

SERVICE MANAGEMENT FOR SMART BUILDING ADMINISTRATION USING VIRTUAL IOT SIMULATION AND PREDICTIVE ANALYTICS

David Kubart¹, Zuzana Špendel¹

¹Department of Economics, Faculty of Business and Economics, Faculty of Business and Economics, Mendel University in Brno, Zemědělská 1, 613 00 Brno, Czech Republic

DOI: https://doi.org/10.11118/ecos2026_018

ABSTRACT

This article addresses the design and implementation of a virtual service management model for smart building management within an enterprise environment. The work innovatively links the ITIL 4 methodology with a containerized IoT infrastructure, edge-cloud architecture, and predictive analytics. The objective of the research is to demonstrate that even a fully software-defined smart building environment can be systematically managed as a complex service with a clearly defined Service Level Agreement (SLA) and processes for effective incident and event management. To validate the proposed approach, a fully portable laboratory environment was developed using Docker containers, MQTT asynchronous communication, an InfluxDB telemetry data store, Grafana visualization, and Python modules for data generation and advanced analytics. Building behavior is dynamically configured via configuration files, allowing a single codebase to simulate various types of facilities with different operational priorities and SLA requirements. The proposed model was thoroughly verified through experimental scenarios, including simulations of network communication outages, MQTT message delays, sensor data errors, and complete physical disasters. The results demonstrate the practical usability and benefits of ITIL 4 principles (ITIL Foundation 2019) for digitalized building management, increasing data reliability, and automating operations.

Keywords: ITIL 4, Smart Building, Internet of Things (IoT), IT Service Management (ITSM), Predictive Analytics, MQTT, Service Level Agreement (SLA)

JEL Code: C88



This article is licensed under CC BY-SA 4.0 Attribution-ShareAlike 4.0 International license. <https://creativecommons.org/licenses/by-sa/4.0/deed.en>

1 INTRODUCTION

In recent years, the digitalization of enterprise buildings has significantly transformed how organizations ensure operational reliability, security, energy efficiency, and user comfort. The development of the Internet of Things (IoT) and continuous telemetry collection allows building status monitoring at a much higher granularity than before, using the acquired data not only for operational oversight but also for qualified operations management (Cirani 2019). Within an enterprise environment, however, the mere availability of raw data is often insufficient. What matters critically is whether this data is correctly interpreted in relation to the quality of the service provided, operational risks, and the strategic goals of the organization. In practice, smart building management is often understood merely as a technical task encompassing data collection, status visualization, and basic automation interventions. While functional at the subsystem level, such a concept by itself is inadequate for comprehensive and systematic service management. There is a clear need to view smart building management as an integrated corporate service with established responsibilities, quality parameters, response rules for anomalies, and continuous improvement mechanisms. The ITIL 4 methodology represents a dominant and exceptionally suitable framework for this approach, as it allows purely technical events and operational states to be translated into the structured language of service management.

The primary goal of this work is to design and experimentally validate a service management model for smart building management using the principles of ITIL 4 to their full extent. The core of the research does not lie in the physical technology itself, but in the design and validation of the process and service approach. The technological layer acts here as a means through which the proposed model can be realized in a laboratory, simulated, and critically assessed for its real benefits and limits.

2 CURRENT STATE

Traditional Building Management Systems (BMS) are frequently proprietary, isolated, and lack the flexibility to integrate with modern cloud platforms and advanced analytics. While the rise of the Internet of Things opens possibilities for distributed management and detailed monitoring, it creates a fundamental problem with interpreting the enormous amount of data generated. A smart building is a typical example of a cyber-physical system where technical monitoring must be closely tied to operational interpretation. A single measured value (e.g., an elevated temperature) gains true meaning only in relation to the wider context, the purpose of the building, SLA limits, and the actual impact on end-users.

2.1 The ITIL 4 Methodological Framework in the Context of IoT

The ITIL 4 framework defines service management as a set of specialized organizational capabilities for generating value for customers in the form of services. For the smart building domain, the following practices form the core of our proposal:

- **Monitoring and Event Management:** Continuous monitoring of infrastructure states, filtering sensor noise, and capturing significant events that may indicate an emerging problem.
- **Incident Management:** Rapid detection, classification, logging, and effective resolution of operational deviations to restore standard service operations as quickly as possible (MTTR) and minimize negative impact (Adhikari *et al* 2024).

- Service Level Management (SLM): Defining clear, measurable parameters of service quality (monitoring availability, environmental stability) and relentlessly evaluating them against contractual obligations.
- Problem Management: Identifying and analyzing the deeper root causes of recurring incidents and finding preventive solutions to prevent their further occurrence.

2.2 The Role of Predictive Analytics

Time series are the natural data model of the Internet of Things. Predictive analytics (IBM 2026) applied to this data allows transforming reactive management into proactive and preventive management. Early prediction of physical variables (e.g., predicting that a critical temperature threshold in a server room will be breached within the next few tens of minutes) gives the system or operator time to react before an SLA breach and an incident actually occur. At the same time, however, it is necessary to view the predictive layer as a decision support tool rather than a 100% replacement for operational judgment, due to its inherent dependence on the quality of input data and the potential for false alarms (Géron 2022).

3 APPLICATION AND SYSTEM ARCHITECTURE DESIGN

The proposed system is conceived as a distributed edge-cloud architecture that reflects modern trends in IoT and edge computing. The underlying idea is to divide responsibility between a local autonomous unit (Edge layer) located directly in the facility and a central coordination and analytics platform (Cloud layer).

3.1 Distributed Architecture and Communication Model

The system consists of three main structural layers:

- Building Edge Layer: Runs directly within the simulated facility. It includes virtual IoT sensors generating telemetry (temperature, humidity, device status), a local MQTT communication bus, and a key component—the Edge Controller. This controller possesses local decision-making logic, a local Telegraf/InfluxDB database, and is capable of fully autonomous building management (e.g., HVAC control) even in the event of a total loss of cloud connectivity.
- VPN Layer: Provides a secure and isolated connection between the distributed edge locations and the central cloud via a virtual private network, closely simulating a real enterprise deployment.
- Cloud Layer: Represents the central brain of the system. It receives aggregated data, performs advanced data quality validation, runs predictive models, maintains a global service incident log, evaluates SLAs, and provides a web-based dispatching interface for operators. The communication model is strictly built on the MQTT protocol (Publish/Subscribe). The designed topic model is hierarchical (format: building/floor/room/sensor_type) and strictly distinguishes between local operations (frequent telemetry with a low QoS 0 level) and cloud communication (critical status messages, heartbeats, incidents, and reverse control commands with QoS 1 and the Retain flag).

3.2 Data and Simulation Model

The simulation model implements a real physical runtime loop that models thermal inertia, ambient environment influences, and the behavior of ventilation and heating. A key feature of the design is the ability to inject faults. On demand, the system allows simulating both software and hardware anomalies—ranging from generating invalid data and artificially delaying messages to simulating complete infrastructure crashes.

4 APPLICATION DESCRIPTION AND IMPLEMENTATION

The application was completely implemented as a fully containerized open-source solution. The entire system is divided into separate projects managed via Docker Compose, guaranteeing 100% portability, service isolation, and easy replicability of the laboratory environment (Docker 2026).

4.1 Edge Layer Implementation and Double MQTT Architecture

The local building utilizes a lightweight Docker image based on python:3.9-slim to run the simulation scripts and the Edge Controller. The Edge Controller is the heart of local intelligence and implements a critical double MQTT architecture, utilizing two parallel MQTT clients:

- `local_client`: Communicates exclusively with the local broker (Eclipse Mosquitto) inside the building's isolated network. It processes granular telemetry from all sensors at one-second intervals.
- `cloud_client`: Maintains an encrypted connection through the VPN to the central cloud. It publishes regular heartbeat messages, opens/closes incidents in the cloud registry, and sends only smartly aggregated data, radically conserving WAN network bandwidth.

The client includes a Last Will and Testament mechanism, ensuring that if an unexpected outage or connection breakdown occurs, the cloud broker immediately detects the OFFLINE status and the cloud core automatically initiates a highest-priority (P1) incident—loss of facility monitoring.

4.2 Cloud Core Engine and Data Quality Control Pipeline

The central cloud core (ITIL Core Engine) is implemented in Python and functions as the main service layer. A new and essential element of the implementation is a dedicated protective layer of the cloud data pipeline (`validator.py`). This component intercepts incoming MQTT messages from buildings before they are stored or forwarded to analytics modules, subjecting them to strict validation:

- **Syntactic Check**: Verifying correct JSON structure, the presence of mandatory metadata (building identifier, timestamp), and correct data types.
- **Semantic Check**: Detecting time drift between the generation time on the edge device and the arrival time in the cloud, uncovering hidden synchronization or network delay issues.
- **Physical Validation**: Checking whether measured values fall within physically possible boundaries (e.g., a room temperature showing a sudden jump of 50 °C within a single second), which indicates a failure of the hardware sensor itself.

If a message fails validation, it is immediately diverted to the error history and is not passed further into automation schemes or linear regression predictive models. A cloud State Tracker monitors the frequency of these errors within a sliding time window. If the number of faulty messages exceeds defined thresholds (DQ_P3_THRESHOLD and DQ_P2_THRESHOLD), the system automatically escalates a data quality incident because the data loses trustworthiness and cannot be safely used to manage the building.

4.3 Observability and Data Model

An InfluxDB time-series database was deployed to store telemetry data and operational logs at both local and cloud levels. The translation of asynchronous messages is handled by Telegraf middleware, which parses topics and transforms them into tags and metrics. The visualization layer is implemented in Grafana, where two types of dashboards are created: detailed technical monitoring for engineers at the local level, and managerial-dispatch reporting for tracking SLA fulfillment, service availability, and incident overviews at the cloud level.

5 EXPERIMENTS AND RESULTS EVALUATION

To experimentally verify the functionality of the entire proposed concept and its real link to ITIL 4 practices, an extensive set of comprehensive testing scenarios was conducted. The behavior of the entire ecosystem was rigorously monitored and logged using a unified logging model managed via .env configuration.

5.1 Test: Reference Autonomous Operation of the Building

The aim of the first experiment was to verify the Edge layer's basic capability to function autonomously and correctly synchronize its state after connectivity was restored. The building started in local mode without an immediate connection to the cloud. During this operation, the Edge Controller successfully detected local anomalies and stored them in a local incident ledger (ledger.json).

Following a simulated establishment of the VPN and MQTT connection with the cloud, immediate and flawless synchronization took place. A one-time transfer of 15 pending incident records was delivered to the cloud layer. The central dispatch then issued reverse control commands for the ventilation system (HVAC), leading to environmental stabilization and the successful, automatic closure of all incidents. The incident closure success rate reached 100%, precisely demonstrating the high reliability of local autonomy and distributed storage.

5.2 Test: MQTT Data Quality and Escalation of Data Quality Incidents

This key experiment focused on verifying the protective function of the validator.py component in the cloud data pipeline. Faulty and delayed sensor data were intentionally injected into the system to test the cloud's ability to detect a decline in data stream trustworthiness.

The test results confirmed the 100% success of the design. The system intercepted invalid messages, diverted them, and prevented their hazardous influence on automated building control. When the error limit within the sliding window was exceeded, the cloud immediately and correctly

generated a dedicated Data Quality incident (initially with P3 priority, subsequently escalated to P2). This experimentally demonstrated that the system can effectively protect management integrity and proactively alert the operator that telemetry data from the given facility is unreliable.

5.3 Test: Complex Physical Disaster in the Facility

To demonstrate the robustness of the solution in critical situations, a scenario simulating a discrete physical disaster—specifically, a massive water leak in a data center facility (DC_Brno_1)—was executed. This type of event poses an immediate and fatal risk to service availability, hardware safety, and overall business continuity.

The system's reaction took place across three fully coordinated layers:

- Edge Layer: The sensor immediately detected critical humidity, the Edge Controller received the error flag, instantly activated emergency mode, and rapidly generated a local incident with the highest priority, P1.
- Cloud Layer: The disaster message was transferred to the cloud as a priority using QoS 1. The central ITIL Core Engine immediately registered the building as an affected Configuration Item (CI), logged the P1 incident in the global registry, triggered a visual alarm on the Grafana dispatch dashboard, and started the countdown for evaluating strict SLA limits.
- Network Layer: VPN and network logs (vpn_audit.log) confirmed that stable and prioritized communication between the edge location and the cloud SLA engine occurred throughout the incident without any packet loss.

The incident was held open and active in the system until the disaster condition was physically resolved in the simulator and a confirming clear_fault command was sent. Upon receiving it, the cloud core correctly closed the incident, calculating the precise duration and final MTTR metric. This test successfully proved that the proposed model does not represent merely an isolated technical solution, but forms an integrated, auditable, and fully traceable incident process from the onset of a physical failure to its procedural closure.

6 EVALUATION AND DISCUSSION

Experimental data clearly confirms that linking the ITIL 4 methodology with the technical capabilities of the Internet of Things brings an entirely new quality to building management. The main benefits of the designed and implemented solution can be summarized into three key points:

- Operational Continuity and Autonomy: Thanks to the implementation of the Edge Controller with a local database and decision logic, the building is capable of operating safely and regulating its internal environment even when completely disconnected from the cloud. No data is lost, as incidents are stored locally and automatically synchronized immediately upon network restoration.
- High Data Trustworthiness: Deploying the validation component `validator.py` resolves a critical vulnerability in many IoT systems, which is blind trust in incoming data. The capability to detect syntactic, semantic, and physical anomalies and translate them into data quality incidents radically increases the security and stability of automated management.

- **Managerial Oversight and SLA Measurability:** Translating raw telemetry into standardized incidents with a clearly defined priority (P1 to P3) allows operations managers to allocate resources efficiently, track the actual availability of buildings as corporate services, and exactly evaluate compliance with contractual obligations to both suppliers and customers.

Regarding limitations and constraints, it must be acknowledged that the implemented solution was operated in a laboratory setup. In a real production deployment, it would be essential to eliminate certain intentional simplifications—most notably, implementing strict security for MQTT brokers (disabling anonymous access, introducing device authentication via certificates, TLS/MQTTs encryption, and detailed ACL rules for topic access). However, these steps were intentionally omitted from our laboratory version to preserve maximum code clarity and log readability, which in no way diminishes the validity of the scientific conclusions reached.

7 CONCLUSIONS

The goal of this research was to design, implement, and experimentally validate an innovative way in which the principles and practices of the international ITIL 4 framework can be effectively utilized to manage modern IoT smart building infrastructure supported by predictive analytics. The results of our work clearly show that the ITIL 4 methodology can be applied meaningfully and successfully far beyond its traditional domain of managing purely software and IT information systems. In the context of the Internet of Things and smart buildings, ITIL 4 functions as a robust translation and process framework that helps transform disparate technical data from hundreds of sensors and operational events into a coherently managed, transparent, and measurable corporate service. This approach creates a highly stable environment where it is possible not only to reliably monitor the current physical state of buildings but also to proactively evaluate the real impacts of incidents, effectively support operator decision-making, continuously increase operational data trustworthiness, and establish a firm foundation for deploying advanced predictive and preventive management for modern corporate facilities.

Acknowledgements

This paper was supported by the project CZ.02.1.01/0.0/0.0/16_017/0002334 Research Infrastructure for Young Scientists, this is co-financed from Operational Programme Research, Development and Education.

REFERENCES

- ADHIKARI, Deepak, JIANG, Wei, ZHAN, Jinyu, RAWAT, Danda B., BHATTARAI, Asmita. (2024). Recent advances in anomaly detection in Internet of Things: Status, challenges, and perspectives. *Computer Science Review*. 54, 100665. ISSN 1574-0137. <https://doi.org/10.1016/j.cosrev.2024.100665>
- CIRANI, Simone, FERRARI, Gianluigi, PICONE, Marco, VELTRI, Luca. (2019). *Internet of things: architectures, protocols and standards*. Hoboken: Wiley. ISBN 978-1-119-35967-8
- DOCKER. (2026). The Docker platform. *Docker Docs*. [accessed 2026-04-10]. <https://docs.docker.com/get-started/docker-overview/>

- GÉRON, Aurélien. (2022). *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. 3rd ed. Sebastopol: O'Reilly. ISBN 978-1-098-12561-5.
- GRIFFOR, Edward. (2017). *Handbook of Cyber-Physical Systems*. Cham: Springer. ISBN 978-3-319-51827-5
- IBM. (2026). What is predictive analytics? *IBM Topics*. [accessed 2026-03-15]. <https://www.ibm.com/topics/predictive-analytics>.
- ITIL Foundation. (2019). *ITIL 4 Edition, 2019*. London: AXELOS / TSO. ISBN 978-0-113-31599-4

Contact information

Polina Komárková: e-mail: polia.tselisheva@mail.ru