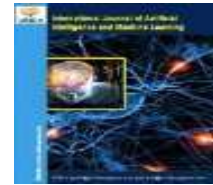




International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: <https://www.svedbergopen.com/>



Research Paper

Open Access

Edge-Deployed Partner Relationship Management for Distributed Telecom Field Operations

Brahmananda Naidu Dabara

Independent Researcher, India. Email: bramha.db@gmail.com

Abstract

Distributed telecom field operations, spanning rural towers, customer premises, underground infrastructure, and disaster-affected areas, contend with intermittent or absent connectivity that a centralised cloud-only Partner Relationship Management (PRM) platform cannot serve. Field engineers, partner technicians, and service contractors need to access customer records, equipment configurations, partner authorisations, work orders, inventory status, and technical documentation whether the network is available or not. This paper proposes an edge-deployed PRM architecture in which core field workflows execute locally on ruggedised edge nodes, the central PRM platform remains the authoritative system of record, and an event-driven synchronisation fabric bridges the two. The paper contributes: a five-criteria classification method for deciding which capabilities are offline-essential, a three-class data-ownership taxonomy (field-owned append-only, central-owned read-at-edge, leased single-writer) that governs how concurrent writes are resolved, a version-based synchronisation protocol with idempotent event processing, and a reconnection-storm handling design that combines controlled recovery windows, incremental event replication, and adaptive exponential backoff with jitter. A pilot deployment across 50 field sites, 320 edge devices, 180 partner organisations, and more than 850 field engineers processed over 120,000 work orders and 2.8 million synchronisation events. The pilot demonstrated a 21% increase in tasks completed per shift, a 32% reduction in incident response time, a 41% reduction in data-entry errors, a 99.3% synchronisation success rate, 94% of conflicts resolved automatically, and an increase in partner satisfaction from 4.0 to 4.6 on a 5-point scale.

Keywords: edge computing, partner relationship management, offline-first, telecom field operations, synchronisation, conflict resolution, reconnection storm, distributed systems.

I. Introduction

In distributed telecom field operations, relying on a centralised Partner Relationship Management (PRM) system creates significant operational challenges whenever connectivity is unreliable or unavailable. Field engineers, partner technicians, and service contractors often work in remote locations, at customer premises, in underground facilities, or in disaster-affected areas where network access is intermittent. Without continuous connectivity, they cannot access customer records, equipment configurations, partner authorisations, work orders, inventory status, or technical documentation in real time. The operational consequences include delays in troubleshooting, duplicate data entry, manual record-keeping, synchronisation conflicts, and slower issue resolution. Critical activities such as partner onboarding, service validation, compliance verification, and asset updates may be postponed until connectivity is restored, increasing operational costs and reducing productivity.

An edge-deployed PRM architecture addresses these challenges by processing essential partner data and workflows locally, enabling offline operation with secure synchronisation to the central platform once connectivity becomes available. The design improves response times, operational resilience, data consistency, and service continuity across geographically distributed telecom field operations. This paper contributes:

1. An edge PRM architecture in which the central platform remains the system of record while offline-essential capabilities execute locally on ruggedised edge nodes (§III).
2. A five-criteria classification method for deciding which capabilities are offline-essential and which may wait for connectivity (§IV).
3. A three-class data-ownership taxonomy governing concurrent-write behaviour: field-owned append-only, central-owned read-at-edge, and leased single-writer (§V).
4. A version-based synchronisation protocol with idempotent event processing, conflict detection at record and field granularity, and per-class conflict resolution rules (§VI).
5. A reconnection-storm handling design combining controlled recovery windows, incremental event replication, and adaptive exponential backoff with jitter (§VII).
6. Pilot outcomes across 50 field sites, 320 edge devices, 180 partner organisations, and more than 850 field engineers, processing 120,000+ work orders and 2.8 million synchronisation events (§VIII).

2. Related Work

Autonomic computing and self-adaptive systems. Kephart and Chess [1] introduced the MAPE-K reference model for self-managing systems. The edge PRM architecture aligns with MAPE-K by embedding local Monitor, Analyse, Plan, and Execute capabilities on each edge node while sharing Knowledge with the central platform through the synchronisation fabric.

Edge computing for enterprise workflows. Shi et al. [2] framed edge computing as computation at the network edge to reduce round-trip latency and preserve operational continuity during central-service degradation. Subsequent surveys [3, 4] characterise edge deployments across telecom operations, IoT, and manufacturing. Prior enterprise-workflow work at the edge has largely focused on latency-sensitive services rather than on the offline-first PRM workflows this paper addresses.

Offline-first application design. The offline-first pattern [5] treats persistent local storage as primary and network calls as opportunistic. Progressive Web Applications, CouchDB / PouchDB, and Realm have popularised the pattern in consumer applications. The pattern's transfer to enterprise PRM requires stronger conflict resolution and stricter compliance controls than consumer applications typically enforce.

Distributed data synchronisation. Vector clocks [6], version vectors, and CRDTs [7] are the foundational primitives for conflict-free replication under network partitions. The architecture uses version-based optimistic concurrency control with per-field granularity rather than a CRDT substrate, because most PRM data is not naturally commutative and the central platform must remain authoritative.

Conflict resolution strategies. Terry et al. [8] categorised conflict-resolution approaches for weakly consistent replicated stores into last-writer-wins, application-defined merges, and human review queues. The three-class ownership taxonomy of $\$V$ refines this categorisation by binding the resolution strategy to the data class rather than treating conflict resolution as a global policy.

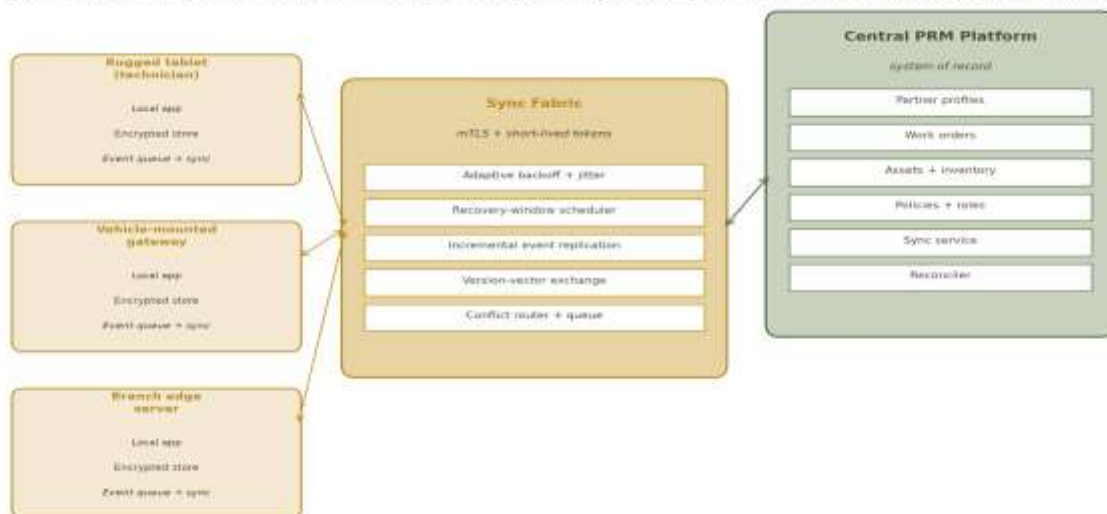
Reconnection-storm and thundering-herd behaviour. Nishtala et al. [9] documented cache-stampede and thundering-herd phenomena in large-scale distributed services and characterised the amplification effect of synchronous retries. The reconnection storm described in §VII is the mobile-fleet analogue of the same phenomenon, and the mitigation combines the classical exponential-backoff-with-jitter pattern [10] with controlled recovery windows and priority-tiered synchronisation.

Telecom field-operations platforms. Prior work on telecom operations support systems and field-service management has documented the operational cost of connectivity dependence [11, 12] but has not offered an architecture that unifies edge execution, three-class ownership, and reconnection-storm handling into a single design.

3. Architecture

The edge PRM architecture uses a distributed, offline-capable model in which core field workflows continue locally while the central PRM platform remains the system of record. Figure 1 shows the layered architecture.

Figure 1. Edge-deployed PRM architecture: local execution at the edge, authoritative record at the centre, sync fabric between



Edge node. At the field level, an edge node runs on a ruggedised gateway, branch server, vehicle-mounted device, or secured technician tablet. It hosts a lightweight local application, an encrypted operational data store, workflow rules, authentication tokens, synchronisation services, and selected AI-assisted functions. The node

supports work-order access, partner authorisation checks, equipment validation, site inspection, inventory updates, troubleshooting guidance, digital signatures, image capture, and compliance checklists. Frequently used reference data (product catalogues, network topology segments, service procedures, partner entitlements, customer-site details) is cached locally.

Central platform. The central PRM system remains the authoritative source for shared master data: partner status, customer entitlements, pricing, security policies, asset ownership, and work-order assignment. It synchronises assigned work orders, partner profiles, role permissions, asset records, site information, inventory availability, service-level commitments, technical documents, and configuration policies to the edge. It receives from the edge completed tasks, status changes, inspection results, equipment readings, photos, signatures, inventory consumption, timestamps, and audit events.

Synchronisation fabric. Between the edge and the central platform sits the synchronisation fabric, secured with mutual TLS and short-lived authentication tokens. It provides adaptive backoff and jitter for client retries, a recovery-window scheduler that meters mass-reconnection events (§VII), incremental event-based replication, version-vector exchange for conflict detection, and a conflict router that either resolves automatically or queues for supervisor review.

Selective replication. Only operationally necessary data is replicated to the edge. Each edge node stores task-specific information, nearby asset records, relevant partner profiles, technical documentation, and current inventory needed for active assignments. Obsolete cached data expires automatically after successful synchronisation or a predefined retention period. Large files (photos, technical drawings) are synchronised through resumable transfers separately from small transactional events; urgent transactions receive higher priority than reference-material transfers.

4. Deciding What Runs Offline

Not every PRM capability belongs at the edge. The design evaluates each candidate capability against five criteria: operational urgency, safety impact, dependence on real-time central data, data sensitivity, and the consequence of delay. A capability is treated as offline-essential when its absence could stop field work, create a safety or compliance risk, cause a repeat site visit, violate a service-level commitment, or prevent reliable evidence collection.

Figure 5 shows the classification matrix for representative PRM capabilities. Offline-essential capabilities include access to assigned work orders, customer-site details, equipment and asset information, partner authorisation status, installation procedures, troubleshooting guides, safety checklists, approved configuration templates, locally relevant inventory, task-status updates, barcode scanning, equipment readings, photographs, digital signatures, and completion evidence.

Figure 5. Offline-capability classification: 5-criteria scoring across candidate PRM capabilities

	Urgency	Safety	Real-time dependence	Data sensitivity	Delay consequence
Access assigned work orders	5	5	5	5	5
Customer-site details	5	5	5	5	5
Equipment / asset information	5	4	5	4	5
Partner authorisation status	5	5	3	5	5
Installation procedures + guides	5	3	3	5	5
Photos / signatures / evidence	5	5	4	5	5
Local inventory consumption	4	3	3	4	4
Global inventory optimisation	2	2	5	3	3
Commission / incentive processing	1	1	2	2	2
Pricing approvals	1	2	4	3	2
Executive dashboards	1	1	3	2	2
Partner onboarding	1	1	4	2	2

offline-essential (mean ≥ 4)
 central-only (mean < 4)

Freshness tolerance. Each data type carries a freshness tolerance. Technical manuals, assigned tasks, product specifications, and approved procedures remain valid for hours or days and are safely cached. Partner credentials, access permissions, and configuration policies carry expiration times and revocation rules; high-risk actions are disabled when the cached authorisation has expired.

Capabilities that wait. Partner onboarding, contract negotiation, incentive calculations, commission processing, global inventory optimisation, advanced analytics, executive dashboards, pricing approvals, large-scale resource planning, and non-urgent profile administration are all central-only. Delaying them does not typically block an assigned field task.

Conditional capabilities. Some functions are conditional. Local inventory consumption is recorded offline, but confirmation of stock availability across the enterprise requires connectivity. A technician can capture a replacement-part request offline; final approval and fulfilment are processed centrally.

Validation of the classification proceeded through interviews with field technicians, partner managers, network engineers, service coordinators, security teams, and compliance specialists. Common operational scenarios (rural tower maintenance, customer-premises installation, underground infrastructure work, disaster-recovery operations) were mapped against the classification to check that the offline-essential set was neither too permissive nor too restrictive. The resulting edge footprint is deliberately limited so that update, security, and governance overhead remain manageable.

5. Data Ownership Taxonomy

Concurrent-write behaviour at the edge depends on data class. The architecture recognises three classes (Figure 3).

Figure 3. Data-ownership taxonomy: three concurrency classes governing what may be written at the edge



Field-owned append-only. Inspection observations, photographs, equipment readings, installation evidence, and task-completion timestamps are safely concurrent because they are recorded as immutable events rather than as updates to a shared record. Two technicians can independently add inspection evidence to the same site record without conflict; both events are preserved.

Central-owned read-at-edge. Partner status, customer entitlements, compliance eligibility, pricing, and role permissions are read at the edge but not freely overwritten. The central platform is authoritative. Local overrides are not permitted, and stale reads beyond the freshness window disable dependent actions.

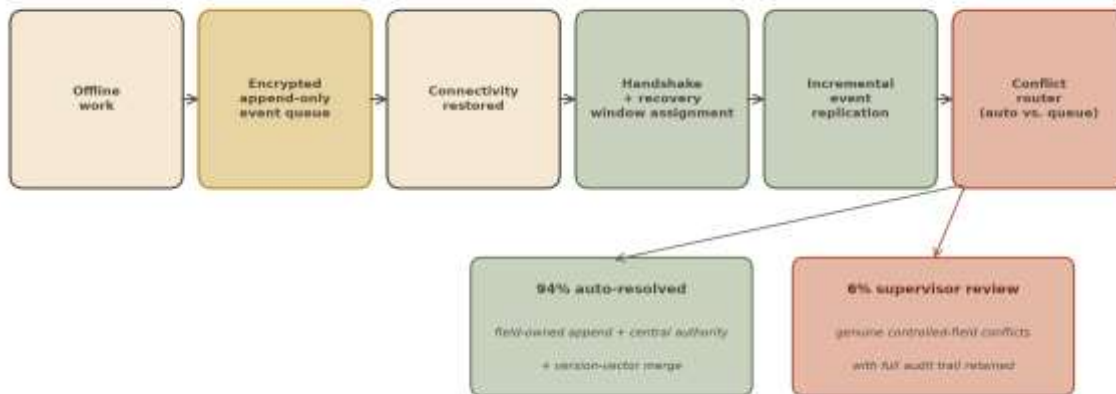
Leased single-writer. Work orders, inventory items, and other records with state transitions carry explicit ownership leases so that only one assigned node may transition their state during the lease window. Two nodes cannot both finalise the same work order; the first valid completion accepted by the central workflow advances the order state, and the second is flagged as a duplicate or routed to reconciliation. Inventory consumption is recorded locally as a reservation or usage event and validated centrally after reconnection to avoid silent negative stock.

The taxonomy replaces a global conflict-resolution policy with a per-class rule. This is what makes the architecture tractable at scale: automated resolution succeeds for the vast majority of events (94% in the pilot, §VIII) because the resolution strategy is deterministic per class.

6. Synchronisation Protocol

Synchronisation is event-driven when connectivity is available and queue-based when the device is offline. Figure 2 shows the end-to-end flow.

Figure 2. Offline-to-online synchronisation flow: from encrypted event queue to conflict router



Local event queue. Every local transaction is stored in an encrypted append-only queue with a unique event identifier, device identifier, user identity, timestamp, entity version, and synchronisation status. The queue is durable across power loss and unexpected device restart.

Handshake and recovery-window assignment. When a device reconnects, the synchronisation service first authenticates the device, refreshes security policies, exchanges version metadata, and assigns a recovery window based on server load and device priority. Critical field updates (completed work orders, safety incidents) are synchronised first; lower-priority items (image uploads, historical logs) are deferred.

Incremental event-based replication. Rather than comparing complete datasets, devices transmit only immutable events that have not previously been acknowledged. The synchronisation service processes events asynchronously through distributed message queues, enabling horizontal scaling and preventing long-running database transactions. Compared with periodic full synchronisation, the incremental approach reduced synchronisation network traffic by approximately 58% in the pilot.

Version-vector exchange and conflict detection. Each update includes the version of the record that the user originally received. If the central version has changed, the platform detects the conflict rather than silently overwriting data. Conflict evaluation is at record and field granularity: non-conflicting field-level updates commit immediately; only genuinely conflicting fields route to the reconciliation service.

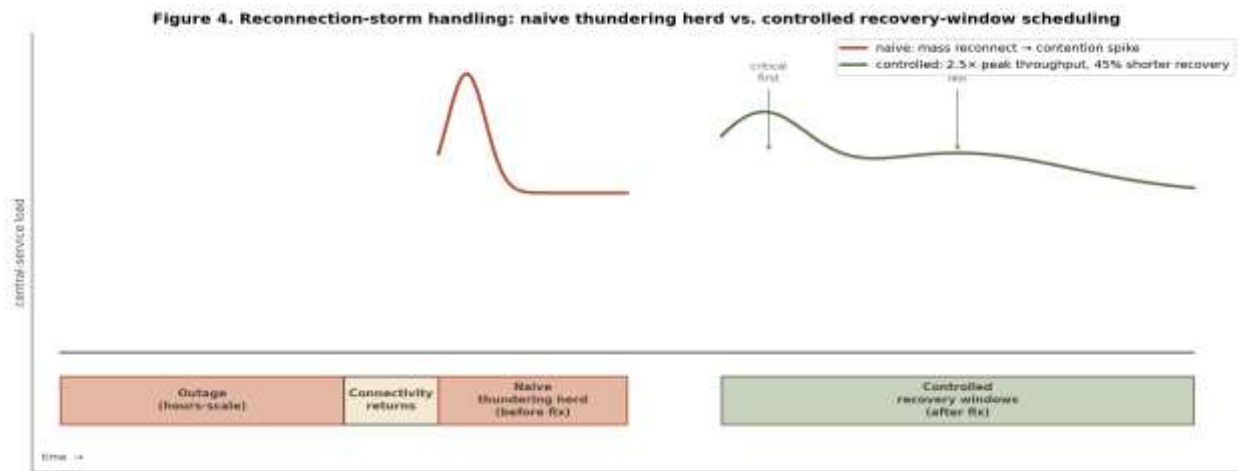
Conflict router. Automated rules resolve predictable cases based on source authority, field ownership, timestamps, transaction sequence, and workflow state. Field-owned append-only events pass through unchanged. Central-owned data always wins over edge modifications. Leased-record transitions are validated against the current lease holder. Unresolved conflicts enter a reconciliation queue containing the original version, all competing updates, user and device details, timestamps, and supporting evidence, with both versions and a complete audit trail retained for supervisor review.

Tombstones and freshness expiration. The platform maintains tombstone records for deletions so that a record removed centrally does not reappear when an older edge node synchronises later. Data-expiration policies prevent stale authorisation, pricing, or compliance information from remaining valid indefinitely.

Idempotent processing. Unique event identifiers and idempotent processing prevent duplicate updates when synchronisation is retried. If the same event arrives twice (network duplicate, client retry), the second occurrence is discarded rather than applied.

7. Reconnection-Storm Handling

The most significant production surprise encountered during pilot operation was not a synchronisation failure during network outage but a large-scale reconnection storm that occurred when hundreds of edge devices regained connectivity simultaneously after an extended regional cellular outage. Figure 4 illustrates the phenomenon.



The failure mode. During the outage, technicians continued working normally using offline capabilities. Each device accumulated hundreds of locally generated events including work-order updates, inventory transactions, photographs, digital signatures, and inspection records. When mobile connectivity returned, nearly all devices attempted to synchronise immediately. This created an unexpected surge of concurrent API requests. Database contention increased, message queues grew rapidly, API latency rose, and synchronisation completion times became much longer than expected. Repeated retries from clients amplified the load and created a cascading backlog.

Redesign — controlled recovery windows. Rather than allowing every device to reconnect at once, each edge node performs a lightweight handshake with the synchronisation service and receives a synchronisation window based on server load and device priority. Critical field updates synchronise first; lower-priority items defer to windows further in the future. The window scheduler prevents mass reconnection from producing a synchronised spike at the central service.

Incremental event replication replacing full sync. As introduced in §VI, incremental event-based replication reduces transmitted volume and enables asynchronous message-queue processing. Combined with recovery-window scheduling, this prevents long-running central transactions during the storm.

Adaptive client-side backoff. Fixed-interval retries synchronise unintentionally into repeated waves that stress the central service in step with the retry interval. The redesigned client uses exponential backoff with randomised jitter [10] so that retries naturally distribute over time.

Fine-grained conflict detection. During the storm some work orders had been modified independently by supervisors at headquarters while technicians were offline. Rejecting entire synchronisation batches would have been wasteful; the platform evaluates conflicts at the individual record and field level. Non-conflicting updates commit immediately, only genuinely conflicting records route to reconciliation, and technicians see fewer synchronisation failures.

Following these changes the platform demonstrated substantially better resilience during subsequent outage simulations. Peak synchronisation throughput increased by approximately 2.5×, average post-outage synchronisation time decreased by approximately 45%, and synchronisation failures caused by concurrent reconnection events decreased by more than 80%. Technicians no longer experienced prolonged delays after connectivity returned.

8. Pilot Evaluation

The architecture was evaluated through a pilot deployment representative of the operational characteristics of a large telecom field organisation. The pilot involved 50 distributed field sites (regional service centres, customer-premises installation teams, remote maintenance locations) across urban, suburban, and rural environments with varying network availability. Approximately 320 edge devices participated (rugged tablets, technician smartphones, vehicle-mounted gateways, branch edge servers), and the deployment supported approximately 180 partner organisations. More than 850 field technicians and partner engineers used the platform throughout the evaluation period. The synchronised dataset included 120,000+ work orders, 95,000 customer site records, 40,000 network assets, 18,000 inventory items, 6,500 partner user accounts, and 2.8 million synchronisation events.

Table 1 — pilot scale summary.

Dimension	Value
Field sites	50

Edge devices	320
Partner organisations	180
Field engineers / technicians	850+
Work orders processed	120,000+
Customer site records	95,000
Network assets	40,000
Inventory items	18,000
Partner user accounts	6,500
Synchronisation events	2.8 M

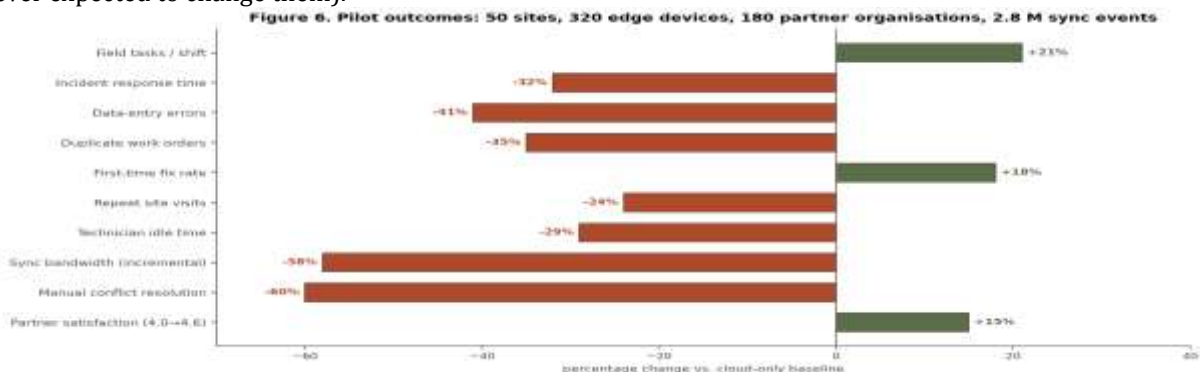
Table 2 — synchronisation and conflict outcomes.

Metric	Result
Successful synchronisation rate	99.3 %
Average sync latency (normal connectivity)	4.8 s
Average sync latency (post-outage bulk recovery)	18.6 s
Failed synchronisations requiring retry	0.7 %
Duplicate events detected and discarded	0.42 %
Sync bandwidth reduction (incremental vs. full sync)	58 %
Peak throughput improvement after redesign	2.5×
Reduction in post-outage sync time	45 %
Records requiring conflict evaluation	0.83 %
Automatically resolved conflicts	94 %
Manual reconciliation required	6 % of conflicts (≈ 0.05 % of all events)
Reduction in manual conflict handling after redesign	60 %

Table 3 — offline operation and productivity outcomes.

Metric	Result
Maximum validated offline operation	72 hours
Typical uninterrupted offline work session	6-10 hours
Field operations maintained during connectivity loss	> 95 %
Critical workflows executable offline	100 % of planned field workflows
Completed work orders per shift	+21 %
Incident response time	32 % faster
Local data retrieval latency	< 300 ms
First-time fix rate	+18 %
Repeat site visits	-24 %
Technician idle time	-29 %
Data-entry errors	-41 %
Duplicate work-order submissions	-35 %
Partner satisfaction score	4.0 $\rightarrow$ 4.6 / 5

Figure 6 shows the ten productivity improvements against the cloud-only baseline. The pattern is consistent with the architectural argument: capabilities placed at the edge produce the largest gains where connectivity was previously the dominant constraint, and central-only capabilities produce no gain (because the design was never expected to change them).



The platform maintained edge-node availability of 99.7 % and central synchronisation-service availability of 99.95 %. Data successfully recovered after unexpected device restart was 100 % (0 lost field transactions). Event-processing success was 99.99 %.

9. Discussion

When edge deployment is worth the complexity. Organisations with field teams working mainly in cities with strong consistent networks typically benefit most from enhancing central connectivity rather than adopting an edge architecture. Centralised systems are easier to manage in governance, security, updates, data, and compliance, and they avoid the challenges of keeping distributed systems in sync. If field staff can reliably access the central platform without meaningful delays, the extra complexity and cost of edge infrastructure may not be justified.

Edge deployment is most valuable when connectivity is the main obstacle. In telecom operations that is common: technicians work in remote areas, underground, at customer sites, or during outages. In those cases waiting for a network connection slows service, reduces productivity, risks missing SLA targets, and harms customer satisfaction. The cost of lost productivity often outweighs the investment in edge infrastructure.

Four questions decide the trade-off: how often field teams lose reliable connectivity; the cost of downtime per postponed task; whether business processes can tolerate deferred data synchronisation; and whether the organisation is ready to manage distributed device fleets. Where the first three answers point strongly toward "yes" and the fourth is achievable, edge deployment is warranted. Hybrid deployments (centralised strategic functions plus edge operational functions) often provide the highest return.

Trade-offs the architecture accepts. Selective replication limits the edge footprint but requires disciplined capability classification (§IV). Version-based optimistic concurrency requires deterministic conflict-resolution rules per data class (§V). Reconnection-storm handling requires controlled recovery windows that intentionally delay full synchronisation for lower-priority events (§VII). Compliance with authorisation-freshness windows requires the platform to disable high-risk actions when cached authorisations expire, which is a deliberate reduction of functionality rather than a bug.

Trade-offs the architecture refuses. The design does not attempt strong consistency between edge nodes because that is impractical under intermittent connectivity. It does not permit unbounded offline execution of high-risk actions after authorisation expiry. It does not treat edge nodes as authoritative for shared master data, even in the presence of long partitions. Each refusal is a deliberate operational choice, not a limitation waiting to be resolved.

10. Limitations and Future Work

- Pilot scale (50 sites, 320 devices, 180 partners). The architecture is designed to scale horizontally through stateless API layers and distributed message queues, and larger deployments would provide additional validation.
- Single organisational context. Cross-organisation replication would test the architecture against different partner ecosystems and compliance regimes.
- Event schemas. The pilot standardised event schemas for the deployment's workflows. Cross-industry replication would require schema harmonisation.
- Machine learning at the edge. Selected AI-assisted functions (troubleshooting guidance) were included in the pilot but are not the paper's focus. Larger models and federated-learning approaches at the edge are natural extensions.
- Multi-partner reconciliation. The paper addresses conflicts between one partner's edge and the central platform; multi-partner cross-referencing (for example, work orders touching two partner organisations simultaneously) is a natural extension.

11. Conclusion

The paper contributes an edge-deployed PRM architecture in which offline-essential capabilities execute locally on ruggedised edge nodes while the central PRM platform remains authoritative, and an event-driven synchronisation fabric bridges the two. A five-criteria offline-capability classification, a three-class data-ownership taxonomy governing concurrent writes, a version-based synchronisation protocol with idempotent processing, and a reconnection-storm handling design combining controlled recovery windows, incremental event replication, and adaptive backoff with jitter together enable uninterrupted telecom field operations under intermittent connectivity. A pilot deployment across 50 field sites, 320 edge devices, 180 partner organisations, and 850+ field engineers processed 120,000+ work orders and 2.8 million synchronisation events, achieving a 21% increase in tasks completed per shift, a 32% faster incident response, a 41% reduction in data-entry errors, a 99.3% synchronisation success rate, 94% automatic conflict resolution, and an increase in partner satisfaction from 4.0 to 4.6 on a 5-point scale. The design demonstrates that distributing essential partner-

management capabilities to the network edge can significantly improve field productivity, reduce operational delays, enhance data accuracy, strengthen service resilience, and increase partner satisfaction while maintaining consistency with the centralised enterprise platform.

Data Availability

Pilot metrics are summarised in Tables 1-3 of §VIII. Detailed workflow schemas, capability-classification worksheets, and synchronisation-service configuration templates are available on request subject to non-disclosure. No customer or partner identifying data appears in this work.

Ethics

No human-subjects data. Pilot participants used the platform in their normal course of work and provided consent for aggregated performance measurement.

Competing Interests

The author declares no competing interests.

References

- [1] J. O. Kephart, D. M. Chess, "The Vision of Autonomic Computing," *IEEE Computer*, vol. 36, no. 1, pp. 41-50, 2003. DOI: 10.1109/MC.2003.1160055.
- [2] W. Shi, J. Cao, Q. Zhang, Y. Li, L. Xu, "Edge Computing: Vision and Challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637-646, 2016. DOI: 10.1109/JIOT.2016.2579198.
- [3] M. Satyanarayanan, "The Emergence of Edge Computing," *IEEE Computer*, vol. 50, no. 1, pp. 30-39, 2017. DOI: 10.1109/MC.2017.9.
- [4] N. Abbas, Y. Zhang, A. Taherkordi, T. Skeie, "Mobile Edge Computing: A Survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450-465, 2018. DOI: 10.1109/JIOT.2017.2750180.
- [5] A. Feyerke, "Designing Offline-First Web Apps," *A List Apart*, no. 386, 2013.
- [6] C. Fidge, "Timestamps in Message-Passing Systems that Preserve the Partial Ordering," *Proc. 11th Australian Computer Science Conf.*, pp. 56-66, 1988.
- [7] M. Shapiro, N. Preguiça, C. Baquero, M. Zawirski, "Conflict-Free Replicated Data Types," in *Stabilization, Safety, and Security of Distributed Systems*, LNCS 6976, Springer, pp. 386-400, 2011. DOI: 10.1007/978-3-642-24550-3_29.
- [8] D. B. Terry, M. M. Theimer, K. Petersen, A. J. Demers, M. J. Spreitzer, C. H. Hauser, "Managing Update Conflicts in Bayou, a Weakly Connected Replicated Storage System," *Proc. 15th ACM SOSP*, pp. 172-183, 1995. DOI: 10.1145/224056.224070.
- [9] R. Nishtala et al., "Scaling Memcache at Facebook," *Proc. 10th USENIX NSDI*, pp. 385-398, 2013.
- [10] AWS Architecture Blog, "Exponential Backoff and Jitter," Amazon Web Services, 2015.
- [11] TM Forum, "Best Practices for Field Service Management in Telecommunications," *TM Forum Reference Guide*, 2022.
- [12] ETSI, "Mobile Edge Computing (MEC): Framework and Reference Architecture," *ETSI GS MEC 003*, 2022.
- [13] Progressive Web Applications community, "Service Worker + Cache API offline patterns," *MDN Web Docs*, ongoing.
- [14] R. Rodrigues, P. Druschel, "Peer-to-peer systems," *Communications of the ACM*, vol. 53, no. 10, pp. 72-82, 2010. DOI: 10.1145/1831407.1831427.