic was used to measure the overall level of agreement among raters. Given the ordinal nature of the Likert-scale data, Kendall's W provided an appropriate indicator of consensus across all experts for both accuracy and relevance dimensions.

**Internal consistency:** To assess whether the metric evaluations consistently reflected underlying constructs, Cronbach's alpha [56] was calculated. Values closer to 1.0 indicate high internal consistency.

**Inter-rater reliability:** Fleiss' Kappa [57] was computed to evaluate the extent of agreement beyond chance across multiple raters for categorical Likert-scale responses. This complemented Kendall's W by offering a more conservative measure that accounts for probability of random agreement.

The selected statistical techniques were chosen to comprehensively assess the reliability, consistency, and interpretability of the expert evaluations, particularly given the ordinal nature of the data and the small sample size typical of expert judgment studies.

#### 4.6. Tier-based classification of metrics

To support selection of a refined and robust set of evaluation metrics, a tier-based classification scheme is proposed. It integrates both the central tendency and variability of expert ratings. The algorithm (see Algorithm 1) systematically categorizes each metric based on its mean and standard deviation values for both accuracy and relevance.

This approach balances two critical dimensions:

1. *Rating quality (mean)*: Higher average scores reflect stronger expert endorsement of the metric's effectiveness and relevance.
2. *Rating consistency (standard deviation)*: Lower variability indicates greater agreement among raters and stronger interpretative cohesion.

The algorithm was implemented in R and applied to each metric in the dataset, classifying metrics into one of three mutually exclusive tiers:

**Tier 1 – Core Metrics**: Metrics that meet stringent quality and consistency thresholds. These include metrics with exceptionally high relevance (mean  $\geq 4.25$ ), or those with above-average accuracy ( $\geq 3.66$ ) and relevance ( $\geq 4.0$ ), combined with low variability (SD  $\leq 1.0$  for accuracy and  $\leq 0.89$  for relevance). These metrics represent the most reliable and well-supported items.

**Tier 2 – Moderate-agreement Metrics**: Metrics that demonstrate generally positive evaluations but fall short of Tier 1 criteria. These items show acceptable accuracy and relevance with moderate variability and may be candidates for further refinement or contextual validation.

**Tier 3 – Low-agreement Metrics**: Metrics characterized by low perceived quality (mean  $< 3.0$  for either dimension) or high disagreement among experts (SD  $> 1.2$  for accuracy or relevance). These items are flagged as potentially unreliable or inconsistent and are less suitable for continued use without revision.

---

Algorithm 1. Tier-based classification scheme of metrics

---

```

1: for each row in metrics_summary do
2:    $a_\mu \leftarrow$  accuracy_mean
3:    $a_\sigma \leftarrow$  accuracy_sd
4:    $r_\mu \leftarrow$  relevance_mean
5:    $r_\sigma \leftarrow$  relevance_sd
6:   if  $r_\mu \geq 4.25$  then
7:     assign Tier  $\leftarrow$  "Tier 1: Core metrics"
8:   else if  $a_\mu \geq 3.66$  and  $r_\mu \geq 4.0$  and  $a_\sigma \leq 1.0$  and  $r_\sigma \leq 0.89$  then
9:     assign Tier  $\leftarrow$  "Tier 1: Core metrics"
10:  else if  $a_\mu < 3.0$  or  $r_\mu < 3.0$  or  $a_\sigma > 1.2$  or  $r_\sigma > 1.2$  then
11:    assign Tier  $\leftarrow$  "Tier 3: Low-agreement metrics"
12:  else
13:    assign Tier  $\leftarrow$  "Tier 2: Moderate-agreement metrics"
14:  end if
15: end for

```

---

This tiered structure facilitates evidence-based metric selection by distinguishing high-confidence metrics from those requiring further scrutiny.

## 5. Findings

### 5.1. Population characterization

In total, seven participants took part in the study. The initial part of the survey aimed to profile the respondents, offering context for interpreting their perspectives. This characterization included institutional affiliation, professional roles, experience in the field, familiarity with relevant technologies, and engagement with the API-related community.

All participants met predefined expertise criteria, including recognized contributions to the API management domain, substantial professional experience with API technologies, and active participation in API development, research, or platform implementation. Professional affiliation, years of experience, and community involvement are shown in Figure 1 and summarized below.

**Professional affiliation:** Participants came from a mix of sectors including academia, private industry, and the public sector. The sample reflects a balanced representation between academic and corporate environments (three participants each), with more limited input from government organizations (one participant).

**Years of experience:** Participants demonstrated considerable API expertise across varying career stages. As illustrated in Figure 1a, five participants (71%) reported 10 or more years of API-related experience, one participant (14%) reported 7-10 years, and one participant (14%) reported 3 years of experience. This distribution indicates that expert evaluations were predominantly provided by senior professionals with extensive accumulated knowledge in the API domain, while still maintaining the minimum expertise threshold of 3 years for all participants.

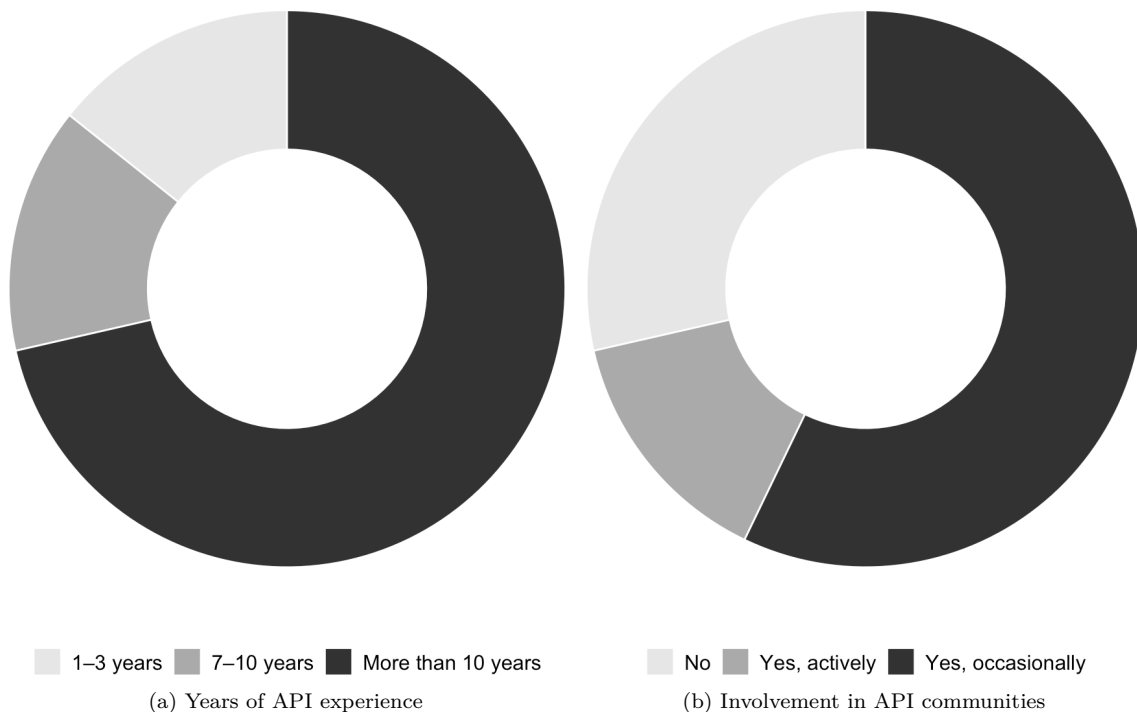


Figure 1. Professional background and community engagement of the respondents

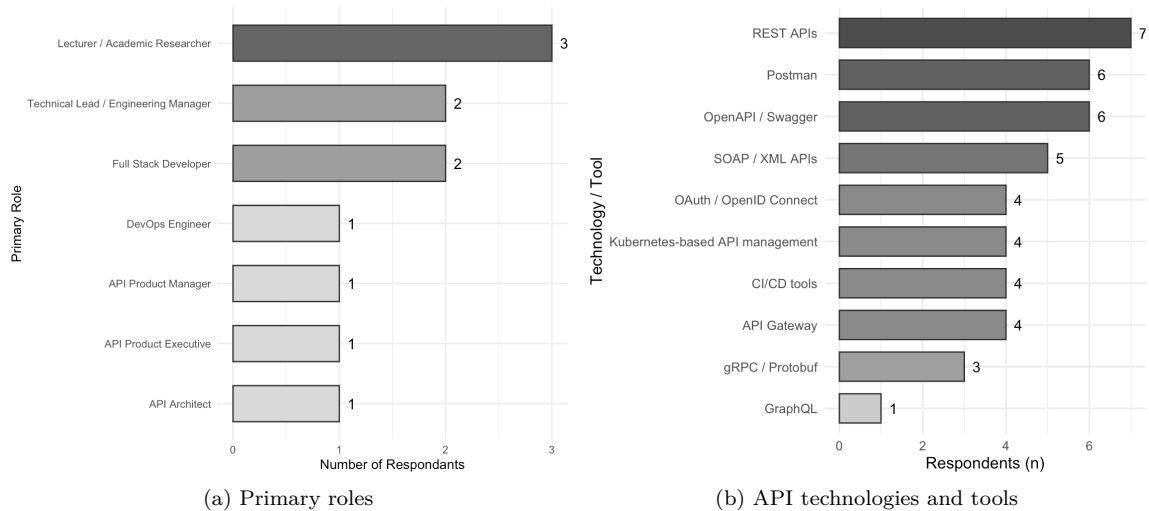


Figure 2. Professional roles and API-related tools expertise of specialists

**Community engagement:** Figure 1b shows that most respondents have at least some involvement in API-related communities, events, or open-source projects, though active engagement is relatively rare. This suggests a group that is technically experienced, but more selectively engaged in community-oriented activities.

**Professional roles:** Respondents were asked to identify their primary roles in API-related work; multiple selections were allowed. Figure 2a shows a concentration in academic and technical leadership positions, as well as developer roles. This indicates that the surveyed group brings a combination of hands-on technical experience and strategic oversight.

**API technologies and tools:** Participants reported extensive hands-on experience with a wide range of API technologies. As depicted in Figure 2b, REST APIs, OpenAPI/Swagger, and Postman stood out as the most consistently used tools. The data also reflect notable use of legacy protocols like SOAP/XML and growing adoption of cloud-native and CI/CD-oriented practices.

Professional roles and API-related expertise can be seen next in Figure 2.

## 5.2. Expert ratings

This section presents a comprehensive overview of expert ratings across the metrics based on two assessed criteria: accuracy and relevance.

Figure 3 displays the concordance per item. Each bar shows the maximum number of experts who assigned the same rating to a metric's accuracy (3a) and relevance (b).

Overall, expert agreement was moderate. The most common level of concordance ranged between three and four experts assigning the same rating for both accuracy and relevance.

Figure 4 illustrates the distribution of ratings given by each expert. The rating distribution per expert showed no strong bias from any individual rater for both accuracy (4a) and relevance (4b) criteria. Some experts used the full Likert-scale range, while others gravitated toward middle and upper ratings, indicating slight variation in interpretive thresholds.

Finally, Figure 5 depicts how ratings were distributed across individual metrics. Metric-level rating distributions (subfigures 5a and 5b) revealed that experts generally evaluated the metrics favorably, particularly in terms of relevance (5b). The most common

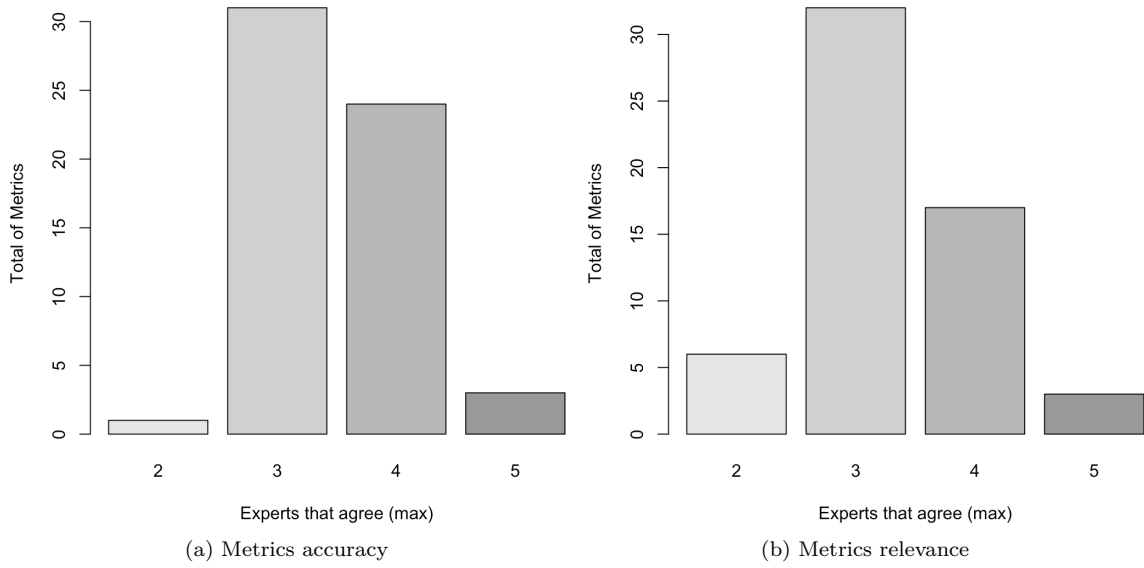


Figure 3. Expert concordance per metric

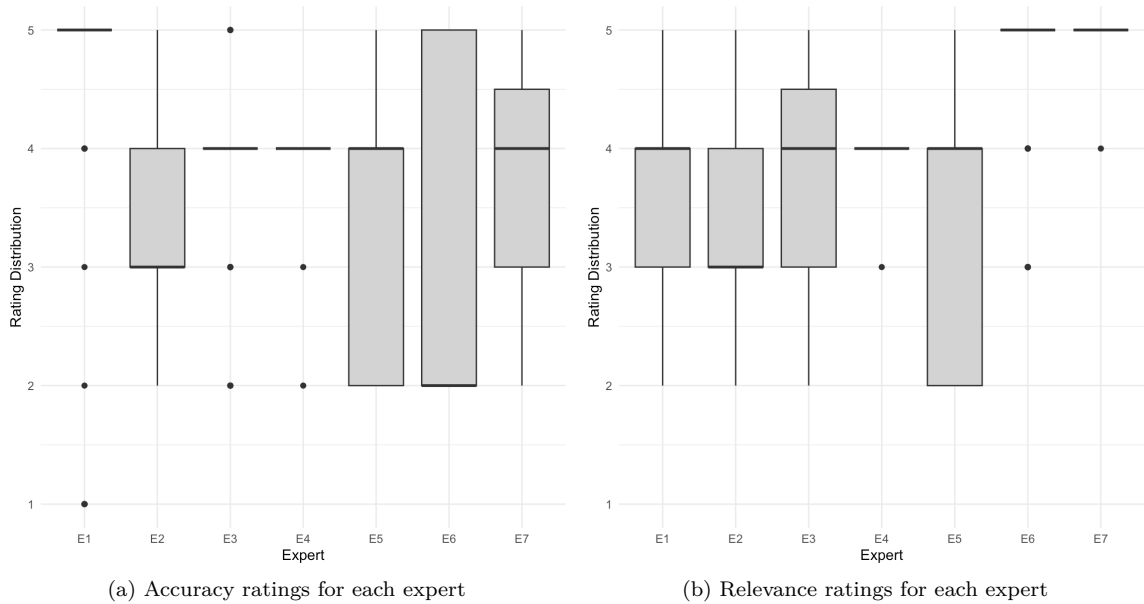
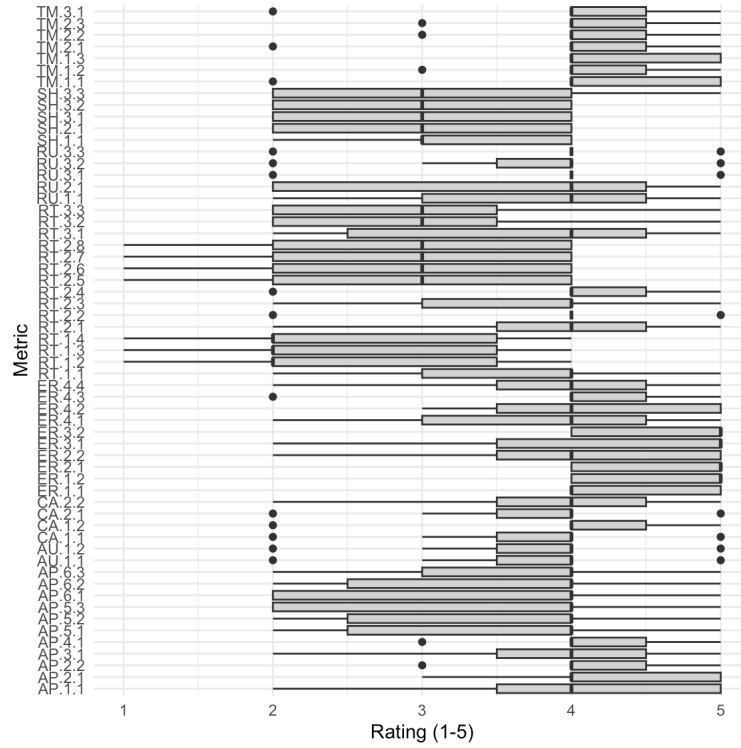


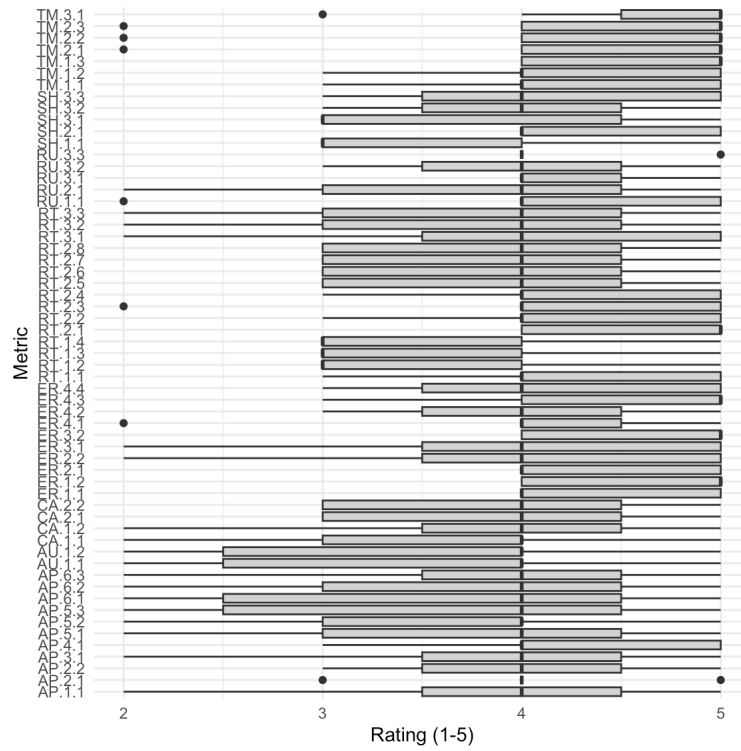
Figure 4. Overview of experts ratings' distribution

response was “Agree”, followed by “Strongly Agree”. No “Strongly Disagree” responses were given for relevance, but a few appeared under accuracy (5a), suggesting that while most metrics were seen as useful, some definitions may need refinement.

These observed patterns in expert behavior reinforced the need for further analysis of inter-rater consensus using statistical methods suggested in the literature. The analysis conducted in this work is described in the following section.



(a) Accuracy ratings distribution for each metric



(b) Relevance ratings distribution for each metric

Figure 5. Overview of metrics scores distribution

### 5.3. Statistical validation

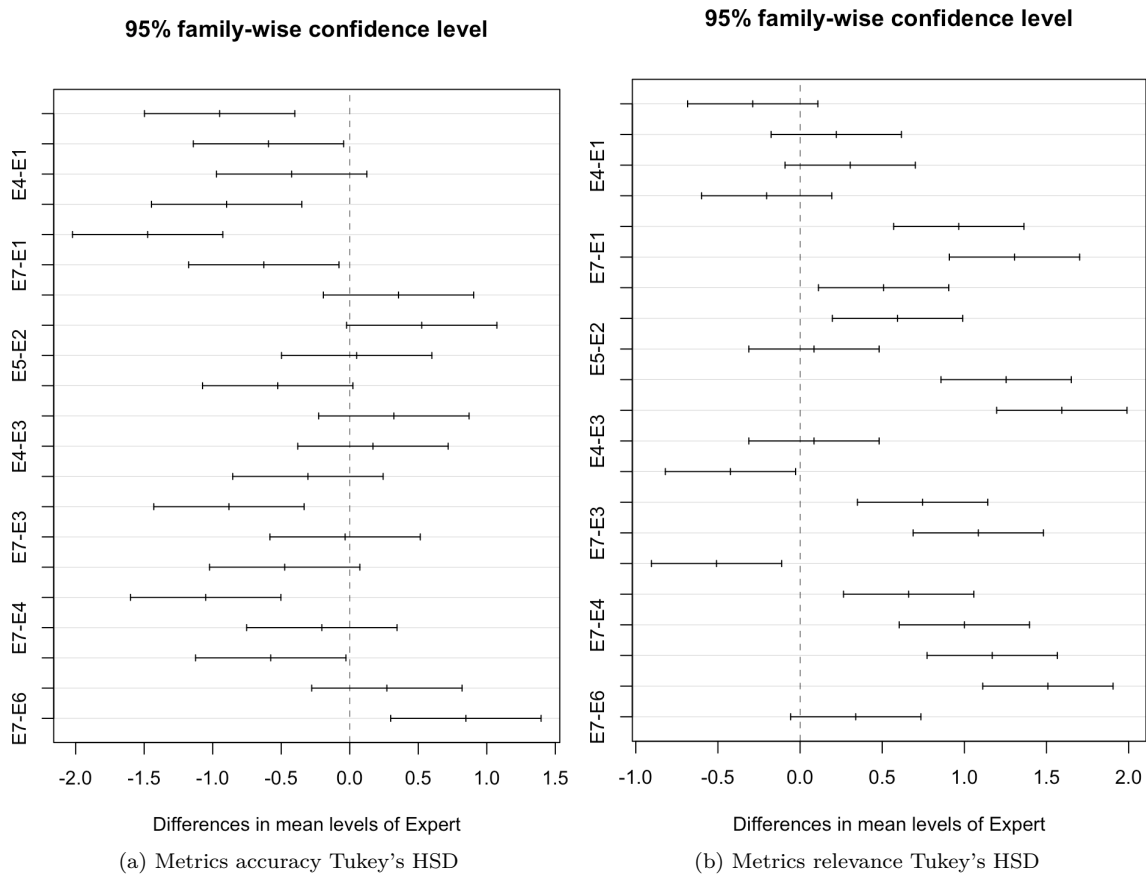
A multi-method statistical analysis was conducted to assess inter-rater agreement. Table 3 summarizes the main indicators. Descriptive statistics of metrics accuracy and relevance ratings can be seen in Appendices B and C, respectively.

Table 3. Summary of inter-rater agreement statistics over metrics

Statistic	Accuracy ratings	Relevance ratings
ANOVA [53]	$F(6, 406) = 12.49, p < 0.001$	$F(6, 406) = 39.9, p < 0.001$
Cronbach's Alpha [56]	$\alpha = 0.57$ (CI: 0.37–0.72)	$\alpha = 0.33$ (CI: 0.03–0.56)
Kendall's W [55]	$W = 0.0814, p = 0.997$	$W = 0.0661, p = 1.000$
Fleiss' Kappa [57]	$\kappa = -0.0393, p = 0.0221$	$\kappa = -0.0389, p = 0.0338$

The results revealed low to moderate levels of inter-rater agreement. Cronbach's alpha showed moderate internal consistency for accuracy but low reliability for relevance. Kendall's W and Fleiss' Kappa suggested low or even sub-chance agreement in both dimensions.

ANOVA tests confirmed statistically significant differences in expert scoring. This led to performing Tukey's HSD post hoc tests to identify specific differences in expert scoring behavior across metrics. Figure 6 summarizes the Appendix D as two subfigures, each corresponding to one evaluation criterion, namely accuracy and relevance.



Regarding accuracy (Figure 6a), significant differences were found between several expert pairs. Specifically, experts E2, E3, E5, E6 and E7 each differed significantly from E1 ( $p < 0.05$ ), with E6–E1 showing the largest mean difference ( $\Delta = -1.47$ ,  $p < 0.0001$ ). Additionally, E6 differed significantly from E3, E4, and E5, and E7 differed from E6. These results suggest that E1 generally gave higher ratings than most others, with E6 showing the lowest relative values.

In contrast, the second Tukey test (6b) revealed that experts E6 and E7 had significantly higher scores than most others. E6 differed significantly from E1, E2, E3, and E4 ( $p < 0.001$ ), and E7 differed from all groups except E6, with a maximum mean difference of 1.59 over E2. Notably, E5 showed significantly lower values than E3 and E4, but did not differ significantly from E1 or E2.

#### 5.4. The core metrics set

Based on the expert assessment, Table 4 presents the definitions of 24 core metrics. Importantly, names and definitions were refined in response to feedback, as specialists remarked on the need for adopting consistent naming conventions and comprehensive descriptions.

Table 4. API Gateway core metrics set

Code	Human-readable name	Description
AP.2.1	Successful Response Rate	Percentage of API responses with HTTP 2xx status codes relative to the total number of API requests during a specified time period.
AP.2.2	Successful Request Count	Total number of API requests that resulted in successful responses (HTTP 2xx status codes) during a specified time period.
AP.4.1	Request Throughput	Number of API requests processed per second (or defined time unit) by the API Gateway.
ER.1.1	Total Error Count	Total number of failed API requests (HTTP 4xx and 5xx status codes) recorded during a specified time period.
ER.1.2	Overall Error Rate	Percentage of failed API requests (HTTP 4xx and 5xx status codes) relative to the total number of API requests during a specified time period.
ER.2.1	Client Error Count	Total number of client-side errors (HTTP 4xx status codes) recorded during a specified time period.
ER.3.2	Server Error Rate	Percentage of server-side errors (HTTP 5xx status codes) relative to the total number of API requests during a specified time period.
ER.4.2	Exception Count	Total number of API requests that encountered unexpected conditions or exceptions during processing, regardless of HTTP status code.
ER.4.3	Authentication Error Count	Total number of authentication-related failures during a specified time period, including invalid credentials, expired tokens, missing authentication headers, and authorization failures.
RT.1.1	Internal Processing Time	Time spent by the API Gateway processing requests internally, including routing, plugin execution, and request transformation.

Table 4 (cont.)

Code	Human-readable name	Description
RT.2.1	Total Response Time	Complete end-to-end response time from when the API Gateway receives a request until it returns a response to the client, including both gateway processing time and upstream service response time.
RT.2.2	Maximum Response Time	Highest observed response time for any API request during a specified time period.
RT.2.4	Response Time Standard Deviation	Statistical measure of response time variability, indicating the consistency of API performance.
RU.3.1	Average Memory Usage (Last Hour)	Average memory consumption by the API Gateway instance over the past hour, measured in bytes or percentage of available memory.
RU.3.2	Maximum Memory Usage	Peak memory utilization observed by the API Gateway instance during its operational period.
RU.3.3	Node Memory Usage	Memory consumption per individual API Gateway node in a distributed or clustered deployment.
SH.2.1	Upstream Health Status	Operational status indicator for upstream services and backend systems, typically represented as healthy, degraded, or unavailable.
TM.1.1	Total API Count	Total number of individual APIs deployed and actively managed within the API Gateway platform.
TM.1.2	Total Product Count	Total number of API products or bundles configured in the API Gateway, where each product may contain multiple APIs with associated usage plans, rate limits, and access policies.
TM.1.3	Total Request Count	Total number of API requests received and processed by the API Gateway during a specified time period, regardless of success or failure status.
TM.2.1	Total Data Volume	Combined volume of data transferred through the API Gateway during a specified time period, including both inbound requests and outbound responses.
TM.2.2	Inbound Data Volume	Total volume of data received by the API Gateway from clients during a specified time period, including request headers, body content, and metadata.
TM.2.3	Outbound Data Volume	Total volume of data transmitted from the API Gateway to clients during a specified time period, including response headers, body content, and metadata.
TM.3.1	Throttling Error Count	Number of API requests rejected due to exceeding configured rate limits, quota restrictions, or throttling policies.

## 6. Discussion

The need for a formal, expert-validated, metrics-based quality model for API Management has been emphasized by recent work [6, 9], which identified fragmentation in existing research and a lack of standardized quality evaluation frameworks. Our study directly addressed this gap by providing a set of metrics, empirically validated by expert judgment. This section interprets the findings from each research question, synthesizes cross-cutting themes, and discusses their practical, theoretical, and methodological implications.

## 6.1. Interpretation of research questions

### 6.1.1. RQ1: Expert agreement levels

RQ1 asked: **To what extent do domain experts agree on the accuracy and relevance of existing API Gateway quality metrics?**

The statistical analysis (Subsection 5.3, Table 3) revealed low to moderate inter-rater agreement. However, this apparent lack of consensus requires careful interpretation in light of the study's deliberate sampling strategy. The expert panel was intentionally composed of diverse profiles spanning academia and industry, ranging from emerging to highly senior professionals (3 to more than 10 years), and representing varied organizational contexts. This heterogeneity was a methodological design choice aimed at ensuring the validated metrics would be applicable across diverse API Gateway deployment scenarios rather than reflecting a narrow, context-specific perspective.

Given this intentional diversity, the observed inter-rater variability reflects two distinct phenomena. First, legitimate perspective diversity: Experts from different backgrounds may appropriately weight accuracy and relevance differently based on their operational contexts. For example, an enterprise architect managing cloud-native deployments may prioritize different metric characteristics than a researcher focusing on quality model formalization. The concordance analysis (Figure 3) supports this interpretation: most metrics had 3–4 experts assigning the same rating, indicating **moderate practical consensus within subgroups** despite low overall statistical coefficients.

Second, scale interpretation differences: Distribution analysis (Figures 4 and 5) showed that experts generally evaluated metrics favorably, particularly for relevance, where “Agree” and “Strongly agree” dominated responses. The ANOVA results ( $F(6, 406) = 12.49$  and  $F(6, 406) = 39.9$ ,  $p < 0.001$ ) and Tukey's HSD post hoc tests (Figure 6) revealed that some experts (e.g., E6 and E7) consistently used higher ratings while maintaining similar relative rankings of metrics. This suggests differences in how individuals use Likert scales rather than fundamental disagreement about metric value.

Importantly, this variability should not be interpreted as validation failure. Rather, it demonstrates that the expert panel successfully captured the heterogeneity of real-world API Gateway stakeholders. The tier-based classification (RQ4) was specifically designed to handle this diversity by identifying metrics with robust agreement despite varied perspectives (Tier 1), while flagging those requiring context-specific refinement (Tiers 2 and 3).

### 6.1.2. RQ2: Core metrics identification

RQ2 asked: **Which metrics demonstrate strong consensus and high scores among experts, and can thus be considered core indicators?**

Despite moderate overall agreement, the tier-based classification algorithm (Algorithm 1) successfully identified **24 core metrics** (Tier 1) that met stringent quality and consistency thresholds. These metrics demonstrated either exceptionally high relevance (mean  $\geq 4.25$ ) or combined above-average accuracy ( $\geq 3.66$ ) and relevance ( $\geq 4.0$ ) with low variability (SD  $\leq 1.0$  for accuracy and  $\leq 0.89$  for relevance).

The core set spans all major capability categories: **Errors** (9 metrics), **Traffic monitoring** (7 metrics), **Latency and response time** (4 metrics), **Resource utilization** (3 metrics), and **System health** (1 metric). Notably, error-related metrics (ER.x codes)

received the highest combined accuracy and relevance scores, with metrics such as ER.1.2 (Overall Error Rate) and ER.2.1 (Client Error Count) achieving means above 4.4 with standard deviations below 0.54.

This finding directly addresses a practical need: **operational metrics that experts unanimously recognize as essential** can serve as a validated foundation for quality assessment frameworks. The 24 core metrics represent the intersection of theoretical importance and practical feasibility, as they are both expert-endorsed and already implemented in production systems [9].

### 6.1.3. RQ3: Capability coverage

**RQ3 asked: What categories of API management capabilities are mostly represented by the validated metrics?**

The core metrics set (Table 4) demonstrates **comprehensive coverage across key API Gateway concerns**, with particularly strong representation in areas aligned with practitioner priorities identified in [3]:

- **Error handling and reliability** (9 metrics): total error counts, error rates by type (client/server), authentication failures, and exceptions. This aligns with reliability being consistently prioritized by practitioners.
- **Traffic monitoring and functional suitability** (7 metrics): API counts, request volumes, data transfers, and throttling. These support functional suitability assessment by tracking what the gateway actually processes.
- **Performance efficiency** (4 metrics): response times, processing latencies, and performance variability. Directly addresses performance efficiency, another top practitioner priority.
- **Resource utilization** (3 metrics): memory usage patterns across deployment configurations. Essential for operational efficiency and scalability assessment.
- **System health** (1 metric): upstream service status monitoring.

Notably absent from Tier 1 are **Caching** and **API Utilization** metrics, which showed lower consensus. This gap may reflect variability in how organizations implement caching strategies or the context-dependent nature of utilization thresholds.

The coverage pattern validates the practice-oriented approach: validated metrics concentrate in areas where **both academic research [6] and practitioner concerns [3] converge** – namely, performance efficiency, reliability, and functional suitability.

### 6.1.4. RQ4: Classification framework

**RQ4 asked: How can API Gateway quality metrics be classified based on expert ratings to guide their adoption, review, or exclusion?**

The tier-based classification (Algorithm 1) successfully translated mixed expert assessments into **actionable categories**:

- **Tier 1 (24 metrics, 41%)**: ready for immediate adoption in quality models, dashboards, and SLA definitions. High scores with low variability indicate robust expert endorsement.
- **Tier 2 (moderate-agreement metrics)**: candidates for definitional refinement. These metrics address relevant concerns but require improved clarity or contextual framing.
- **Tier 3 (low-agreement metrics)**: flagged for revision or exclusion. Many involved percentile-based indicators (e.g., RT.1.2–RT.1.4: IPT percentiles) or compound constructs that experts found ambiguous.

This classification mechanism provides **decision support guidance** for multiple stakeholder groups: researchers can prioritize Tier 1 metrics in quality models; platform developers can focus implementation efforts on high-consensus measures; and practitioners gain clarity about which metrics warrant investment in monitoring infrastructure.

Importantly, the framework is **iterative**: Tier 2 and 3 metrics can be refined and re-evaluated, creating a pathway for continuous improvement of the metric catalog.

## 6.2. Synthesis and cross-cutting themes

Having interpreted each research question individually, several cross-cutting themes emerge that illuminate the broader validation challenge.

**Rating asymmetry:** in all RQs, relevance scores consistently exceeded accuracy scores (Figures 5a vs. 5b). This asymmetry reveals that experts agree on *what* should be measured, but disagree on *how* clearly it is defined. Multiple experts noted that “all metrics may be relevant,” yet accuracy ratings varied significantly (Section 5.3). This suggests the catalog captured the right concepts but requires semantic refinement – a finding that directly motivates the implications discussed below.

**Interpretive variability as diagnostic signal:** The low agreement coefficients (RQ1) should not be interpreted as validation failure. Rather, they reflect the field’s evolving maturity in API Gateway quality assessment. Given the novelty and diversity of deployment patterns, strong consensus on metric priorities does not yet exist. Expert divergence thus serves as a diagnostic indicator: it identifies metrics needing contextual framing (Tier 2) versus those with broad agreement (Tier 1).

**Convergence with prior research:** The capability coverage patterns (RQ3) and core metrics identified (RQ2) align remarkably with findings from our previous studies. The SMS [6] identified performance efficiency, reliability, and functional suitability as most-addressed attributes. The practitioners’ survey [3] confirmed these as top priorities. The platform analysis [9] showed most implementations focus on throughput, latency, and error tracking. The current validation study reinforces this convergence: Tier 1 metrics concentrate precisely in these areas, providing strong triangulated evidence for their centrality to API Gateway quality.

In an effort to improve the preliminary catalog assessed in this paper, the evaluation process enabled the extraction of a high-confidence subset of metrics suitable for immediate adoption, while identifying others in need of refinement. The tier-based classification served as a constructive lens through which to interpret mixed expert assessments. Rather than enforcing a strict inclusion/exclusion boundary, the proposed structure enabled a more nuanced prioritization, by identifying metrics that are already robust (Tier 1), potentially useful but needing refinement (Tier 2), and those not yet ready for adoption (Tier 3).

Interestingly, many Tier 3 metrics involved percentile-based indicators or compound constructs. These are often seen in modern observability tools but may require contextual framing – if the system is considered “as a whole,” as some experts noted – or improved phrasing to be uniformly understood by diverse evaluators. Lower scores may reflect this interpretive challenge rather than a lack of utility per se.

For Tier 2 metrics, refinement is warranted. This may include improving wording, providing contextual examples, or offering visualizations to support interpretation. Iterative validation, possibly with practitioner feedback or scenario-based calibration, could improve consensus.

From a practical standpoint, the Tier 1 catalog can serve as a ready-to-adopt artifact for API Gateway operators and tool developers, among other roles. These metrics offer validated indicators that align with expert expectations and observed best practices, supporting immediate integration into dashboards, alert systems, Service Level Indicators (SLI), and Service Level Agreement (SLA) implementation.

Although 59 proposed metrics were initially comprised, the refined Tier 1 set is still broader and more comprehensive than what most mainstream API Gateway products currently provide. According to the quantitative analysis performed in our previous work [9], most commercial and open-source API Gateway platforms expose between 8 and 20 metrics, typically focused on throughput, latency, and error tracking. This highlighted the added value of the current validation effort, which consolidates expert consensus while advancing beyond the limited operational visibility offered by typical platforms.

### 6.3. Practical and theoretical implications

The findings from RQ1–RQ4 give rise to several practical and theoretical implications for API Gateway quality assessment:

**Semantic clarity as a design imperative:** The rating asymmetry identified in RQ1 (high relevance, variable accuracy) directly demonstrates that ambiguity in metric definitions erodes consensus even when concepts are recognized as important. For example, percentile-based metrics (RT.1.2–RT.1.4, classified as Tier 3) received lower accuracy scores despite measuring conceptually relevant constructs. This empirical finding underscores that metric definitions must be unambiguous, operational, and supported by measurement context to achieve widespread adoption in diverse enterprise settings.

**Diversity as validation strength:** The moderate inter-rater agreement observed in this study should be contextualized within the deliberate expert selection strategy. High agreement among homogeneous experts would indicate narrow applicability, while moderate agreement among diverse experts suggests the validated metrics address concerns across varied organizational contexts. This heterogeneity strengthens the external validity of the core metrics set: the 24 Tier 1 metrics achieved consensus despite panel diversity, indicating they represent genuinely universal API Gateway quality concerns rather than context-specific preferences. Future validation studies should consider panel composition as a tunable parameter: homogeneous panels for context-specific metrics, heterogeneous panels for general-purpose frameworks.

This study also underscores the limitations of expert-based validation studies. While expert judgment is indispensable in the early stages of metric validation, its effectiveness improves substantially when combined with rater training, controlled scenarios, or complementary quantitative validation.

Furthermore, the variability observed here reflects the field's evolving maturity. Given the novelty and flexibility of API Gateway deployments, strong consensus on metric priorities may not yet exist. In this light, expert divergence is not a failure but a diagnostic signal: a call for contextualized metrics that adapt to organizational needs rather than a one-size-fits-all solution.

Another important consideration emerging from both prior literature and expert feedback is the role of metric metadata in ensuring understanding, re-utilization, integration, and automation. In this sense, as argued in [58], including such metadata can mitigate inconsistent or erroneous interpretations in metrics analysis.

#### 6.4. Methodological implications

The methodological approach employed in this study – integrating multiple evidence sources through a practice-oriented validation strategy – yielded insights beyond the specific metrics validated:

The methodological approach introduced in this paper reflects a deliberate research strategy that integrates multiple evidence sources to address the API Gateway quality assessment gap comprehensively. The convergence across four complementary studies strengthens confidence in the resulting validated core metrics set:

- SMS [6] documented what research says about API management quality by identifying practices, capabilities, research methods, and outcomes, while revealing fragmentation and absent formal frameworks.
- Practitioners’ survey [3] captured what users prioritize, revealing that functional suitability, performance efficiency, reliability, and security are consistently highlighted as most critical – reinforcing the overlap between academic focus and practitioner needs.
- Platform analysis [9] revealed what is actually measured in production environments across 68 diverse implementations, with most platforms exposing 8–20 metrics focused on throughput, latency, and error tracking.
- Expert validation (current study) determined what domain experts consider both accurate and relevant, yielding 24 high-confidence core metrics organized in seven capability-based categories.

The remarkable convergence on performance efficiency, reliability, and functional suitability across all four perspectives provides strong validity evidence. This demonstrates that our core metrics address concerns that are simultaneously theoretically recognized in academic research, prioritized by practitioners in operational contexts, implemented in real-world platforms, and endorsed by domain experts.

This four-way validation significantly strengthens confidence in the resulting catalog as a foundation for standardization and enables bidirectional knowledge transfer:

- **From practice to research:** The validated catalog informs future quality model development with proven, measurable constructs that have demonstrated feasibility in production environments.
- **From research to practice:** Expert validation (including academic perspectives) elevates ad-hoc industry metrics to theoretically informed standards, providing the rigor necessary for formal standardization efforts.

The practice-oriented approach for the validation phase proved methodologically sound for several reasons that became evident through the validation process:

First, experts evaluated operational metrics more consistently when those metrics reflected actual implementation patterns. Although inter-rater agreement was moderate overall (Section 5.3), experts recognized these metrics as addressing genuine operational concerns, as evidenced by generally high relevance scores (Figure 5b). Several experts noted that the metrics reflected “what we actually monitor in production,” suggesting that grounding in real-world practice aided expert assessment.

Second, the variability in accuracy ratings (Section 5.3) revealed that even operationalized metrics benefit from definitional refinement. Had we included purely theoretical metrics without implementation precedent, agreement might have been even lower, as experts would lack shared reference points for interpretation. The practice-grounded approach provided a common foundation for evaluation.

Third, the finding that the proposed core metrics set is broader and more comprehensive than what most mainstream API Gateway products currently provide validates the synthesis approach: by aggregating metrics across 68 platforms, we identified a richer measurement space than any single platform exposes. This demonstrates the value of systematic analysis over reliance on any particular product's metric set.

## 7. Threats to validity

### 7.1. Construct validity

To mitigate ambiguity, all experts received a common set of definitions and instructions. The questionnaire was reviewed and refined by two authors prior to deployment. Accuracy and relevance were evaluated using standardized five-point Likert scales, and concise operational definitions were provided for both criteria. However, some metric descriptions may still have included terminology open to multiple interpretations, especially in the absence of domain-specific calibration or contextual examples.

No rater calibration session or shared interpretation workshop was conducted before or after the evaluation, potentially allowing diverging understandings of the criteria. Future replications should consider including a calibration phase to improve consistency.

Although expert feedback was based on clearly defined Likert scales (1 to 5), and criteria definitions were provided, the absence of fully specified metric metadata (e.g., scope, temporal constraints, or measurement conditions) may have affected how consistently experts interpreted some metrics – an issue previously highlighted [58] as critical for robust evaluation processes.

### 7.2. Internal validity

Experts were selected through formal invitations based on predefined criteria: authorship of widely cited research, leadership in API-related communities, and authorship of influential books. The deliberate inclusion of diverse profiles (academia, industry, public sector; varying experience levels; different organizational contexts) was a methodological design choice to mitigate selection bias and ensure broad applicability of validated metrics.

While this heterogeneity may contribute to lower statistical agreement coefficients, it strengthens the ecological validity of findings by capturing real-world stakeholder diversity. The trade-off between consensus (favoring homogeneous panels) and generalizability (favoring heterogeneous panels) was intentionally resolved in favor of the latter, consistent with the study's goal of developing general-purpose quality metrics rather than context-specific solutions.

In particular, the study deliberately excluded API Gateway developers to avoid evaluation bias tied to specific implementations. However, this exclusion may have omitted valuable operational insights.

Additionally, the evaluation was not anonymized – participants were asked to provide contact information for potential follow-up. Although identities were not considered during analysis, the lack of full anonymity could introduce minor social desirability bias. No such effects were observed, and rating distributions suggest a diversity of perspectives.

### 7.3. External validity

This study aimed to validate metrics applicable across varied API Gateway architectures, including cloud-native and enterprise-grade environments. The expert panel was intentionally diverse in academic and industry backgrounds, as well as in geographic representation, to increase the generalizability of results. Nonetheless, the inherent variability of API Gateway deployments and organizational contexts presents a challenge to universal applicability.

To enhance external validity, the metric set evaluated in this study was derived from real-world implementations analyzed in prior work [9]. This grounding in practical usage data complements the expert assessments and increases the likelihood that the validated metrics are broadly relevant.

### 7.4. Conclusion validity

The statistical analyses indicated low agreement among raters. These results reflect the known difficulty of achieving consensus in expert-based evaluations. Nonetheless, the adoption of a tiered classification algorithm allowed us to convert variability into actionable categorization – balancing mean scores with standard deviation to identify Core, Promising, and Low-Agreement metrics. This approach mitigated the threat posed by low agreement and provided a structured path for metric refinement and prioritization.

### 7.5. Study limitations

Beyond the methodological validity concerns discussed above, this study has several inherent limitations that should be considered when interpreting and applying the findings:

**Sample size and generalizability:** While seven experts represent an adequate sample for judgment studies, the relatively small panel limits statistical power and may not capture the full diversity of perspectives across the global API management community. The findings reflect expert consensus within this specific group and may not generalize to all organizational contexts or emerging API Gateway paradigms.

**Temporal scope:** The metric catalog was derived from platforms analyzed in 2024–2025. As API Gateway technologies and operational practices evolve, new metrics may emerge and current metrics may become obsolete or require redefinition. The validated set should be viewed as a snapshot of contemporary best practices rather than a permanent, unchanging standard.

**Practical validation absence:** This study validates metrics through expert judgment but does not include empirical testing in operational environments. The metrics' actual predictive validity – their ability to correlate with real system failures, performance issues, or user satisfaction – remains to be demonstrated through field studies.

**Metric definition granularity:** While the study validates metric concepts and relevance, it does not specify complete measurement procedures, including data collection methods, aggregation rules, threshold values, or temporal windows. Research and organizations implementing these metrics will need to operationalize these details based on their specific contexts.

**Coverage gaps:** The catalog focuses on metrics actually deployed in existing platforms, potentially overlooking theoretically important but not yet implemented quality aspects. Novel quality concerns emerging from new architectural patterns (e.g., service mesh integration, serverless gateways) may not be adequately represented.

These limitations suggest directions for future research, including larger-scale validation studies, longitudinal tracking of metric evolution, and field deployment studies.

## 8. Concluding remarks and future work

By combining practical definitions, expert judgment, and a tiered classification mechanism, this study proposes a validated and categorized set of API Gateway quality metrics, grounded in expert evaluation and aligned with both academic and practitioner perspectives. The resulting classification represents a significant advancement toward the standardization of observability-based quality assessment in the API Management field.

The statistical techniques employed during this research were chosen to comprehensively assess the inter-reliability, internal consistency, and interpretability of the expert evaluations, particularly given the ordinal nature of the data and the small sample size typical of expert judgment studies. In this sense, variability in expert judgment should not be viewed solely as a limitation, but as a reflection of the heterogeneity during expert judgment studies.

The insights provided by the expert panel reflected diverse perspectives from both academia and industry, which added credibility and depth to the findings. Importantly, results reinforced the need for semantic clarity and contextual adaptability in metric design. Experts acknowledged that relevance often depends on implementation scenarios.

While statistical methods guided the consensus analysis, this work also explored practical strategies to translate expert feedback into concrete catalog improvements: thus, metrics were classified under three different tiers, and a set of 24 core metrics was outlined. This validated subset of metrics contributes to bridge the gap between metric adoption in real-world platforms and their theoretical grounding in software quality models. The validation effort not only confirms the feasibility of metric-based assessment but also expands the operational scope beyond what is typically available in current API Gateway offerings in the industry.

Several research directions can build upon the findings of this study.

First, a key next step is to consolidate the validated metrics into a formal quality model tailored to API Gateway systems. This model should: (i) Offer a multi-perspective structure encompassing quantitative, operational, and conceptual views of quality. (ii) Include a complete and standardized metadata specification to describe each metric, including its definition, interpretation guidance, aggregation rules, thresholds, scope, and implementation considerations. (iii) Link metrics to higher-level software product quality attributes, enabling traceability between observed values and desired system properties.

Second, some Tier 2 and Tier 3 metrics may still offer potential value if reformulated. Further rounds of validation – potentially using Delphi methods, group interpretation workshops, or scenario-based rating – could improve definitions and agreement levels.

Third, the core metrics (Tier 1) set establishes a ready-to-use artifact which can be deployed into monitoring tools, dashboards, or observability platforms. Prototypes or reference implementations can test their effectiveness in conveying quality insights, guiding operations, and supporting decision making.

Finally, future studies should apply the validated metrics in real operational environments (e.g., enterprise or cloud-native gateways) to evaluate: (i) Their correlation with actual system behavior and failures. (ii) Their usefulness in incident diagnosis, trend analysis, or capacity planning. (iii) Their acceptance and interpretability by practitioners and teams.

## CRediT authorship contribution statement

E. dos Santos: conceptualization, data curation, formal analysis, methodology, investigation, resources, software, validation, visualization, writing – original draft, writing – review and editing.

S. Casas: conceptualization, funding acquisition, investigation, methodology, project administration, resources, supervision, writing – review and editing.

All authors reviewed the results and approved the final version of the manuscript.

## Declaration of competing interest

The authors declare that no competing interests exist.

## Data and code availability

The dataset supporting this study is publicly available through the Argentinean National Council of Scientific and Technical Research (CONICET) digital repository and can be accessed from <http://hdl.handle.net/11336/264879>.

It is provided in two files which contain: (i) Anonymized data collected from expert evaluations of the proposed software metrics (all personally identifiable information has been removed to preserve respondent confidentiality). (ii) The list of metrics evaluated by the experts, including each metric's definition and its final classification according to the tier-based scheme described in this article.

Both files are provided in a machine-readable format and are intended to support reproducibility and further analysis.

Additionally, the complete survey instrument (PDF export from Google Forms) is publicly available through the Universidad Nacional de la Patagonia Austral institutional repository at <https://drive.google.com/file/d/1Yuws6AUKg2LMkX0AdQZF3M0BSBDC2bgb>.

## Funding

This research was supported by the Universidad Nacional de la Patagonia Austral (Argentina) under research project 29/A531, titled Administracion de APIs Web: Ecosistemas y Plataformas.

## References

- [1] S. Preibisch, *API development: A practical guide for business implementation success*. New York, NY: Springer Science+Business Media, 2018.
- [2] M. Waseem, P. Liang, M. Shahin, A. Di Salle, and G. Márquez, "Design, monitoring, and testing of microservices systems: The practitioners' perspective," *arXiv preprint arXiv:2108.03384*, 2021.
- [3] E. dos Santos and S. Casas, "API management and SQuaRE: A comprehensive overview from the practitioners' standpoint," in *Argentine Congress of Computer Science*. Springer, 2023, pp. 137–150.

- [4] D. Bermbach and E. Wittern, “Benchmarking web API quality – revisited,” *Journal of Web Engineering*, Vol. 19, No. 5-6, 2020, pp. 603–646.
- [5] B. De, *API Management: An Architect’s Guide to Developing and Managing APIs for Your Organization*. Apress Berkeley, CA, 2023.
- [6] E. dos Santos and S. Casas, “Unveiling quality in API management: A systematic mapping study,” in *L Latin American Computer Conference (CLEI)*. IEEE, 2024, pp. 1–10.
- [7] B. Iyer and M. Subramaniam, “The strategic value of APIs,” *Harvard Business Review*, Vol. 1, No. 7, 2015, p. 2015.
- [8] *ISO/IEC 25010:2011 – Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models*, International Organization for Standardization; International Electrotechnical Commission Std., 2011.
- [9] E. dos Santos and S. Casas, “A catalog of API gateway metrics and its quantitative evaluation,” *DYNA*, Vol. 92, No. 237, 2025, pp. 106–114.
- [10] K.J. Stol and B. Fitzgerald, “The ABC of software engineering research,” *ACM Transactions on Software Engineering and Methodology*, Vol. 27, No. 3, 2018. [Online]. <https://doi.org/10.1145/3241743>
- [11] S.L. Pfleeger and B.A. Kitchenham, “Principles of survey research: part 1: turning lemons into lemonade,” *ACM SIGSOFT Software Engineering Notes*, Vol. 26, No. 6, 2001, pp. 16–18.
- [12] B. Kitchenham and S.L. Pfleeger, “Principles of survey research part 4: Questionnaire evaluation,” *ACM SIGSOFT Software Engineering Notes*, Vol. 27, No. 3, 2002, pp. 20–23.
- [13] *ISO/IEC 25010:2023 – Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – Product Quality Model*, International Organization for Standardization; International Electrotechnical Commission Std., 2023.
- [14] Y. Duan, G. Fu, N. Zhou, X. Sun, N.C. Narendra et al., “Everything as a service (XaaS) on the cloud: Origins, current and future trends,” in *8th International Conference on Cloud Computing*. IEEE, pp. 621–628. [Online]. <http://ieeexplore.ieee.org/document/7214098/>
- [15] C. Fehling, F. Leymann, R. Retter, W. Schupeck, and P. Arbitter, *Cloud Computing Patterns: Fundamentals to Design, Build, and Manage Cloud Applications*. Springer Vienna. [Online]. <https://link.springer.com/10.1007/978-3-7091-1568-8>
- [16] R.T. Fielding, “Architectural styles and the design of network-based software architectures.”
- [17] S. Andreo and J. Bosch, “API management challenges in ecosystems,” in *Software Business*, S. Hyrynsalmi, M. Suoranta, A. Nguyen-Duc, P. Tyrväinen, and P. Abrahamsson, Eds. Springer International Publishing, Vol. 370, pp. 86–93, series Title: Lecture Notes in Business Information Processing. [Online]. [http://link.springer.com/10.1007/978-3-030-33742-1\\_8](http://link.springer.com/10.1007/978-3-030-33742-1_8)
- [18] A. Brown, J. Fishenden, and M. Thompson, *API Economy, Ecosystems and Engagement Models*. Palgrave Macmillan UK, pp. 225–236. [Online]. [http://link.springer.com/10.1057/9781137443649\\_13](http://link.springer.com/10.1057/9781137443649_13)
- [19] J. Bloch, “How to design a good API and why it matters,” in *Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications*. ACM, pp. 506–507. [Online]. <https://dl.acm.org/doi/10.1145/1176617.1176622>
- [20] A. Gamez-Diaz, P. Fernandez, and A. Ruiz-Cortés, “Governify for APIs: SLA-driven ecosystem for API governance,” in *Proceedings of the 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2019. Association for Computing Machinery, pp. 1120–1123, event-place: Tallinn, Estonia. [Online]. <https://doi.org/10.1145/3338906.3341176>
- [21] M. Medjaoui, E. Wilde, R. Mitra, and M. Amundsen, *Continuous API management*. O’Reilly Media, Inc., 2021.
- [22] M. Mathijssen, M. Overeem, and S. Jansen, “Identification of practices and capabilities in API management: A systematic literature review,” *arXiv preprint arXiv:2006.10481*, 2020.
- [23] *ISO 9001:2015 – Quality Management Systems – Requirements*, International Organization for Standardization Std., 2015.
- [24] *ISO/IEC/IEEE 24765:2017 – Systems and Software Engineering – Vocabulary*, International Organization for Standardization; International Electrotechnical Commission; Institute of Electrical and Electronics Engineers Std., 2017.

- [25] P. Nistala, K.V. Nori, and R. Reddy, "Software quality models: A systematic mapping study," in *International Conference on Software and System Processes (ICSSP)*. IEEE, 2019, pp. 125–134.
- [26] *ISO/IEC 9126:1991 – Software Engineering – Product Quality*, International Organization for Standardization; International Electrotechnical Commission Std., 1991, iSO/IEC 9126:1991.
- [27] *ISO/IEC 9126:2001 – Software Engineering – Product Quality*, International Organization for Standardization; International Electrotechnical Commission Std., 2001.
- [28] *ISO/IEC 14598:1999 – Software Engineering – Product Evaluation*, International Organization for Standardization; International Electrotechnical Commission Std., 1999.
- [29] *ISO/IEC 25000:2014 – Systems and Software Engineering – SQuaRE – Guide to SQuaRE*, International Organization for Standardization; International Electrotechnical Commission Std., 2014.
- [30] *ISO/IEC 25023:2016 – Systems and Software Engineering – Measurement of System and Software Product Quality*, International Organization for Standardization; International Electrotechnical Commission Std., 2016.
- [31] *ISO/IEC 25040:2024 – Systems and Software Engineering – Quality Evaluation Framework*, International Organization for Standardization; International Electrotechnical Commission Std., 2024.
- [32] S.S. Stevens, "On the theory of scales of measurement," *Science*, Vol. 103, No. 2684, 1946, pp. 677–680.
- [33] N. Fenton and J. Bieman, *Software metrics: A rigorous and practical approach*. CRC Press, 2014.
- [34] G. Poels and G. Dedene, "DISTANCE: A framework for software measure construction," KU Leuven Departement Toegepaste Economische Wetenschappen, DTEW Research Report 9937, 1999.
- [35] N. Dalkey and O. Helmer, "An experimental application of the Delphi method to the use of experts," *Management Science*, Vol. 9, No. 3, 1963, pp. 458–467.
- [36] J. Kontio, J. Bragge, and L. Lehtola, "The focus group method as an empirical tool in software engineering," in *Guide to advanced empirical software engineering*. Springer, 2008, pp. 93–116.
- [37] M. Shaw, "Writing good software engineering research papers," in *25th International Conference on Software Engineering, 2003. Proceedings*. Portland, OR, USA: IEEE, 2003, pp. 726–736.
- [38] M. Overeem, M. Mathijssen, and S. Jansen, "API-m-FAMM: A focus area maturity model for API management," *Information and Software Technology*, Vol. 147, 2022, p. 106890.
- [39] R. Yamamoto, K. Ohashi, M. Fukuyori, K. Kimura, A. Sekiguchi et al., "A quality model and its quantitative evaluation method for web APIs," in *25th Asia-Pacific Software Engineering Conference (APSEC)*, 2018, pp. 598–607.
- [40] A. Machini and S. Casas, "A preliminary GQM model to evaluate web API usability," *Memorias de las JAIIO*, Vol. 10, No. 2, 2024, pp. 1–13.
- [41] V.R. Basili, G. Caldiera, and H.D. Rombach, "The goal question metric approach," in *Encyclopedia of software engineering*, 1994.
- [42] M. Constanzo, S.I. Casas, G.B. Vidal, and D. Cruz, "Usos y problemas de las APIs web en la República Argentina," No. 44, pp. 79–97. [Online]. <https://rtyc.utn.edu.ar/index.php/rtyc/article/view/913>
- [43] A. Machini and S. Casas, "An empirical study on web API usability: The consumer-developer perspective," *Brazilian Journal of Technology*, Vol. 7, No. 4, 2024, p. e74474.
- [44] K. Peppers, T. Tuunanen, M.A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of Management Information Systems*, Vol. 24, No. 3, 2007, pp. 45–77.
- [45] B. Kitchenham and S.L. Pfleeger, "Principles of survey research part 6: Data analysis," *ACM SIGSOFT Software Engineering Notes*, Vol. 28, No. 2, 2003, pp. 24–27.
- [46] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell et al., *Experimentation in Software Engineering*. Springer Berlin Heidelberg. [Online]. <http://link.springer.com/10.1007/978-3-642-29044-2>
- [47] F.J. Fowler Jr, *Survey research methods*. Sage Publications, 2013.
- [48] M. Kasunic, "Designing an effective survey," *Carnegie Mellon University*, 2005.

- [49] RStudio, PBC, *RStudio: Integrated Development Environment for R*, 2025, version 2025.05.1+513. [Online]. <https://posit.co/download/rstudio-desktop/>
- [50] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2024, version 4.5.0. [Online]. <https://www.R-project.org/>
- [51] H. Wickham, *ggplot2: Elegant Graphics for Data Analysis*, 2023, version 3.5.2. [Online]. <https://ggplot2.tidyverse.org>
- [52] R.C. Schmidt, “Managing delphi surveys using nonparametric statistical techniques,” *Decision Sciences*, Vol. 28, No. 3, 1997, pp. 763–774. [Online]. <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-5915.1997.tb01330.x>
- [53] R.A. Fisher, “Statistical methods for research workers,” in *Breakthroughs in statistics: Methodology and distribution*. Springer, 1970, pp. 66–70.
- [54] J.W. Tukey, “Comparing individual means in the analysis of variance,” *Biometrics*, 1949, pp. 99–114.
- [55] M.G. Kendall and B.B. Smith, “The problem of m rankings,” *The Annals of Mathematical Statistics*, Vol. 10, No. 3, 1939, pp. 275–287.
- [56] L.J. Cronbach, “Coefficient alpha and the internal structure of tests,” *Psychometrika*, Vol. 16, No. 3, 1951, pp. 297–334.
- [57] J.L. Fleiss, “Measuring nominal scale agreement among many raters,” *Psychological Bulletin*, Vol. 76, No. 5, 1971, pp. 378–382.
- [58] P. Becker, L. Olsina, and M.F. Papa, “Especificación de métricas de mantenibilidad para mejorar código Java: Caso aplicado en un curso avanzado de ingeniería en sistemas,” in *Actas del XXI Congreso Nacional de Ingeniería Informática/Sistemas de Información (CoNaIISI), Tucumán, Argentina*, 2023, pp. 1–15.

## Appendix A. Preliminary catalog of metrics

This section presents the preliminary catalog of metrics validated in this work, derived from [9].

Category	Metric Code	Metric Name
API Performance (AP)	AP.1.1	Informational Response Rate
API Performance (AP)	AP.2.1	Successful Response Rate
API Performance (AP)	AP.2.2	Total Successful Requests
API Performance (AP)	AP.3.1	Redirection Response Rate
API Performance (AP)	AP.4.1	Requests Throughput
API Performance (AP)	AP.5.1	Average inbound data
API Performance (AP)	AP.5.2	Maximum inbound data
API Performance (AP)	AP.5.3	Minimum inbound data
API Performance (AP)	AP.6.1	Average outbound data
API Performance (AP)	AP.6.2	Maximum outbound data
API Performance (AP)	AP.6.3	Minimum outbound data
API Utilization (AU)	AU.1.1	Active inbound data
API Utilization (AU)	AU.1.2	Active outbound data
Caching (CA)	CA.1.1	Total Cache Hits
Caching (CA)	CA.1.2	Cache Hit Rate
Caching (CA)	CA.2.1	Total Cache Misses
Caching (CA)	CA.2.2	Cache Miss Rate
Errors (ER)	ER.1.1	Total Error Count
Errors (ER)	ER.1.2	Overall Error Rate
Errors (ER)	ER.2.1	Client Error Count
Errors (ER)	ER.2.2	Client Error Rate
Errors (ER)	ER.3.1	Server Error Count

Category	Metric Code	Metric Name
Errors (ER)	ER.3.2	Server Error Rate
Errors (ER)	ER.4.1	Total Other Errors
Errors (ER)	ER.4.2	Exception Count
Errors (ER)	ER.4.3	Total Authentication Errors
Errors (ER)	ER.4.4	Total Target Connection Errors
Latency and Response Time (RT)	RT.1.1	Internal Processing Time (IPT)
Latency and Response Time (RT)	RT.1.2	IPT (90th Percentile)
Latency and Response Time (RT)	RT.1.3	IPT (95th Percentile)
Latency and Response Time (RT)	RT.1.4	IPT (99th Percentile)
Latency and Response Time (RT)	RT.2.1	Total Response Time (TRT)
Latency and Response Time (RT)	RT.2.2	Maximum Response Time
Latency and Response Time (RT)	RT.2.3	Minimum Response Time.
Latency and Response Time (RT)	RT.2.4	Standard Deviation of Response Time
Latency and Response Time (RT)	RT.2.5	TRT (50th Percentile)
Latency and Response Time (RT)	RT.2.6	TRT (90th Percentile)
Latency and Response Time (RT)	RT.2.7	TRT (95th Percentile)
Latency and Response Time (RT)	RT.2.8	TRT (99th Percentile)
Latency and Response Time (RT)	RT.3.1	Upstream Response Time (UpRT)
Latency and Response Time (RT)	RT.3.2	UpRT (90th Percentile)
Latency and Response Time (RT)	RT.3.3	UpRT (95th Percentile)
Resource Utilization (RU)	RU.1.1	CPU Utilization Percentage
Resource Utilization (RU)	RU.2.1	Disk Utilization Percentage
Resource Utilization (RU)	RU.3.1	Memory Usage in the Last Hour
Resource Utilization (RU)	RU.3.2	Maximum Memory Utilization
Resource Utilization (RU)	RU.3.3	Memory Usage by Node
System Health (SH)	SH.1.1	Database Reachability Status
System Health (SH)	SH.2.1	Upstream Health Status
System Health (SH)	SH.3.1	Active Reading Connections
System Health (SH)	SH.3.2	Idle waiting connections
System Health (SH)	SH.3.3	Active writing connections
Traffic Monitoring (TM)	TM.1.1	Total APIs
Traffic Monitoring (TM)	TM.1.2	Total number of Products
Traffic Monitoring (TM)	TM.1.3	Total API Requests
Traffic Monitoring (TM)	TM.2.1	Total Data transferred
Traffic Monitoring (TM)	TM.2.2	Total Inbound data
Traffic Monitoring (TM)	TM.2.3	Total outbound data
Traffic Monitoring (TM)	TM.3.1	Throttling Errors

## Appendix B. Descriptive statistics of metrics' accuracy ratings

metric	mean	median	sd	min	max
TM.1.1	4.142857	4	1.0690450	2	5
TM.1.2	4.142857	4	0.6900656	3	5
TM.1.3	4.428571	4	0.5345225	4	5
TM.2.1	4.000000	4	1.0000000	2	5
TM.2.2	4.142857	4	0.6900656	3	5
TM.2.3	4.142857	4	0.6900656	3	5
TM.3.1	4.000000	4	1.0000000	2	5
SH.1.1	3.285714	3	0.7559289	2	4
SH.2.1	3.000000	3	1.0000000	2	4
SH.3.1	3.000000	3	1.0000000	2	4
SH.3.2	3.000000	3	1.0000000	2	4

metric	mean	median	sd	min	max
SH.3.3	3.142857	3	1.2149858	2	5
CA.1.1	3.714286	4	0.9511897	2	5
CA.1.2	4.000000	4	1.0000000	2	5
CA.2.1	3.714286	4	0.9511897	2	5
CA.2.2	3.857143	4	1.0690450	2	5
AU.1.1	3.714286	4	0.9511897	2	5
AU.1.2	3.714286	4	0.9511897	2	5
RT.1.1	3.571429	4	1.1338934	2	5
RT.1.2	2.571429	2	1.1338934	1	4
RT.1.3	2.571429	2	1.1338934	1	4
RT.1.4	2.571429	2	1.1338934	1	4
RT.2.1	3.857143	4	1.0690450	2	5
RT.2.2	3.857143	4	0.8997354	2	5
RT.2.3	3.571429	4	1.1338934	2	5
RT.2.4	4.000000	4	1.0000000	2	5
RT.2.5	2.857143	3	1.2149858	1	4
RT.2.6	2.857143	3	1.2149858	1	4
RT.2.7	2.857143	3	1.2149858	1	4
RT.2.8	2.857143	3	1.2149858	1	4
RT.3.1	3.571429	4	1.2724180	2	5
RT.3.2	3.000000	3	1.1547005	2	5
RT.3.3	3.000000	3	1.1547005	2	5
RU.1.1	3.714286	4	1.2535663	2	5
RU.2.1	3.428571	4	1.3972763	2	5
RU.3.1	3.857143	4	0.8997354	2	5
RU.3.2	3.714286	4	0.9511897	2	5
RU.3.3	3.857143	4	0.8997354	2	5
AP.1.1	4.000000	4	1.1547005	2	5
AP.2.1	4.285714	4	0.7559289	3	5
AP.2.2	4.142857	4	0.6900656	3	5
AP.3.1	3.857143	4	1.0690450	2	5
AP.4.1	4.142857	4	0.6900656	3	5
AP.5.1	3.428571	4	1.1338934	2	5
AP.5.2	3.428571	4	1.1338934	2	5
AP.5.3	3.285714	4	1.2535663	2	5
AP.6.1	3.285714	4	1.2535663	2	5
AP.6.2	3.428571	4	1.1338934	2	5
AP.6.3	3.571429	4	1.1338934	2	5
ER.1.1	4.428571	4	0.5345225	4	5
ER.1.2	4.571429	5	0.5345225	4	5
ER.2.1	4.571429	5	0.5345225	4	5
ER.2.2	4.000000	4	1.1547005	2	5
ER.3.1	4.142857	5	1.2149858	2	5
ER.3.2	4.571429	5	0.5345225	4	5
ER.4.1	3.714286	4	1.2535663	2	5
ER.4.2	4.142857	4	0.8997354	3	5
ER.4.3	4.000000	4	1.0000000	2	5
ER.4.4	3.857143	4	1.0690450	2	5

## Appendix C. Descriptive statistics of metrics' relevance ratings

metric	mean	median	sd	min	max
TM.1.1	4.285714	4	0.7559289	3	5
TM.1.2	4.285714	4	0.7559289	3	5
TM.1.3	4.571429	5	0.5345225	4	5
TM.2.1	4.285714	5	1.1126973	2	5
TM.2.2	4.285714	5	1.1126973	2	5
TM.2.3	4.285714	5	1.1126973	2	5
TM.3.1	4.571429	5	0.7867958	3	5
SH.1.1	3.571429	3	0.7867958	3	5
SH.2.1	4.428571	4	0.5345225	4	5
SH.3.1	3.714286	3	0.9511897	3	5
SH.3.2	4.000000	4	0.8164966	3	5
SH.3.3	4.142857	4	0.8997354	3	5
CA.1.1	3.571429	4	0.9759001	2	5
CA.1.2	3.857143	4	1.0690450	2	5
CA.2.1	3.857143	4	0.8997354	3	5
CA.2.2	3.857143	4	0.8997354	3	5
AU.1.1	3.428571	4	1.1338934	2	5
AU.1.2	3.428571	4	1.1338934	2	5
RT.1.1	4.285714	4	0.7559289	3	5
RT.1.2	3.571429	3	0.7867958	3	5
RT.1.3	3.571429	3	0.7867958	3	5
RT.1.4	3.571429	3	0.7867958	3	5
RT.2.1	4.571429	5	0.5345225	4	5
RT.2.2	4.285714	4	0.7559289	3	5
RT.2.3	4.142857	4	1.0690450	2	5
RT.2.4	4.285714	4	0.7559289	3	5
RT.2.5	3.857143	4	0.8997354	3	5
RT.2.6	3.857143	4	0.8997354	3	5
RT.2.7	3.857143	4	0.8997354	3	5
RT.2.8	3.857143	4	0.8997354	3	5
RT.3.1	4.000000	4	1.1547005	2	5
RT.3.2	3.714286	4	1.1126973	2	5
RT.3.3	3.714286	4	1.1126973	2	5
RU.1.1	4.142857	4	1.0690450	2	5
RU.2.1	3.714286	4	1.2535663	2	5
RU.3.1	4.285714	4	0.4879500	4	5
RU.3.2	4.000000	4	0.8164966	3	5
RU.3.3	4.142857	4	0.3779645	4	5
AP.1.1	3.857143	4	1.0690450	2	5
AP.2.1	4.000000	4	0.5773503	3	5
AP.2.2	4.000000	4	0.8164966	3	5
AP.3.1	3.857143	4	1.0690450	2	5
AP.4.1	4.285714	4	0.7559289	3	5
AP.5.1	3.714286	4	1.1126973	2	5
AP.5.2	3.571429	4	0.9759001	2	5
AP.5.3	3.571429	4	1.2724180	2	5
AP.6.1	3.571429	4	1.2724180	2	5
AP.6.2	3.714286	4	1.1126973	2	5
AP.6.3	3.857143	4	1.0690450	2	5
ER.1.1	4.428571	4	0.5345225	4	5
ER.1.2	4.571429	5	0.5345225	4	5
ER.2.1	4.428571	4	0.5345225	4	5
ER.2.2	4.000000	4	1.1547005	2	5

metric	mean	median	sd	min	max
ER.3.1	4.000000	4	1.1547005	2	5
ER.3.2	4.571429	5	0.5345225	4	5
ER.4.1	4.000000	4	1.0000000	2	5
ER.4.2	4.000000	4	0.8164966	3	5
ER.4.3	4.428571	5	0.7867958	3	5
ER.4.4	4.142857	4	0.8997354	3	5

## Appendix D. Tukey HSD pairwise comparisons

Expert pair	diff	lwr	upr	p adj
E2-E1	-0.28813559	-0.68469491	0.1084237	0.3237969
E3-E1	0.22033898	-0.17622033	0.6168983	0.6521410
E4-E1	0.30508475	-0.09147457	0.7016441	0.2564151
E5-E1	-0.20338983	-0.59994915	0.1931695	0.7328451
E6-E1	0.96610169	0.56954238	1.3626610	0.0000000
E7-E1	1.30508475	0.90852543	1.7016441	0.0000000
E3-E2	0.50847458	0.11191526	0.9050339	0.0031533
E4-E2	0.59322034	0.19666102	0.9897797	0.0002408
E5-E2	0.08474576	-0.31181355	0.4813051	0.9956855
E6-E2	1.25423729	0.85767797	1.6507966	0.0000000
E7-E2	1.59322034	1.19666102	1.9897797	0.0000000
E4-E3	0.08474576	-0.31181355	0.4813051	0.9956855
E5-E3	-0.42372881	-0.82028813	-0.0271695	0.0274732
E6-E3	0.74576271	0.34920340	1.1423220	0.0000010
E7-E3	1.08474576	0.68818645	1.4813051	0.0000000
E5-E4	-0.50847458	-0.90503389	-0.1119153	0.0031533
E6-E4	0.66101695	0.26445763	1.0575763	0.0000236
E7-E4	1.00000000	0.60344068	1.3965593	0.0000000
E6-E5	1.16949153	0.77293221	1.5660508	0.0000000
E7-E5	1.50847458	1.11191526	1.9050339	0.0000000
E7-E6	0.33898305	-0.05757627	0.7355424	0.1503445

## Authors and affiliations

Eder dos Santos  
e-mail: [esantos@uarg.unpa.edu.ar](mailto:esantos@uarg.unpa.edu.ar)  
ORCID: <https://orcid.org/0000-0001-6729-0303>  
Instituto de Tecnología Aplicada, Universidad  
Nacional de la Patagonia Austral, Argentina

Sandra Casas  
e-mail: [sicasas@uarg.unpa.edu.ar](mailto:sicasas@uarg.unpa.edu.ar)  
ORCID: <https://orcid.org/0000-0002-8289-6132>  
Instituto de Tecnología Aplicada, Universidad  
Nacional de la Patagonia Austral, Argentina