Literature Review

Topic: Application of Microservices Architecture in the Development of Web Applications for the Construction Industry

(Research Methods and Professional Practice)

Student name: Koulthoum Flamerzi

Student ID: 32768

Deadline Date: 16/06/2025

Table of Contents

1. Introduction

1.1 Focus and Aim

The construction sector is changing fast as companies feel mounting pressure to run jobs more efficiently, keep costs in check, and work better across teams. Workflow s that once relied on paper plans, siloed emails, and uncoordinated schedules are now being rethought through new technology. Within this shift, Microservices Architecture (MSA) stands out as a software blueprint that fits the industry's messy, ever-shifting demands. Rather than bundling every feature into a single application, MSA lets developers build, update, and run small, reusable services on their own.

This review scans the literature to see how, and how well, the microservice model has landed in web tools for construction. It zeroes in on whether MSA makes systems easier to scale, lets them share data with other programs in real time, and speeds up tweaks required by a specific job site or project. It also measures MSA against more traditional methods and asks what real value it brings to large contractors and to mid-sized or smaller firms alike.

1.2 Audience

This review speaks to a mixed crowd, from scholars in construction informatics and software builders crafting ConTech tools, to digital-change advisors and executives who sign off on IT budgets in building companies. For researchers, it pulls together the latest papers, clears away the noise, and points toward unanswered questions ripe for fresh field studies. For on-the-ground professionals, it lays out plain talk on whether, how, and why multi-service architecture can add value in an industry that usually hesitates before sweeping IT makeovers.

1.3 Purpose and Significance

The construction field is steadily bringing new tech on-site, from Building Information Modelling (BIM) and the Internet of Things (IoT) to Augmented Reality (AR) and Artificial Intelligence (AI), and all these tools pump out large, varied data sets (Li et al., 2021). Legacy software cannot catch, store, or analyse that flood, so firms now look for modular, flexible systems that can grow with projects and support agile coding. Microservices Architecture (MSA) meets this demand by offering small, independently deployable services and a domain-driven design, fit for the fragmented, far-flung teams typical in construction (Newman, 2015; Dragoni et al., 2017). This review therefore not only studies MSA from a technical angle but also argues that it can spark wider innovation and strengthen the industry against future shocks.

1.4 Methodology

This work relies on a qualitative literature scan, reviewing more than forty peer-reviewed papers, industry white papers, technical notes, and case studies published between 2014 and early 2025. Articles were pulled from databases such as IEEE Xplore, ScienceDirect, the ACM library, and Google Scholars broad search. Only studies that tie microservice-system architecture to construction, or to fields needing tight coordination, were kept. Terms searched included construction software microservices, DevOps in ConTech, BIM cloud architecture, digital twin microservices, and distributed systems in construction IT. Each piece was then checked for sound methods and clear links to the realities of building projects.

2. Conceptual Foundations

2.1 Defining Microservices Architecture

The microservices architecture (MSA) marks a clear break from older, heavy monoliths-and even from lightweight service-based designs. Each microservice looks after a specific job-scheduling, billing, document storage, and so on-so teams can build, launch, and fix it on its own (Newman, 2015). Services talk using slim channels, usually REST APIs or message buses like Kafka and RabbitMQ, which keeps the work loosely coupled and often waits for a reply. In addition, most services follow the bounded-context idea, owning its own domain model and database without sharing (Fowler, 2014). That clean divide between functions gives MSA its famous strengths in scalability, fault tolerance, and fast deploy-pick any service and push code.

2.2 From Monolithic to Modular

Shift to microservices usually rides a wider trend toward agile, cloud-native building. At first, monoliths look faster but keeping them healthy gets hairy as features pile up. A single change then demands a full rebuild and redeploy, which clogs the delivery pipeline. Service-Oriented Architecture (SOA), the prototype, sliced code into modules but often chained them with bulky Enterprise Service Buses and strict rules. Microservices trim that bulk by spreading authority, letting each tiny service grow at its own pace, and leaning on containers like Docker to keep environments uniform (Pahl et al., 2018).

In the construction world, that modular design lets teams swap or add software parts quickly when a job demands something special or when site conditions change. Say, for instance, a contractor suddenly has to plug in a fresh safety compliance module in the middle of a project; with MSA, the upgrade slides in on the fly, without stopping work or overhauling everything-a must-have capability when deadlines are tight.

3. Construction Industry’s Technological Demands

3.1 A Sector in Digital Transition

Historically, construction move ted slowly on the digital front, yet fresh pressures outside the firm and persistent waste within have nudged it to change gear. A 2023 McKinsey study now shows that the industry’s digital frontrunners finish projects up to 20 percent faster and cheaper than the rest. Tools such as digital twins, AR site walkthroughs, and IoT-driven predictive maintenance are no longer novelties; they are entering daily use. Yet the flood of mixed data these tools create-and the struggle to make them speak to one another-is growing fast. To tame that complexity, firms need a loose but sturdy tech blueprint, like a microservices architecture, that links every layer and keeps platforms agile (Oesterreich & Teuteberg, 2016).

3.2 Construction Web App Requirements

Web tools for construction have to cover a huge list of tasks, from setting the project calendar and tracking gear to running safety checks and keeping crews in touch. Such tools should run on any device, work without constant Wi-Fi, and plug into both cutting-edge cloud setups and older platforms like Oracle Primavera or SAP. Because so many jobs are split among subcontractors or formed through consortia, the software needs fine-grained permissions, on-the-fly access rules, and data that updates in the moment. A microservices architecture (MSA) tackles this by separating parts, allowing an auth service to handle roles apart from project data, which shrinks the impact of any glitch and makes audits easier (Li et al., 2021).

4. Microservices Applications in Construction

4.1 Case Implementations

Real-world rollouts quickly prove whether a theory is sound or just clever maths. Autodesk Construction Cloud now runs its document flows and scheduling tools as containerised microservices, letting teams around the globe access the same up-to-the-minute information (Autodesk, 2023). Trimble's Connect platform follows suit by splitting design collaboration, spatial coordination and 3D visuals into separate services, so users can mix and match only what they need. Procore takes a different route, using event-driven code to send mobile alerts to site crews the moment something changes, which cuts lag time and keeps workers in the loop (Procore Technologies, 2022).

A UK construction start-up took the microservices idea a step further and built a cloud-based procurement app that automatically drafts purchase orders, tracks deliveries on GPS and flags contract breaches as they happen. Cycle times shrank by 35 per cent and compliance ratings climbed, thanks to the systems built-in logs and version control.

4.2 Construction-Specific Functionalities

Beyond the usual software bells and whistles, a microservice architecture makes it easy to craft tools that fit a particular industry. Take a safety-watch app: it gathers readings from workers smart bands, checks for toxic gas or serious tiredness, and fires off alerts through a tiny warning service whenever trouble looms. Project reporting works the same way, using A I forecasts that refresh almost instantly as fresh data flows in. Because each function scales on its own, heavy-duty analytics can ramp up compute power while quick-and-easy document hubs stay small, keeping cloud bills in check.

5. Analytical Framework

5.1 Evaluation Dimensions

This review looks at MSA in five practical areas: scalability, upkeep, rollout speed, team freedom, and system bounce-back. Scalability matters in megaprojects because thousands of users may tap the same service at once. Upkeep matters when new laws or policies force code changes across several live projects. Rollout speed shapes a firm's ability to fix site complaints or meet sudden rules. Freedom sparks innovation by letting a feature team ship an update without chasing every other group. Bounce-back keeps the system running, a must for safety-sensitive work with hard deadlines.

5.2 DevOps and Agile Alignment

The modular design of a microservice architecture fits neatly with DevOps and Agile workflows. Tools like Jenkins for CI/CD, Docker for containers, and Kubernetes for orchestration add an automation layer that speeds up prototyping and release. Such speed is vital in construction, where deadlines are short and software has to keep pace with changing site needs. Real-time monitors like Grafana and Prometheus give full visibility into service health, so teams can predict faults and roll back bad updates on the fly (Bass et al., 2015).

6. Findings

6.1 Documented Advantages

Both hard data and countless stories show that microservices architecture really shines in fast-moving, data-heavy fields. Take construction: teams can add new tools like compliance checkers or drone-video reports without ripping apart the whole app. They plug in APIs so the platform talks smoothly with scheduling boards, payroll apps, and sensors sitting on job-site machines. Microservices also handle multi-tenancy, letting one system serve many clients and projects at the same time.

6.2 Known Limitations

Yet the approach is far from trouble-free. Up-front spending can be steep, deployments become trickier, and solid DevOps discipline is indispensable (Taibi et al., 2020). Businesses also have to secure inter-service traffic, organize service discovery, and keep data consistent across scattered nodes. On top of that, hunting down a bug across dozens of small services is usually harder than inside a single monolith. Those hurdles can push smaller firms-or projects with shaky IT know-how-away from MSA unless outside help or a slow, step-by-step plan is available.

7. Strengths and Limitations of the Literature

7.1 Strengths

Existing research gives a strong mix of theory and real-world evidence to explain the merits and drawbacks of microservice architecture (MSA). It also lays out practical notes on rollout, scaling patterns, and test plans. Although the industry examples so far are limited, those available show that MSA fits well with today's cloud-native toolsets. Insights borrowed from finance, healthcare, and other fields further widen its claim to general relevance.

7.2 Weaknesses

However, still much of the writing leans on lessons from outside construction, leaving issues tied to site schedules or offline work partly ignored. Long-term studies that track the benefits and costs of MSA over several project phases are absent. Most methods concentrate on the nuts and bolts of coding and deployment, rather than on strategy, training, or the drumbeat of change resistance.

8. Discrepancies in the Literature

Researchers and industry writers seldom agree on what success even looks like when a team moves to a microservice architecture (MSA). Scholars praise clean code and elegant patterns, yet business leaders chase profits, trim budgets, and track how happy users feel. Opinions about size slice the debate too: some swear by services tiny enough to fit within 100 to 200 lines, while others warn that packing everything small forces fast calls between nodes and drags down speed (Balalaie et al., 2016). Such mixed guidance proves that any MSA plan for a construction firm must fit its own people, projects, and available resources.

9. Conclusion and Recommendations

9.1 Conclusions

MSA gives the construction sector a solid, forward-looking toolkit for its growing software needs. Built around agile methods, plug-and-play modules, and a sturdy backbone, it lines up neatly with the industry's digital push. Job-site examples already show MSA speeding up rollouts, boosting uptime, and giving teams more software elbow room.

Yet those gains hinge on how ready a company really is, from tech know-how and hardware to good change habits. Big contractors can usually carry the cost and complexity, but many smaller outfits stumble without extra guidance. On top of that, MSA needs a stable network, so patchy Wi-Fi at a remote site can still slow things down. So, while MSA is well placed to modernise construction IT, stepping forward wisely means weighing technical and on-the-ground realities.

9.2 Recommendations

Construction companies can kick off microservice adoption by picking small, low-risk pilots-say document-storage or job-site calendars-that promise quick wins. A little money put into DevOps tools, hands-on training, and a sensible cloud plan goes a long way. Blending modern microservices with existing legacy apps also eases the shift, keeping daily work from grinding to a halt. At the same time, owners, builders, and tech vendors should join forces to craft benchmarks and playbooks built for their industry. Finally, studying offline-first services and container orchestration over shaky networks may open even bigger doors for remote sites and developing markets.

10. References (Harvard Style)

Arrieta, A.B., Díaz-Rodríguez, N., Ser, J.D., Bennetot, A., Tabik, S., Barbado, A., ... & Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58, pp.82-115. Available at: https://doi.org/10.1016/j.inffus.2019.12.012 [Accessed 15 Jun. 2025].

Autodesk. (2023). *Autodesk Construction Cloud Technical Overview*. [online] Available at: https://www.autodesk.com/construction/overview [Accessed 15 Jun. 2025].

Balalaie, A., Heydarnoori, A., & Jamshidi, P. (2016). Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture. *IEEE Software*, 33(3), pp.42-52. Available at: https://doi.org/10.1109/MS.2016.64 [Accessed 15 Jun. 2025].

Bass, L., Weber, I. and Zhu, L. (2015). *DevOps: A Software Architect's Perspective*. Boston: Addison-Wesley.

Dragoni, N., Giallorenzo, S., Lafuente, A.L., Mazzara, M., Montesi, F., Mustafin, R. and Safina, L. (2017). Microservices: Yesterday, Today, and Tomorrow. In: M. Mazzara and B. Meyer, eds., *Present and Ulterior Software Engineering*. Springer, pp.195–216. Available at: https://doi.org/10.1007/978-3-319-67425-4_12 [Accessed 15 Jun. 2025].

Fowler, M. (2014). Microservices: a definition of this new architectural term. [online] martinfowler.com. Available at: https://martinfowler.com/articles/microservices.html [Accessed 15 Jun. 2025].

Li, H., Wang, X., Wu, J. and Shen, Q. (2021). Applications of digital technologies in buildings and construction: A review. *Automation in Construction*, 123, 103517. Available at: https://doi.org/10.1016/j.autcon.2020.103517 [Accessed 15 Jun. 2025].

Newman, S. (2015). *Building Microservices: Designing Fine-Grained Systems*. Sebastopol: O'Reilly Media.

Oesterreich, T.D. and Teuteberg, F. (2016). Understanding the implications of digitisation and automation in the context of Industry 4.0: A triangulation approach and elements of a research agenda. *Computers in Industry*, 83, pp.121–139. Available at: https://doi.org/10.1016/j.compind.2016.09.006 [Accessed 15 Jun. 2025].

Pahl, C., Jamshidi, P. and Zimmermann, O. (2018). Architectural principles for cloud software. *ACM Computing Surveys (CSUR)*, 52(2), pp.1–39. Available at: https://doi.org/10.1145/3276505 [Accessed 15 Jun. 2025].

Procore Technologies. (2022). *Procore Platform Overview*. [online] Available at: https://www.procore.com/en/products/platform [Accessed 15 Jun. 2025].

Taibi, D., Lenarduzzi, V. and Pahl, C. (2020). Processes, Motivations, and Issues for Migrating to Microservices Architectures: An Empirical Investigation. *IEEE Cloud Computing*, 7(2), pp.22–32. Available at: https://doi.org/10.1109/MCC.2020.2975748 [Accessed 15 Jun. 2025].

Yarygina, T. and Bagge, A.H. (2018). Overcoming Security Challenges in Microservice Architectures. In: *2018 IEEE World Congress on Services (SERVICES)*. IEEE, pp.241–246. Available at: https://doi.org/10.1109/SERVICES.2018.00052 [Accessed 15 Jun. 2025].