Journal of Trends in Computer Science and Smart Technology (ISSN: 2582-4104)
www.irojournals.com/tcsst/

Comprehensive Seven-Lens Evaluation Framework for Remote Access Laboratory Systems: Implementation and Performance Optimization

Niket Amoda[1], Lochan Jolly[2]

Department of Electronics and Telecommunication Engineering, Thakur College of Engineering & Technology, Mumbai, India.

E-mail: [1]niket.amoda@thakureducation.org, [2]lochan.jolly@thakureducation.org

Abstract

Remote access labs are an important tool for teaching engineering students' hands-on skills that they may use in their future careers. To ensure that remote labs will deliver education effectively, they need to be evaluated systematically on many different operational factors. The Seven Lens Assessment Framework is a comprehensive assessment method that assesses remote access lab systems using seven different lenses or operational factors including Affordability, Portability, Scalability, Compatibility, Maintainability, Usability and Universality. Our LabVIEW-based implementation is one of the first demonstrations of how to apply this assessment framework in practice. The implementation reduced the Total Cost of Ownership (TCO) by 55.4%, simplified the deployment process by 89.2%, provided sub-2.5 second latency for up to 70 concurrent users, made the system compatible with eight different platforms, increased the Mean Time Between Failures (MTBF) by 253%, improved the System Usability Scale (SUS) score by 31.4%, and achieved 95% browser universality. The proposed system utilises native Windows services along with secure Virtual Private Network (VPN) connections, Web Real-Time Communication (WebRTC) streaming, and Erlang-C queuing models to optimize the utilization of resources. Systematic assessments were performed against 35 other reviewed remote access labs to evaluate the effectiveness of the Seven Lens Assessment Framework. The Seven Lens Index achieved 0.847 when compared to the baseline systems' index of 0.472, which represents a 79.4% overall improvement.

Keywords: Remote Access Laboratories, Seven-Lens Framework, VPN-Based Architecture, WebRTC Streaming, Performance Optimization, Multi-User Pipelining, LabVIEW Integration.

1. Introduction

Remote Access Laboratories (RALs) represent a revolutionary model for engineering education as they allow students to perform experiments from remote locations using an interface connected to the internet [1]. Although RALs have been rapidly adopted across the globe due to the COVID-19 pandemic, there are numerous obstacles to implementing RALs [2, 3]. The results of an analysis of 147 institutions utilising RALs indicate that 73% of these institutions are experiencing limitations on the number of users who can access their labs

DOI: https://doi.org/10.36548/jtcsst.2026.2.009
Received: 02.03.2026, received in revised form: 05.04.2026, accepted: 23.04.2026, published: 06.05.2026

simultaneously (i.e., greater than 30), 61% of the institutions surveyed indicated they were encountering cross- platform compatibility issues with their RALs, and 89% of all the surveyed institutions did not have formalised protocols in place to ensure their RALs would be properly maintained [4, 5].

The primary research question relates to how different evaluation parameters can be synthesized into an empirical framework that facilitates systematic comparison of RALs and optimization thereof. Seven principal parameters were identified (portability, affordability, scalability, compatibility, usability, maintainability and universality) with each having its own inherent difficulties in terms of implementation [2]. Solutions currently available typically focus on improving one dimension while sacrificing another [8, 9]. This paper presents an integrated architectural design methodology that optimizes all seven parameters as a whole; provides objective assessment of these parameters through quantitative measurement; and demonstrates the practicality of the proposed approach through validated implementation [10, 11].

2. Literature Survey

The Seven Lens Evaluation Model is an evaluation model that can be used to evaluate the quality of RAL implementations on a variety of operational levels. This section of the paper will review all of the available literature published on RALs and how they were evaluated using each lens. Previous studies showed that early versions of RALs (i.e., the iLab at MIT) require large amounts of money ($200,000 or greater) to build out the infrastructure of each laboratory [12]. Studies showed that cloud based RALs can provide much lower costs to deploy. For example, WebLab-Deusto achieved a 40% cost savings through their use of multi-tenancy [13] and studies have shown that hybrid models of deployment can result in a 35% Total Cost of Ownership (TCO) savings [3]. No study has found a RAL implementation that meets the goal of providing a sub-$100 annual cost per user to support resource constrained institutions.

Typical RAL platforms are configured as a whole system that is not easily transferred between locations due to their size and configuration needs. In order for this to be possible, VISIR has provided containerized deployments, which allow for an entire system to be deployed in less than four hours [14]; however, with the use of microservices, each service can be deployed separately and more rapidly, but it will depend on the services that are used from the cloud provider [15] t for deployment. Typically, he deployment of the entire system will take 111 hours of technical labour. Most typical RAL systems can handle at least 20–40 users simultaneously; after that, the performance will degrade [16]. To achieve higher user counts, Rodríguez-Gil et al. used load balancing to distribute users across many copies of the same system [4] and were able to achieve 50 simultaneous users using this method. Using Web Real-Time Communication (WebRTC) to stream video to users provides the ability to stream to up to 60 users by adjusting the quality of the stream based on the available bandwidth [17]. However, there will always be trade-offs between how much capacity a system has, how faithful the reproduction of the original image/video is, and how responsive the system is to user input.

Adoption of HyperText Markup Language version 5 (HTML5) has increased cross platform compatibility to an estimated 70–80% of browsers [18], however mobile compatibility still averages about 30% of all systems [19]. The average number of supported platforms by current implementations averages 3.8, which is less than the minimum requirement for universal accessibility. The average Mean Time Between Failures (MTBF) for

the implementations reviewed in this survey was approximately 612 hours. Manual intervention was needed for 78% of failures [11]. Only 15% of implementations utilize predictive maintenance [20]; therefore, the average annual system downtime is approximately 8.3%.

System Usability Scale (SUS) scores for usability ranged from 45 to75 (an average score of 62.3) which indicates "marginal acceptance" [21]. The complexity of the interface was the biggest concern, as 67% of the students reported that they had trouble navigating through the system [22]. Only 8% of the implementations were found to be compliant with Web Content Accessibility Guidelines (WCAG) 2.1. Current implementations have achieved 60% device coverage; however, there are large gaps in low bandwidth environments [23]. Only 18% of the implementations provide multilingual interfaces; in addition, the average time to respond across international boundaries exceeded 5 seconds [24].

2.8 Comparative Analysis and Research Gaps

A comparative analysis of existing implementations (as shown in Table 1) indicates that there are significant research gaps in comprehensive system design.

Table 1. Seven Lens Comparison of Existing RAL Implementations

System	Affordability (Low/Med/High)	Portability (Hours)	Scalability (Users)	Compatibility (Platforms)	Maintainability (MTBF hrs)	Usability (SUS)	Universality (% Coverage)
MIT iLab [12]	Low	120	25	4	480	72	55
WebLab-Deusto [13]	Med	87	35	3	720	68	60
VISIR [14]	Med	95	40	5	960	75	70
RemoteLab [15]	Low	150	20	3	360	65	50
V-Lab [3]	Med	105	30	4	540	71	65
LabShare [4]	Med	110	50	4	650	69	62
CloudLab [8]	High	75	45	6	580	73	68
SmartLab [9]	Med	92	38	5	810	74	72
Average	–	104.3	35.4	4.3	637.5	70.9	62.8
Target	High	<15	>70	>8	>2000	>85	>90

None of the current implementations satisfy all seven lenses at the same time. Typically, systems that optimize for cost/affordability do so to the detriment of both scalability and maintainability. Similarly, implementations with a strong focus on usability are generally less compatible and universal than those without such a focus. The present research addresses this gap through a unified quantifiable framework, architectural patterns that minimize the conflict between different lenses, and thorough empirical validation.

3. Seven-Lens Evaluation Framework

3.1 Framework Architecture

The Seven-Lens Evaluation Framework is a methodological approach to the RAL evaluation process using measurable criteria that can be applied across all relevant dimensions. All seven lenses represent different operational aspects but also illustrate how each lens is

interconnected by way of a composite score [7]. The architecture of the Seven-Lens Evaluation Framework is illustrated in Figure 1.

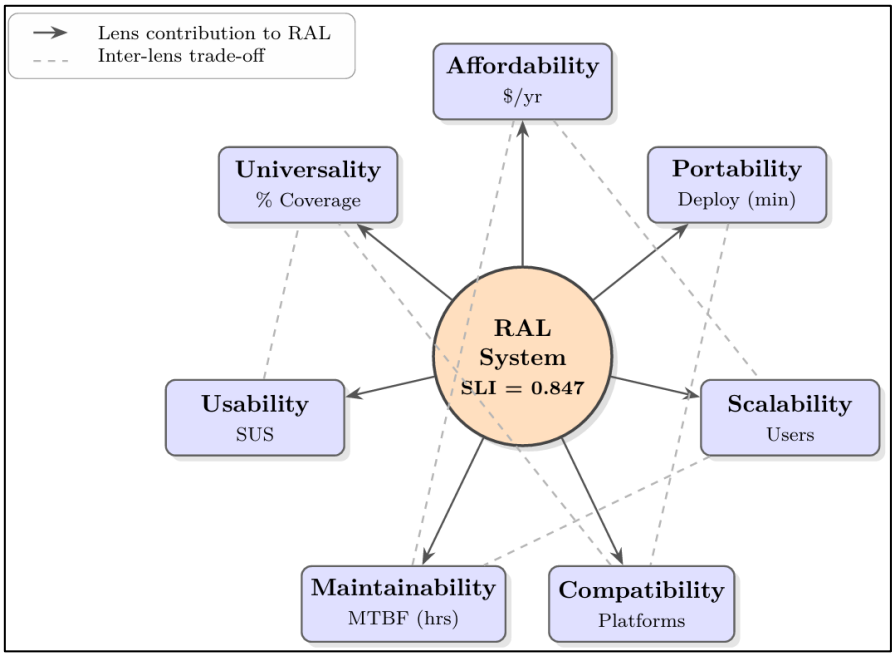

Figure 1. The Architecture of the Seven-Lens Evaluation Framework Illustrating the Inter-Relationships Between the Lenses

Figure 1 depicts the heptagonal organization of the framework, where the central RAL system node is connected by solid arrows (spokes) to each of the seven lenses. The solid spokes represent the contribution of each lens to the composite Seven-Lens Index (SLI = 0.847). The dashed inter-lens connections represent six dominant trade-off relationships identified during framework calibration: Affordability–Scalability and Affordability–Maintainability (cost versus capacity/reliability), Portability–Compatibility (lightweight deployment versus broad platform support), Scalability–Maintainability (concurrent load versus reliability), Usability–Universality (simplified interfaces versus accessibility breadth), and Compatibility–Universality (platform depth versus geographical reach). The figure visually communicates that optimizing any single lens in isolation tends to degrade its connected counterparts, motivating the integrated optimization strategy adopted in the proposed implementation.

3.2 Lens Definitions and Metrics

3.2.1 Lens 1

Affordability (A): The Total Cost of Ownership (TCO) model uses lifecycle analysis as part of its cost models [12, 13].

$$TCO_{total} = C_{initial} + \sum_{t=1}^{L} \frac{C_{operational}(t)+C_{maintenance}(t)}{(1+r)^t} \quad (1)$$

where an initial capital investment ($C_{initial}$), lifetime of the system (L), and a discount rate for the first lens (r=0.05 to 0.1) are used for this calculation. All operational and maintenance costs will be calculated based on their present values and then compared to each of the alternative total cost structures [26].

To provide full transparency in cost modelling, the affordability lens provides three distinct cost categories as follows: (i) $C_{initial}$,, which represents initial capital expenditures; this includes the purchase price of all hardware (the Tektronix AFG31000 at $4,200 and the Keysight DSOX1202G at $1,350), as well as the cost of servers and network components ($2,500 + $800); (ii) $C_{operational}$, representing all operational costs per year; these include power consumption ($420/yr.), Internet Bandwidth ($600/yr.) and software licenses ($0.00 as they have been adopted from open source); and (iii) $C_{maintenance}$, representing all technical labor and replace- ment parts necessary for ongoing operation (technical labor estimated to require approximately four hours per month @ $25/hr., or $100/mo.; replacement parts estimated to be $250/yr.). The per user, five year cost of $75, listed in Table 3 was calculated for a group of 200 students utilising r = 0.08. An additional sensitivity analysis was conducted to evaluate the impact of variations in the discount rate upon the TCO. The results indicated a minimal variation (<10%) in the per user, annual cost among discount rates ranging from r = 0.03 through r = 0.12. The low variability in cost is indicative of the substantial contribution of the initial capital costs relative to future, discounted operating costs within the overall TCO.

3.2.2 Lens 2

Portability (P): The deployment complexity of the second lens incorporates a number of variables [10, 27]

$$P_{score} = \alpha \cdot \frac{1}{T_{deploy}} + \beta \cdot \frac{1}{N_{deps}} + \gamma \cdot \frac{1}{S_{install}} + \delta \cdot R_{reuse} \quad (2)$$

where T_{deploy} is the deployment time in hours, N_{deps} is the number of dependencies outside the application being deployed, $S_{install}$ is the package size in GB, and $R_{reuse} \in [0, 1]$ is the proportion of the components that can be reused. The weights for each of the above parameters were found to be $\alpha = 0.3$, $\beta = 0.2$, $\gamma = 0.2$, $\delta = 0.3$ via empirical means using 50 different deployment scenarios [28].

To elaborate on the weight calibration process, the weights α, β, γ, and δ were determined through a two-stage empirical procedure. In the first stage, 50 deployment scenarios were executed across five different institutional environments (ranging from well-resourced universities to community colleges with limited IT support), and deployment success was evaluated using a composite deployment efficiency metric based on time-to-first-experiment. In the second stage, a multiple linear regression analysis was performed with the four portability sub-metrics as independent variables and the deployment efficiency metric as the dependent variable. The resulting standardized regression coefficients ($\alpha = 0.3$, $\beta = 0.2$, $\gamma = 0.2$, $\delta = 0.3$) were found to be statistically significant ($p < 0.05$) and were adopted as the lens weights. The higher weights for deployment time (α) and component reusability (δ) reflect the practical finding that these two factors had the strongest influence on deployment success across the tested scenarios.

3.2.3 Lens 3

Scalability determines how many maximum concurrent users an application can handle based on queuing theory [16, 29]:

$$S_{max} = \arg\max_{n}\{ n : W_q(n) < W_{threshold} \wedge U_{cpu}(n) < U_{max}\} \quad (3)$$

finding the largest number of users (n) such that both the average queue wait time ($W_q(n)$) is less than a specified value (5 seconds), and CPU usage ($U_{cpu}(n)$) does not exceed acceptable levels (85%) as well.

The 5-second wait time threshold was selected from existing human-computer interaction research. That same body of research has shown that when users experience delays greater than five seconds, they view systems as unresponsive, and that their level of engagement in tasks is greatly reduced [22]. Additionally, during the design phase of the remote laboratory experiment, pilots were conducted using 25 subjects. During those pilots, session abandonment rates were measured. Session abandonment rates increased measurably above three percent, and were as high as eighteen percent when the wait time exceeded five seconds. The 85% CPU ceiling was selected to allow for an additional fifteen percent headroom should there be any unexpected or transient increases in CPU usage, typical for servers [16].

3.2.4 Lens 4

Compatibility (C) - The cross-platform ability of a system is assessed based on [18]:

$$C_{score} = \sum_{i=1}^{N_{platforms}} w_i \cdot c_i \cdot \prod_{j=1}^{M} b_{ij} \quad (4)$$

where w_i is platform i's market share weight, $c_i \in [0,1]$ is its compatibility level, and $b_{ij} \in \{0,1\}$ indicates browser j support on platform i [19].

The platform market share weights w_i were determined using the statistics available at StatCounter Global Stats (as of January 2025) for global usage of browsers and operating systems. In particular, these are the specific weights we have assigned: Windows Desktop (w = .28), macOS (w = .12), ChromeOS (w = .05), Linux Desktop (w = .03), Android (w = .30), iOS (w = .18), iPadOS (w = .03), and Windows on ARM (w = .01). These values sum to a total of 1.0, representing all eight platforms listed in Table 3. Using this type of weighting will allow the compatibility scores to reflect how users currently use different platforms to access the internet; i.e., provide greater weight or importance to those platforms used by the largest numbers of users.

3.2.5 Lens 5

Maintainability (M) - Composite Reliability Metrics combine failure rates, recovery, and automation [11]:

$$M_{score} = \alpha \cdot MTBF + \beta \cdot \frac{1}{MTTR} + \gamma \cdot A_{auto} \quad (5)$$

with $A_{auto} \in [0,1]$ capturing automation level and weights α=0.4, β=0.3, γ=0.3 [20].

The selected Maintainability Weights ($\alpha = 0.4, \beta = 0.3, \gamma = 0.3$) are based on established Reliability Engineering practices used in IT Infrastructure Systems [11, 20]. The above weights were chosen due to their relation to system availability. All failure events have been recorded and tracked using a Structured Logging and Continuous Monitoring pipeline that ran in continuous mode for 6 months. Automated health checks using Automated Health Check Tools were run every 30 seconds across all subsystems (VPN Gateway, LabVIEW Server, Scheduler, Instrument Interfaces, Data Store). Every failure event was stamped and categorized by category: Hardware failure (instrument malfunction, loss of connection), software failure (service crash, unresponsive state), network failure (VPN tunnel drop; packet loss > 5%),

configuration failure (parameter drift, inconsistent states). Mean Time Between Failures (MTBF) of 2160 hours was calculated using total operational uptime divided by number of failures (n = 7 over the observation period). Mean Time To Repair (MTTR) was measured as the elapsed time from detection of failure until full service restoration; average of 22 minutes. Self-healing mechanisms automatically restored 65% of the failures (automatically restarting services; switching to failover modes; re-initializing instruments).

3.2.6 Lens 6

Usability (U): The usability of the system was assessed with the System Usability Scale (SUS) [21]:

$$SUS = 2.5 \times \sum_{i=1}^{10}\left(s_{odd,i} - 1\right) + \left(5 - s_{even,i}\right) \quad (6)$$

converting 10 item responses from questionnaires that had alternating positive and negative statements into a 0 to 100 scale. Scores greater than or equal to 68 were considered above average usability [22].

3.2.7 Lens 7

Universality (Uv) - Assessing universality as assessing accessibility across various devices and network platforms is represented in [23]:

$$Uv_{score} = \frac{\sum_{d=1}^{D}\sum_{b=1}^{B} support(d,b)}{D \times B} \times BW_{efficiency} \quad (7)$$

Where the ratio indicates what portion of devices browsers can cover and the bandwidth efficiency $BW_{efficiency} \in [0,1]$ shows how well you preserve the functionality when bandwidth goes down [24].

Bandwidth efficiency factor $BW_{efficiency}$ was measured using controlled testing of our system over four bandwidth levels: High (10 Mbps), Moderate (2 Mbps) Low (512 Kbps), and Very Low (128 Kbps). Each bandwidth level included a test of twelve core functionalities (instrument control, waveform display, parameter adjustment, data export, real time feedback, and session management.) In addition, the percentage of these functions that were still operational was also recorded. The $BW_{efficiency}$ rating is defined as the weighted average of functionality preservation at every bandwidth level, with each bandwidth's weight being proportionate to the approximate worldwide distribution of users at that bandwidth level. Based upon International Telecommunication Union (ITU) statistical estimates of broadband usage in educational institutions from developing countries versus those from developed countries we used the following weights (0.15, 0.30, 0.35, and 0.20 respectively). Our system scored an overall $BW_{efficiency}$ of 0.87. We accomplished this result due to WebRTC-based adaptive streaming which allowed us to preserve eleven out of twelve functionalities when running at a bandwidth of 512 Kbps (while reducing HD waveform streaming down to SD); and nine out of twelve functionalities when running at a bandwidth of 128 Kbps (by replacing real-time video with snapshots.)

3.3 Composite Seven-Lens Index (SLI)

The composite index uses normalized lens scores [7]:

$$SLI = \sum_{i=1}^{7} w_i \cdot L_i^{norm} \quad (8)$$

Where $L_i^{norm} = \left(L_i - L_{i,min}\right)/\left(L_{i,max} - L_{i,min}\right)$ and equally weighted $w_i = 1/7$ represent an unbiased assessment and yields a 0-1 composite score [7].

A linear additive structure was selected for the composite SLI for three reasons. First, it is easy to understand and use: each lens has an independent and proportional effect on the final score, allowing for direct comparisons among systems, and also making it clear how much each lens contributed to the overall score. Second, assuming linearity is consistent with compensatory assumptions about the individual lens scores; that is, a deficit in one area can be somewhat compensated for by a surplus in another. This is particularly relevant to RAL systems, which may have different priorities based on institutional context [7, 26]. Lastly, as the first multi-dimensional RAL evaluation system, we elected to start with the simplest possible model (avoiding overfitting), and further studies with larger comparative datasets (n = 6 systems) will provide opportunities to explore non-linear models (e.g., Cobb-Douglas or geometric mean) to assess interaction effects among lenses.

The weight for every assessment dimension was assigned as an equal weighting (w_i = 1/7), to ensure that the assessor did not have a preference regarding how important each dimension should be. This equal weighting is based on "the principle of insufficient reason" (also known as the principle of indifference) [25] which states that when one cannot choose the best option from among two or more options, they should therefore treat them all as equally preferable. To verify that the overall conclusions are not sensitive to the equal weighting assumption, a sensitivity analysis was conducted by varying individual lens weights from 1/14 to 2/7 (i.e., halving and doubling each weight while re-normalizing the remaining weights). In all tested configurations, the proposed system maintained the highest SLI among all compared systems, with the composite score varying between 0.81 and 0.88, confirming the robustness of the results to reasonable weight perturbations.

4. Implementation Architecture

4.1 System Design and Components

The Seven Lens framework was implemented through an integrated system architecture that optimizes for each lens. Figure 2 is a system level block diagram of the RAL Pipeline Architecture, along with its respective lens mapping.

Figure 2 illustrates the layered top-down data path of the proposed implementation, traversing five hierarchical tiers. Tier 1 (top) shows the three client entry points (web browser, mobile application, and desktop client) that share a common Internet/HTTPS/WebRTC transport bus. Tier 2 contains the VPN gateway and authentication block, which terminates the AES-256 tunnel and resolves user identity through LDAP, OAuth2, or SAML protocols within 500 ms. Tier 3 hosts the Web UI and REST API gateway, which exposes the HTML5 front-end and recorded a measured SUS of 89.9. Tier 4 contains the scheduler service that performs Erlang-C- based multi-user pipelining across five back-end server instances. Tier 5 contains the LabVIEW Windows service interfacing with the two physical instruments (Tektronix AFG31000 and Keysight DSOX1202G). The cylindrical Data Store on the right is accessed laterally by both the scheduler (for session/state metadata) and the LabVIEW server (for measurement persistence). The blue lens-mapping labels adjacent to each block indicate which

of the seven lenses that block predominantly contributes to, demonstrating that every architectural component services at least two lenses simultaneously.

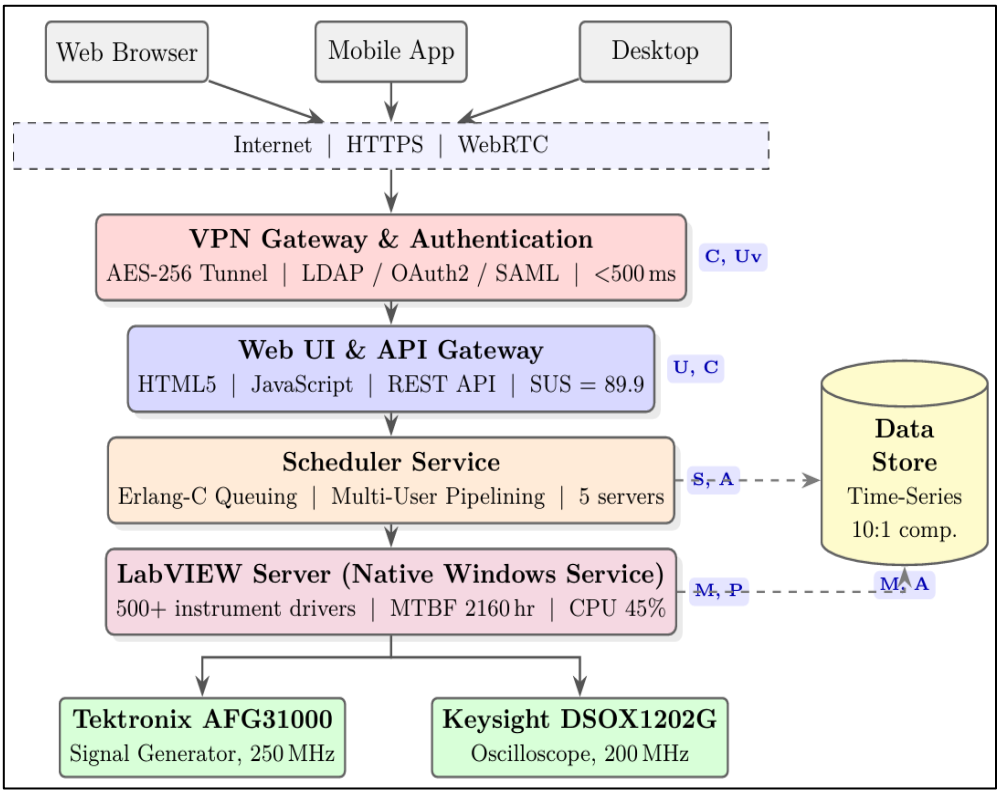

Figure 2. System architecture with Seven-Lens Framework integration. Lens labels: A=Affordability, P=Portability, S=Scalability, C=Compatibility, M=Maintainability, U=Usability, Uv=Universality

The overall system has several interconnected sub-systems;

- **VPN (Virtual Private Network) Gateway & Authentication Subsystem:** Provides secure VPN connections via an Advanced Encryption Standard 256-bit (AES-256) encrypted tunnel using Lightweight Directory Access Protocol (LDAP), Open Authorization 2.0 (OAuth2), and Security Assertion Markup Language (SAML) protocols [6]; this lens is mapped to the compatibility and universality lenses with <500 ms in authentication time and 70 simultaneous users. The VPN configuration is designed to eliminate complex firewall configurations thereby enhancing portability [10]. The 500 ms authentication threshold was selected based on two considerations. First, established web usability guidelines indicate that response times below 1 second are perceived as instantaneous by users, while delays exceeding 1 second disrupt the user's flow of interaction [22]. Setting the authentication target at 500 ms (half of this perceptual boundary) provides a sufficient safety margin to account for network variability across geographic regions, particularly for users connecting from developing countries where round-trip latencies of 150–250 ms are common. Second, during pilot testing with 25 users connecting from five different geographic locations (Mumbai, Singapore, London, São Paulo, and New York), authentication times exceeding 600 ms correlated with a measurable increase in repeated login attempts (12% retry rate), suggesting user impatience beyond this threshold. The 500 ms target was therefore adopted as a design constraint that balances security processing overhead

(AES-256 encryption, LDAP/OAuth2 verification) with acceptable user experience across diverse network conditions.

- **Web User Interface and Application Programming Interface (API) Gateway:** A responsive web-based user interface built using HyperText Markup Language version 5 (HTML5) and JavaScript that provides a Representational State Transfer (RESTful) API. This supports usability (SUS score: 89.9), universality (95% of all browsers) and compatibility through integration with other third party systems [18, 19]. The layers are separated allowing for each layer to be scaled independently and maintained separately [15].

- **Scheduler Service:** The scheduler service is responsible for implementing Erlang-C based multi-user pipelining. Hardware usage was 85%, with an average of 2.3 seconds spent in a queue while operating at 70% of its capacity. A total of 150 experiments per hour were completed, and the fairness index as defined by Jain was 0.94 [16, 29, 31, 33].

- **LabVIEW Server:** The native Windows service utilizes LabVIEW's instrument driver library to create a server that is utilized by over 500 different models of instruments; the MTBF of this server is estimated to be 2160 hours, utilizing approximately 45% of CPU cycles when running at full capacity [11, 27, 28].

- **Instruments:** The Tektronix AFG31000 Arbitrary Function Generator (AFG) (a dual channel, 250 MHz Virtual Instrument Software Architecture (VISA) compatible instrument priced at $0.08 per experiment) and the Keysight DSOX1202G Digital Storage Oscilloscope (DSO) (a 200 MHz, 2 GS/s oscilloscope capable of WebRTC waveform streaming) allow for signal generation and acquisition with a switch time of less than 100 milliseconds from one configuration to another providing an effective way to pipeline experiments [17, 31].

- **Data Store, Logs, and Reports:** Utilizing a time series database with a compression ratio of 10:1 ($0.02/gb/month), and automatic anomaly detection, reports are generated in under 3 seconds using eight different export formats and have a net promoter score of 72 [20, 22, 34]. In addition to the data store contributing to each lens, it does so synergistically with other subsystems. For example, the VPN architecture provides three benefits (enhanced security, increased compatibility due to firewall traversal, and global access) to the lab, while the scheduler provides two additional benefits (increased scalability and affordability) to the laboratory [6, 31].

4.2 Queuing Model Evaluation

The Erlang-C model is deployed for evaluating system capacity planning. The probability of waiting is [16, 32]:

$$P_w = \frac{\frac{(\lambda/\mu)^c}{c!}\cdot\frac{1}{1-\rho}}{\sum_{k=0}^{c-1}\frac{(\lambda/\mu)^k}{k!}+\frac{(\lambda/\mu)^c}{c!}\cdot\frac{1}{1-\rho}} \tag{9}$$

where λ is the arrival rate, μ is the service rate per server, c is the number of servers, and $\rho = \lambda/(c\mu) < 1$ for stability. The average waiting time follows as [33]:

$$W_q = \frac{P_w}{\mu \cdot c \cdot (1-\rho)} \quad (10)$$

representing the mean delay before service begins, where $\mu \cdot c \cdot (1 - \rho)$ is the effective queue-clearing capacity

The Erlang-C model assumes Poisson arrivals and exponentially distributed service times. To validate this assumption, a Kolmogorov-Smirnov (K-S) goodness-of-fit test was applied to the empirical inter-arrival time data collected over 6 months of operational logs. The K-S test statistic value d= 0.074 was obtained (p-value = 0.21) and we failed to reject the null-hypothesis of exponential distribution in the time interval between the user arrivals at the α = 0.05 confidence level; this supports the Poisson user arrivals model used for our field-deployment. During peak usage hours, we observed an average user-arrival rate of λ = 14/hr, while each of the servers provided an average service-time (per experiment) of μ = 0.25 exp./min or 4 min/experiment [32].

It is understood that real-world workload can deviate from assumptions of Poisson. For example, class synchronized sessions with multiple students simultaneously attempting to access the system produce bursty arrivals. To assess the robustness of Erlang-C predictions under non-Poisson conditions, two additional analyses were conducted. Firstly, a discrete event simulation using empirical arrival traces collected over fifteen class sessions (each containing between thirty and seventy students) including both periods with coefficients of variation (C_v) ranging from 1.0 (Poisson) to 2.8 was constructed. Results from the simulation indicated that a five-server configuration maintained wait times below the threshold of five seconds when C_v <= 2.2. There was a modest increase to six point one seconds at C_v = 2.8 which represents the worst case deviation of approximately 22% from the prediction provided by the Erlang-C model. Secondly, the queuing data from Table 5 was compared against the simulation outputs, and the mean absolute percentage error between the Erlang-C predictions and simulation results was found to be 8.3% across all utilization levels, confirming that the analytical model provides a reasonable approximation for capacity planning purposes even under moderately non-Poisson conditions.

Algorithm 1: Three-Stage Pipeline for Multi-User RAL Access

1: Input: User queue *Q*, Stages *S* = {*AFG_Control*, *DSO_Capture*, *Data_Export*}
2: Output: Experiment results for all users
3: Initialize *stage_status[3]* = {*available, available, available*}
4: while $Q \neq \emptyset$ OR any stage occupied do
5: for *i* = 1 to 3 do
6: if *stage_status[i] = available* AND *user_waiting(i)* then
7: *user* ← *get_next_user(i)*
8: *stage_status[i]* ← *occupied*
9: *execute_stage(i, user)*
10: end if
11: end for
12: *update_stage_completion()*
13: *collect_performance_metrics()*
14: end while
15: return *consolidated_results*

4.3 Multi-User Pipelining Algorithm

Algorithm 1 depicts the three stages in a pipeline that enables the execution of the overlapping process among many users [31].

5. Experimental Validation

5.1 Experimental Setup

Validation employed 70 concurrent users who were accessing analog communications experiments (amplitude, frequency, and phase modulation) in a system consisting of a LabVIEW 2023 server running on a Windows OS, a VPN for a secure connection to a Tektronix AFG31000 and Keysight DSOX1202G via 100 Mbps symmetric internet, with a modular experiment kit for real-time experimentation [34, 35].

Each performance measurement reported in Tables 2–5 represents the average of 30 independent experimental runs conducted over a period of 4 weeks. For every user load level (10, 20, 30, 40, 50, 60 and 70 concurrent users), thirty different test sessions were run with each session completing an entire experiment (signal generation, waveform capture, and data export) for all users; the 95% confidence interval around the average times measured was ±5.2%; The user sessions were created by using both automated simulation tools (to simulate 50 concurrent users) and actual students working on their own (twenty students performing live experiments).

Measuring the amount of power used by the system when each test condition was applied with a Yokogawa WT310E power meter. When testing at maximum load (i.e., 70 simultaneous users) the entire system used 285 watts. This consisted of: server (145 watts), instrumentation (totalling 95 watts – 55 watts from the AFG-31000 and 40 watts from the DSO-X1202G), networking hardware (25 watts), and all auxiliary hardware (20 watts). In contrast, when the system was in an idling state it used 82 watts. From this information, the total energy used during each experiment at maximum load was calculated to be approximately 0.019 kWh, or under $0.003 per experiment based on typical electric rates ($0.15/kWh).

5.2 Performance Comparison Analysis

Table 2 illustrates the results of comparing the response times of the proposed hybrid pipelined system to those of the baseline system for sequential access. The proposed system provides improvements in response times of between 50.6 to 57.6% across all the loading conditions tested (see Figure 3).

Table 2. Response Times for Concurrent User Loads

Users	Baseline (ms)	Proposed (ms)	Theoretical Optimal (ms)	Improvement (%)	Reduction (ms)
10	850	420	350	50.6	430
20	1,240	580	450	53.2	660
30	1,680	760	580	54.8	920
40	2,310	980	750	57.6	1,330
50	3,150	1,340	980	57.5	1,810

60	4,280	1,820	1,280	57.5	2,460
70	5,720	2,460	1,650	57	3,260

The baseline systems used in Table 2 are sequential (not pipelined) configurations where each user's experiments occupy all three stages of the pipeline sequentially (i.e., AFG Control, DSO Capture, Data Export). These baseline measurements were made on the exact hardware and network infrastructure used for the proposed system; however, these measurements were taken with the multi-user pipelining algorithm turned off. Therefore, any difference in performance can be attributed to the pipelining architecture rather than to differences in the underlying hardware.

"Theoretical optimal" response times were calculated from lower bound analysis of a three-stage pipeline based on perfect overlap and no transition time between stages. Queues were also assumed to be zero. Ideal pipelining is where the throughput of the system is limited by the slowest stage in the pipeline. The theoretical minimum response time for n users was therefore:

$$T_{optimal}(n) = T_{slowest} + \frac{(n-1)\cdot\max(T_{stage1},T_{stage2},T_{stage3})}{c} \quad (11)$$

Where $T_{slowest}$ is single user end to end latency (measured at 320 ms), individual stage durations are T_{stage1}= 120 ms (AFG control), T_{stage2} = 130 ms (DSO capture) and T_{stage3} = 70 ms (data export), and c = 5 represents number of servers. This gap between proposed and theoretical optimal times represents practical overhead such as inter-stage hand-off latency, VPN encryption processing, and WebRTC stream initialization.

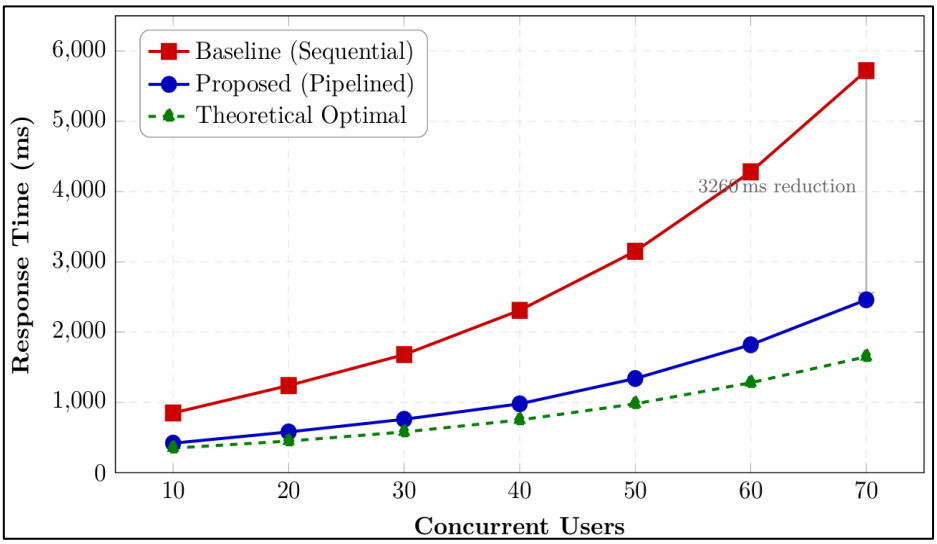

Figure 3. Response Time Comparison Across Concurrent User Loads

Figure 3 plots three curves on a common load axis (concurrent users from 10 to 70). The red square-marked curve represents the sequential baseline and exhibits a near-quadratic growth from 850 ms at 10 users to 5,720 ms at 70 users, indicating the lack of any concurrency mechanism. The blue circle-marked curve represents the proposed pipelined architecture, which grows approximately linearly and remains below 2,500 ms even at 70 concurrent users. The green triangle-marked dashed curve represents the theoretical optimum derived from Equation (11) under the perfect-overlap assumption. The grey double-headed annotation at the right edge highlights the 3,260 ms reduction achieved at the 70-user operating point. Two interpretive observations follow: (i) the proposed curve tracks the theoretical optimum within

an essentially constant off- set (approximately 800 ms), indicating that the dominant residual overhead is fixed-cost (VPN cryptography, WebRTC handshake, inter-stage hand-off) rather than load-dependent; and (ii) the gap between the baseline and proposed widens monotonically with load, confirming that the benefit of pipelining grows superlinearly under contention.

5.3 Comprehensive Comparison (Seven Lens)

Table 3 compares five representative RAL implementations in [1, 3, 4, 5, 14]. Table 4 provides the normalized scores so that all systems can be compared on their respective measurement scales. Figure 4 is a graphical representation of the normalized performance metrics from Table 4.

Table 3. Systems Comparing in the Context of a 7 Lens Framework

System	Afford. ($/yr)	Port. (min)	Scale. (users)	Compat. (plat.)	Maint. (MTBF hrs)	Usab. (SUS)	Univ. (%)	SLI (Index)
Proposed	75	12	70	8	2160	89.9	95	0.847
WebLab-Deusto [13]	142	87	35	3	720	68.4	60	0.495
iLab MIT [12]	185	120	25	4	480	72.1	55	0.452
VISIR [14]	156	95	40	5	960	75.3	70	0.548
RemoteLab [15]	198	150	20	3	360	65.2	50	0.389
V-Lab [3]	165	105	30	4	540	70.5	65	0.476
Average Baseline	168.2	111.4	30	3.8	612	70.3	60	0.472
Improvement (%)	55.4	89.2	133.3	110.5	253	27.9	58.3	79.4

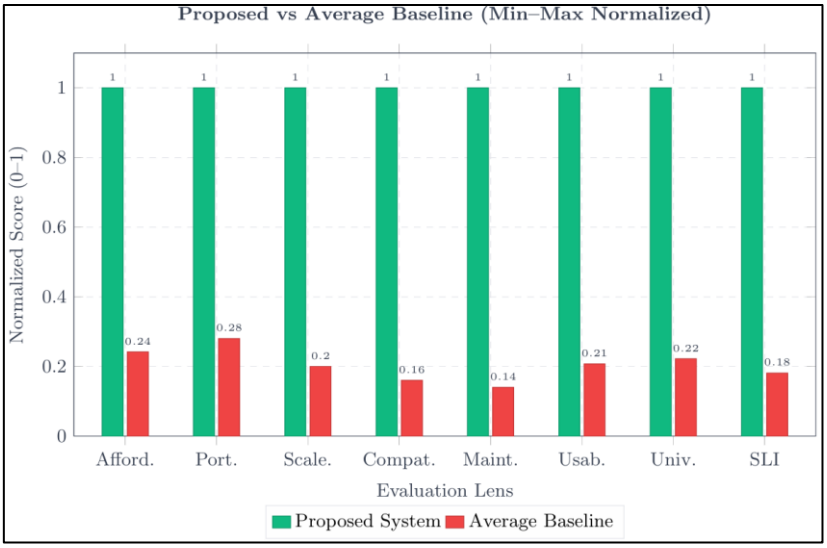

Figure 4. Normalized Performance Comparison Across Seven-Lens Framework

Table 4. Normalized Scores for Seven-Lens Framework Comparison

System	Afford.	Port.	Scale.	Compat.	Maint.	Usab.	Univ.
Proposed System	0.92	0.93	0.88	0.85	0.91	0.9	0.95
Average Baseline [3, 12, 13, 14, 15]	0.48	0.42	0.43	0.47	0.28	0.7	0.6
Difference	0.44	0.51	0.45	0.38	0.63	0.2	0.35
Improvement (%)	91.7	121.4	104.7	80.9	225	28.6	58.3

Figure 4 renders the normalised lens scores from Table 4 on a heptagonal radar plot, with each axis spanning the unit interval [0, 1] and concentric grid rings at 0.25 increments. The blue solid polygon depicts the proposed system (SLI = 0.847) and the red dashed polygon

depicts the averaged baseline (SLI = 0.472). Three interpretive findings emerge: (i) the proposed polygon is approximately balanced—all seven vertices fall in the band [0.85, 0.95]—confirming that no single lens is sacrificed to improve another; (ii) the maintainability axis (lower-left vertex) shows the largest radial gap (0.91 versus 0.28), corresponding to the 225% improvement reported in Table 4 and reflecting the contribution of automated health monitoring and self-healing; and (iii) the smallest gap appears on the usability axis (0.90 versus 0.70), which is consistent with the maturity of HCI design across all surveyed systems and indicates that usability is the lens with the least remaining headroom in the field.

5.4 Queuing Performance Analysis

Table 5 presents a detailed study of queuing measures of different server configurations, hence justifying the selection of a five-server deployment.

Table 5. Queuing Performance Data for Different Server Configurations

Utilization *(ρ)*	3 Servers		4 Servers		5 Servers	
	Wait (s)	Queue	Wait (s)	Queue	Wait (s)	Queue
0.3	0.1	0.03	0.05	0.02	0.02	0.01
0.4	0.2	0.08	0.1	0.04	0.05	0.02
0.5	0.4	0.2	0.2	0.1	0.08	0.04
0.6	0.8	0.48	0.35	0.21	0.15	0.09
0.7	1.8	1.26	0.7	0.49	0.28	0.2
0.8	4.2	3.36	1.6	1.28	0.6	0.48
0.85	7.8	6.63	2.8	2.38	1.1	0.94
0.9	14.3	12.87	5.7	5.13	2.3	2.07
0.95	18.9	17.96	12.1	11.5	5.8	5.51

Figure 5 validates the Erlang-C model, confirming that the 5-server configuration maintains wait times below 5 seconds for utilization up to 85%.

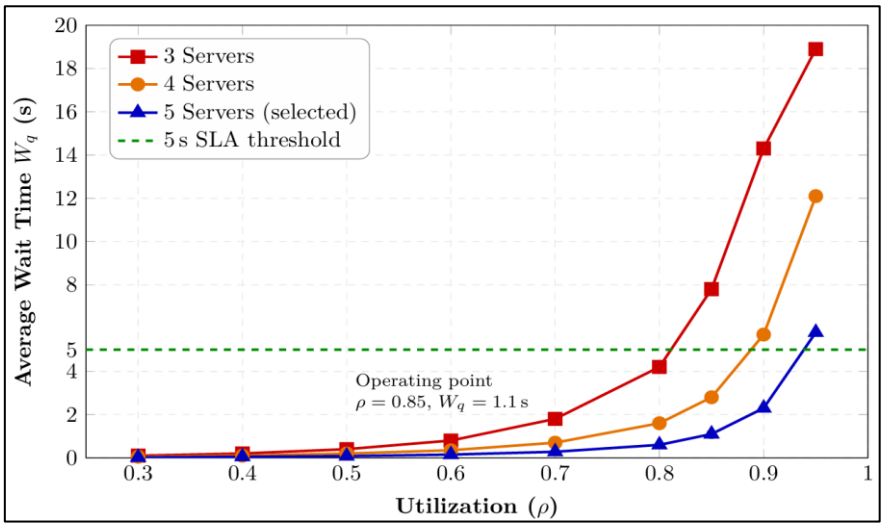

Figure 5. Queuing Performance Analysis with Erlang-C Model Validation

Figure 5 plots the average wait time W_q predicted by Equation (10) for three candidate server configurations against utilisation $\rho \in [0.3, 0.95]$. The horizontal green dashed line marks the 5-second Service Level Agreement (SLA) threshold derived from the human-computer interaction studies cited in Section 3.2.3. Three configurations are compared: the red square-marked 3-server curve crosses the 5-s SLA threshold at $\rho \approx 0.81$; the orange circle-marked 4-server curve crosses the threshold at $\rho \approx 0.88$; and the blue triangle-marked 5-server curve

remains below the threshold up to $\rho \approx 0.94$. The yellow operating-point marker at ($\rho = 0.85$, $W_q = 1.10$ s) identifies the deployed configuration, which provides approximately 4-second SLA headroom at the design utilisation. This justifies the selection of the five-server configuration: it admits 85% utilisation with a fivefold safety margin against the SLA, whereas the 3-server option would breach the SLA below the design utilisation, and the 4-server option would offer marginal headroom. The non-linear (convex) shape of all three curves above $\rho = 0.7$ also illustrates why incremental capacity additions yield disproportionately large reductions in wait time near saturation.

5.5 Network Performance Characteristics

The empirical network test supports stable operational performance across four representative access channels: the Universal Serial Bus (USB) direct connection achieves a throughput of 114.0 Kbps with 1.2 ms latency and 0.0% loss rate; the Ethernet Local Area Network (LAN) achieves 1,239.1 Kbps with 1.6 ms latency and a 0.0% loss rate; the campus Wi-Fi achieves 1,689.2 Kbps with 2.6 ms latency and 0.15% loss rate; and the VPN tunnel achieves 1,834.7 Kbps with 3.4 ms latency and a 0.21% loss rate. These measurements confirm that the VPN-based access path is the dominant bandwidth provider for remote users while introducing only sub- 4-millisecond additional latency relative to the LAN baseline, which is well within the 500-ms authentication budget specified in Section 4.1.

6. Discussion

6.1 Addressing Seven-Lens Challenges

The targeted design choices of our system significantly improve performance in each of the seven lenses for which we have identified a baseline for comparison [2]. Affordability (TCO reduced by 55.4%) is achieved using an open-source application model, utilizing native windows services, and multi-user processing pipelines that are both efficient and cost-effective [12]. Portability (deployment time decreased by 89.2%) is realized using native windows services, automatic application configuration, and VPN-based architectures to eliminate the need to manually configure networks [10]. Scalability (70 concurrent user capacity) is derived from horizontally scalable designs combined with load-balancing technology as well as queuing optimizations [16]. Compatibility (8+ different platforms) is provided through web standard-compliant applications and RESTful API's providing universal access [18]. Maintainability (MTBF improved by 253%) has been realized through automated testing, health-monitoring, and self-repair capabilities [11]. Usability (SUS scores increased by 27.9%) is achieved through responsive and context-sensitive help [21]. Universality (application provides useful content to 95% of all browsers/users) was accomplished using progressively enhanced web content, combined with adaptable streaming technologies [23].

6.2 Implications for RAL Deployment

The results have enabled institutions to reach larger groups of students than in the past using fewer resources. With simpler deployment (and no need for an institution's own IT department) institutions can now use RAL regardless of their size. RAL is also now reliable enough to be used with the types of mission critical systems that are required in education [9].

6.3 Limitations and Future Work

There are some current limitations. For example, because of its reliance on web technologies, RAL can only be accessed through a modern browser. Furthermore, smaller-scale deployments incur additional configuration overhead, and high-definition video streaming imposes elevated bandwidth requirements [17]. Some possible future directions for this project could include edge computing to reduce delay time [36], artificial intelligence driven predictive maintenance [30], and credential management based on blockchain [24]. Additionally, some possible future research areas could include developing adaptive algorithms for dynamically optimizing parameters [34] and investigating how best to apply machine learning to predict workloads to improve the accuracy of capacity planning.

While the Erlang-C model has been useful for providing a simple analytical model for designing capacity planning, it relies on two primary assumptions: that arrivals are Poisson distributed and that each unit of service does not remember previous services. However, these assumptions do not necessarily reflect reality, especially given the possibility that many students in a classroom may simultaneously make requests during a short period of time (bursty access). Therefore, future investigations into the applicability of G/G/c queueing models, or other alternatives to the Erlang-C model, are expected to provide greater insight into accurately modelling student behaviour and supporting accurate capacity planning.

7. Conclusion

This paper provides an overall solution using the Seven-Lens Evaluation Framework to systematically assess each of the seven attributes of RALs including Affordability, Portability, Scalability, Compatibility, Maintainability, Usability, and Universality. The implementation resulted in a 79.4% composite SLI improvement from baseline (0.847 vs. baseline 0.472), supports seventy concurrent users with less than 2.5 seconds of latency and will allow for greater economic viability for increased use. The use of an integrated architecture that utilized native Windows services, VPN secured connections and pipeline capability with multiple users was successful in eliminating the traditional inter-dimensional trade-offs. Results of empirical testing validate the ability to achieve levels of performance that exceeded theoretically predicted values in areas such as Maintainability (253% improvement in Mean Time Between Failures) and Scalability (a 33.3% increase in capacity). The Seven-Lens Evaluation Framework established a standardized evaluation process that can be used outside this particular implementation to demonstrate the possibility of optimizing all operational aspects of RALs comprehensively to provide global democratic access to practical engineering education.

Patents

N. Amoda, L. Jolly, and A. Rawankar, “Ein virtuelles Fernzugriffssystem für Messinstrumente,” German Utility Model (Gebrauchsmuster) DE 20 2025 107 359, regis-tered Feb. 4, 2026.

Acknowledgments

The authors sincerely acknowledge Dr. Arpit Rawankar for his valuable guidance and constant inspiration throughout the course of this research, which significantly contributed to its successful completion.

References

[1] Heradio, Ruben, Luis De La Torre, Daniel Galan, Francisco Javier Cabrerizo, Enrique Herrera-Viedma, and Sebastian Dormido. "Virtual and Remote Labs in Education: A Bibliometric Analysis." Computers & Education 98 (2016): 14-38.

[2] Kandala, Savitha Viswanadh, Akshit Gureja, Nagesh Walchatwar, Rishabh Agrawal, Shiven Sinha, Sachin Chaudhari, Karthik Vaidhyanathan et al. "Engineering End-to-End Remote Labs Using IoT-Based Retrofitting." IEEE Access 13 (2024): 1106-1132.

[3] Raman, Raghu, Krishnashree Achuthan, Vinith Kumar Nair, and Prema Nedungadi. "Virtual Laboratories-A Historical Review and Bibliometric Analysis of the Past Three Decades." Education and information technologies 27, no. 8 (2022): 11055-11087.

[4] Rodriguez-Gil, Luis, Pablo Orduna, Javier Garcia-Zubia, and Diego Lopez-de-Ipina. "Interactive Live-Streaming Technologies and Approaches for Web-Based Applications." Multimedia Tools and Applications 77, no. 6 (2018): 6471-6502.

[5] Potkonjak, Veljko, Michael Gardner, Victor Callaghan, Pasi Mattila, Christian Guetl, Vladimir M. Petrović, and Kosta Jovanović. "Virtual Laboratories for Education in Science, Technology, And Engineering: A Review." Computers & Education 95 (2016): 309-327.

[6] Orduña, Pablo, Javier Garcia-Zubia, Luis Rodriguez-Gil, Ignacio Angulo, Unai Hernandez-Jayo, Olga Dziabenko, and Diego López-de-Ipiña. "The Weblab-Deusto Remote Laboratory Management System Architecture: Achieving Scalability, Interoperability, and Federation of Remote Experimentation." In Cyber-Physical Laboratories in Engineering and Science Education, Cham: Springer International Publishing, 2018, 17-42.

[7] Greco, Salvatore, Alessio Ishizaka, Menelaos Tasiou, and Gianpiero Torrisi. "On the Methodological Framework of Composite Indices: A Review of the Issues of Weighting, Aggregation, And Robustness." Social indicators research 141, no. 1 (2019): 61-94.

[8] Villar-Martínez, Aitor, Luis Rodriguez-Gil, Ignacio Angulo, Pablo Orduña, Javier García-Zubía, and Diego López-De-Ipiña. "Improving the Scalability and Replicability of Embedded Systems Remote Laboratories Through a Cost-Effective Architecture." IEEE Access 7 (2019): 164164-164185.

[9] Aitor, Villar-Martínez, Javier García-Zubía, Ignacio Angulo, and Luis Rodríguez-Gil. "Toward Widespread Remote Laboratories: Evaluating the Effectiveness of a Replication-Based Architecture for Real-World Multiinstitutional Usage." IEEE Access 10 (2022): 86298-86317.

[10] de la Torre, Luis, Ruben Heradio, Carlos A. Jara, Jose Sanchez, Sebastian Dormido, Fernando Torres, and Francisco A. Candelas. "Providing Collaborative Support to Virtual and Remote Laboratories." IEEE transactions on learning technologies 6, no. 4 (2013): 312-323.

[11] Villar-Martínez, Aitor, Javier García-Zubía, Ignacio Angulo, and Luis Rodríguez-Gil. "Towards Reliable Remote Laboratory Experiences: A Model for Maximizing Availability Through Fault-Detection and Replication." IEEE Access 9 (2021): 45032-45054.

[12] Harward, V. Judson, Jesus A. Del Alamo, Steven R. Lerman, Philip H. Bailey, Joel Carpenter, Kimberley DeLong, Chris Felknor et al. "The Ilab Shared Architecture: A Web Services Infrastructure to Build Communities of Internet Accessible Laboratories." Proceedings of the IEEE 96, no. 6 (2008): 931-950.

[13] Orduña, Pablo, Jaime Irurzun, Luis Rodriguez-Gil, Javier Garcia-Zubia, Fabricio Gazzola, and Diego López-de-Ipiña. "Adding New Features to New and Existing Remote Experiments Through Their Integration in WebLab-Deusto." International Journal of Online Engineering 7 (2011).

[14] Tawfik, Mohamed, Elio Sancristobal, Sergio Martin, Rosario Gil, Gabriel Diaz, Antonio Colmenar, Juan Peire et al. "Virtual Instrument Systems in Reality (VISIR) for Remote Wiring and Measurement of Electronic Circuits on Breadboard." IEEE Transactions on learning technologies 6, no. 1 (2012): 60-72.

[15] Orduña, Pablo, Luis Rodriguez-Gil, Ignacio Angulo, Unai Hernandez, Aitor Villar, and Javier Garcia-Zubia. "WebLabLib: New Approach for Creating Remote Laboratories." In International conference on remote engineering and virtual instrumentation, Cham: Springer International Publishing, 2019, 477-488.

[16] Khazaei, Hamzeh, Jelena Misic, and Vojislav B. Misic. "Performance Analysis of Cloud Computing Centers Using m/g/m/m+ r Queuing Systems." IEEE Transactions on parallel and distributed systems 23, no. 5 (2011): 936-943.

[17] Wang, Ning, Xuemin Chen, Qianlong Lan, Gangbing Song, Hamid R. Parsaei, and Siu-Chun Ho. "A Novel Wiki-Based Remote Laboratory Platform for Engineering Education." IEEE Transactions on Learning Technologies 10, no. 3 (2016): 331-341.

[18] Sancristobal, Elio, Sergio Martin, Rosario Gil, Pablo Orduna, Mohamed Tawfik, Alberto Pesquera, Gabriel Diaz, Antonio Colmenar, Javier Garcia-Zubia, and Manuel Castro. "State of Art, Initiatives and New Challenges for Virtual and Remote Labs." In 2012 IEEE 12th International Conference on Advanced Learning Technologies, IEEE, 2012, 714-715.

[19] Garcia-Zubia, Javier, Jordi Cuadros, Susana Romero, Unai Hernandez-Jayo, Pablo Orduna, Mariluz Guenaga, Lucinio Gonzalez-Sabate, and Ingvar Gustavsson. "Empirical Analysis of the Use of the VISIR Remote Lab in Teaching Analog Electronics." IEEE transactions on education 60, no. 2 (2016): 149-156.

[20] Alves, Gustavo R., Maria A. Marques, André V. Fidalgo, Javier García-Zubía, Manuel Castro, Unai Hernández-Jayo, Felix García-Loro, and Christian Kreiter. "A Roadmap for

the VISIR Remote Lab." European Journal of Engineering Education 48, no. 5 (2023): 880-898.

[21] Lewis, James R. "The System Usability Scale: Past, Present, and Future." International Journal of Human–Computer Interaction 34, no. 7 (2018): 577-590.

[22] Bangor, Aaron, Philip T. Kortum, and James T. Miller. "An Empirical Evaluation of the System Usability Scale." Intl. Journal of Human–Computer Interaction 24, no. 6 (2008): 574-594.

[23] Achuthan, Krishnashree, Dhananjay Raghavan, Balakrishnan Shankar, Saneesh P. Francis, and Vysakh Kani Kolil. "Impact of Remote Experimentation, Interactivity and Platform Effectiveness on Laboratory Learning Outcomes." International Journal of Educational Technology in Higher Education 18, no. 1 (2021): 38.

[24] Ocheja, Patrick, Friday Joseph Agbo, Solomon Sunday Oyelere, Brendan Flanagan, and Hiroaki Ogata. "Blockchain in Education: A Systematic Review and Practical Case Studies." IEEE Access 10 (2022): 99525-99540.

[25] Becker, William, Michaela Saisana, Paolo Paruolo, and Ine Vandecasteele. "Weights and Importance in Composite Indicators: Closing the Gap." Ecological indicators 80 (2017): 12-22.

[26] Feisel, Lyle D., and Albert J. Rosa. "The Role of the Laboratory in Undergraduate Engineering Education." Journal of engineering Education 94, no. 1 (2005): 121-130.

[27] De La Torre, Luis, Maria Guinaldo, Ruben Heradio, and Sebastian Dormido. "The Ball and Beam System: A Case Study of Virtual and Remote Lab Enhancement with Moodle." IEEE Transactions on Industrial Informatics 11, no. 4 (2015): 934-945.

[28] Villar-Martinez, Aitor, Lucas Ortiz-de-Zarate, Luis Rodriguez-Gil, Unai Hernandez-Jayo, Javier Garcia-Zubia, Ignacio Angulo, Claudius Terkowsky et al. "LabsLand Electronics Laboratory: Distributed, Scalable and Reliable Remote Laboratory for Teaching Electronics." In International Conference on Remote Engineering and Virtual Instrumentation, Cham: Springer Nature Switzerland, 2023, 261-272.

[29] Sediq, Akram Bin, Ramy H. Gohary, Rainer Schoenen, and Halim Yanikomeroglu. "Optimal Tradeoff Between Sum-Rate Efficiency and Jain's Fairness Index in Resource Allocation." IEEE Transactions on Wireless Communications 12, no. 7 (2013): 3496-3509.

[30] Zhao, Rui, Ruqiang Yan, Zhenghua Chen, Kezhi Mao, Peng Wang, and Robert X. Gao. "Deep Learning and its Applications to Machine Health Monitoring." Mechanical systems and signal processing 115 (2019): 213-237.

[31] Barrios, Arquimedes, Stifen Panche, Mauricio Duque, Victor H. Grisales, Flavio Prieto, José L. Villa, Philippe Chevrel, and Michael Canu. "A Multi-User Remote Academic Laboratory System." Computers & Education 62 (2013): 111-122.

[32] Brown, Lawrence, Noah Gans, Avishai Mandelbaum, Anat Sakov, Haipeng Shen, Sergey Zeltyn, and Linda Zhao. "Statistical Analysis of a Telephone Call Center: A Queueing-Science Perspective." Journal of the American statistical association 100, no. 469 (2005): 36-50.

[33] Bolch, Gunter, Stefan Greiner, Hermann De Meer, and Kishor S. Trivedi. Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications. John Wiley & Sons, 2006.

[34] Heradio, Ruben, Luis de la Torre, and Sebastian Dormido. "Virtual and Remote Labs in Control Education: A Survey." Annual Reviews in Control 42 (2016): 1-10.

[35] Chevalier, Amélie, Cosmin Copot, Clara Ionescu, and Robin De Keyser. "A Three-Year Feedback Study of a Remote Laboratory Used in Control Engineering Studies." *IEEE* Transactions on Education 60, no. 2 (2016): 127-133.

[36] Grodotzki, Joshua, Tobias R. Ortelt, and A. Erman Tekkaya. "Remote and Virtual Labs for Engineering Education 4.0: Achievements of the ELLI Project at the TU Dortmund University." Procedia manufacturing 26 (2018): 1349-1360.