David McNabb

david@aiapplied.uk

Abstract

[Draw your reader in with an engaging abstract. It is typically a short summary of the document.   
When you’re ready to add your content, just click here and start typing.]

Document Title

Subtitle

Smart Manufacturing Data Hub (SMDH)

**Architecture Design Options**

Comprehensive Analysis of Four Architecture Approaches

|  |  |
| --- | --- |
| Document Version | 2.0 FINAL |
| Date | October 2025 |
| Status | For Review |
| Classification | Internal Use |
| Author | AI Applied Technical Team |
| Intended Audience | Technical Architects and Business Stakeholders |

# 1. Executive Summary

This document presents a comprehensive analysis of four distinct architectural approaches for the Smart Manufacturing Data Hub (SMDH) platform. The SMDH is designed as a cloud-native, multi-tenant Internet of Things (IoT) platform that empowers small and medium-sized manufacturing enterprises with real-time visibility into their operations.

## 1.1 Document Purpose

The purpose of this document is to provide both technical architects and business stakeholders with a clear understanding of four viable architecture options for the SMDH platform. Each option has been analysed in detail, considering factors such as cost implications, technical complexity, performance characteristics, scalability, implementation timelines, and operational considerations.

## 1.2 The Four Architecture Options at a Glance

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Criteria | Option A: Flink | Option B: Snowflake | Option C: SiteWise | Option D: Timestream |
| Monthly Cost (30 tenants) | £2,500-4,200 | £2,170-3,450 | £6,334 | £3,965 |
| Complexity | Very High | Low | Medium-High | Medium |
| Timeline | 24 weeks | 20 weeks | 28 weeks | 22 weeks |
| Best Suited For | Sub-second latency required | Most scenarios (Recommended) | SiteWise expertise available | AWS-native mandate |

## 1.3 Key Finding: All Options Meet Budget Requirements

An important finding from our analysis is that all four options meet the budget constraint of £200-300 per tenant per month. A previous cost calculation error for Option C has been corrected, reducing its estimated monthly cost by 70% (from £21,150 to £6,334 for 30 tenants). This means the decision should be based on factors such as technical requirements, team capabilities, and strategic preferences rather than basic affordability.

## 1.4 Recommended Approach

**🏆 Recommended: Option B (Snowflake-Leveraged Architecture)**

For the majority of manufacturing IoT scenarios, Option B (Snowflake) provides the best balance of:

* Lowest cost: £2,170-3,450/month (£72-115 per tenant)
* Lowest complexity: Only 5 core services to manage
* Fastest time-to-market: 20 weeks to production
* SQL-first development: Familiar to most engineering teams
* Native multi-tenancy: Secure row-level security built-in
* Proven in manufacturing: Used by major industrial companies

The default configuration provides 60-65 second dashboard update latency, which can be improved to <5 seconds for critical alerts by adding a Lambda "fast-path" (additional £100-150/month and 2 weeks implementation).

## 1.5 Alternative Options

Whilst Option B is recommended as the default choice, the other options may be preferable in specific circumstances:

**Option D (AWS-Native):** Select this if your organisation has a mandate for AWS-only services (no Snowflake). Provides excellent multi-tenancy, Grafana dashboards, and unified time-series storage. Cost: £3,965/month (15% premium over Option B).

**Option A (Flink-Based):** Select this ONLY if sub-second latency is legally or regulatorily required for all data processing. Provides advanced stream processing but requires Flink expertise and accepts highest complexity. Cost: £2,500-4,200/month.

**Option C (SiteWise):** Select this only if your team has deep AWS IoT SiteWise expertise and you're serving <10 tenants (multi-tenancy limitations). Provides managed asset modelling but requires custom dashboard development. Cost: £6,334/month (most expensive).

## 1.6 How to Use This Document

This document is structured to support both detailed technical review and high-level business decision-making:

* Technical Architects: Review Sections 4-7 for detailed component descriptions, data flows, and technical trade-offs for each option
* Business Stakeholders: Focus on Section 1 (this summary), Section 8 (comparisons), and Section 9 (recommendations)
* Decision-Makers: Review the comparison matrices in Section 8 and the decision framework in Section 8.5
* All Readers: Refer to Section 10 (Glossary) for explanations of technical terms. Terms are explained in plain English throughout the document.

Each architecture option (Sections 4-7) includes a placeholder for inserting the corresponding architecture diagram created in Draw.io. Please insert these diagrams before final distribution.

# Table of Contents

1. 1. Executive Summary

* 1.1 Document Purpose
* 1.2 The Four Architecture Options at a Glance
* 1.3 Key Finding: All Options Meet Budget Requirements
* 1.4 Recommended Approach
* 1.5 Alternative Options
* 1.6 How to Use This Document

# 2. System Overview and Requirements

## 2.1 What is the Smart Manufacturing Data Hub?

The Smart Manufacturing Data Hub (SMDH) is a cloud-based software platform designed to help small and medium-sized manufacturing enterprises (SMEs) monitor and analyse their operations in real-time. Think of it as a centralised "command centre" where data from factory equipment, environmental sensors, energy metres, and tracking systems flows together to provide actionable insights.

Unlike traditional manufacturing systems that require extensive IT infrastructure and expertise, SMDH is designed as a self-service platform. Companies can independently register, configure their equipment, and start viewing data—all through an intuitive web interface without requiring specialised technical knowledge.

### How It Works (Simplified)

The platform operates through a straightforward workflow:

**1. Company Registration:** Manufacturing companies create accounts through a self-service portal. They provide basic information about their organisation and can immediately begin setup.

**2. Site and Device Configuration:** Companies register their manufacturing facilities (sites) and the equipment they want to monitor. A guided wizard helps configure sensors, metres, and tracking devices with minimal technical input.

**3. Automatic Data Collection:** Once configured, sensors begin sending data automatically. Machine status, environmental readings, energy consumption, and job locations flow continuously into the platform.

**4. Data Processing and Storage:** The system receives, validates, and stores data securely. Each company's data is completely isolated from others. Historical data is retained according to compliance requirements.

**5. Real-Time Dashboards:** Pre-built dashboards automatically display relevant information based on the types of equipment registered. Users see real-time status, historical trends, and analytics.

**6. Alerts and Notifications:** The system monitors data against configurable thresholds. When issues arise (equipment failure, poor air quality, excessive energy use), users receive immediate notifications via email or SMS.

## 2.2 Key System Requirements

### Data Volume and Performance Requirements

The platform must handle significant data volumes whilst maintaining responsive performance:

|  |  |
| --- | --- |
| Requirement | Target Specification |
| Daily data ingestion | 2.6 to 3.9 million data points per day |
| Real-time monitoring latency | Under 1 second for dashboard updates |
| Alert notification time | Under 10 seconds from trigger event to notification |
| Dashboard query response | Under 5 seconds for complex analytics queries |
| System availability | 99.9% uptime (maximum 8.76 hours downtime per year) |

These requirements ensure users have immediate visibility into their operations. A machine failure or air quality issue must be detected and escalated within seconds, not minutes, to enable timely response.

### Multi-Tenancy and Data Isolation

As a Software-as-a-Service (SaaS) platform serving multiple manufacturing companies simultaneously, the SMDH must provide absolute data isolation. This is non-negotiable for several reasons:

* Security: Company A must never access Company B's data under any circumstances
* Compliance: GDPR and industry regulations require strict data segregation
* Intellectual Property: Manufacturing data often contains trade secrets and proprietary processes
* Performance: One company's heavy usage must not impact another company's performance
* Scalability: The system must efficiently serve 30-40 companies initially, scaling to 100+ over time

**Technical Implementation:**

Multi-tenancy can be implemented in several ways, and this is a key differentiator between the four architecture options:

**• Row-Level Security (RLS):** Database-enforced filtering that automatically restricts queries to authorised data. Used in Options A and B with Snowflake. Most secure.

**• Partition Keys:** Physical data separation using tenant identifiers as partition keys. Used in Option D with Timestream. Good security with performance benefits.

**• Tag-Based Filtering:** Application-layer filtering using metadata tags. Used in Option C with SiteWise. Requires careful implementation to avoid data leakage.

### Budget Constraints

The platform has a clear budget target to ensure economic viability for SMEs:

**Target Cost per Tenant: £200-300 per month**

For an initial deployment of 30 manufacturing companies, this translates to:

|  |  |
| --- | --- |
| Budget Component | Amount |
| Total Monthly Infrastructure Cost | £6,000 - £9,000 (30 tenants) |
| Per-Tenant Cost Target | £200 - £300 |
| Annual Infrastructure Budget | £72,000 - £108,000 |

Good news: All four architecture options evaluated in this document meet this budget constraint. The decision therefore comes down to factors other than basic affordability, such as complexity, team capabilities, and specific technical requirements.

## 2.3 Use Cases Supported

The SMDH platform must support diverse monitoring scenarios across manufacturing environments. Each use case has distinct characteristics and requirements:

### Use Case 1: Machine Utilisation Monitoring

**Business Objective:** Track how efficiently manufacturing equipment is being used to identify bottlenecks, reduce downtime, and improve overall equipment effectiveness (OEE).

**How It Works:**

Sensors attached to manufacturing equipment (CNC machines, lathes, presses, mills) collect data every second about the machine's operational state:

* State: Running, Idle, or Offline
* Cycle count: Number of production cycles completed
* Performance: Actual vs expected production rate
* Downtime events: When and why machines stop
* Error codes: Specific failure conditions

**Technical Specifications:**

|  |  |
| --- | --- |
| Specification | Value |
| Data frequency | 1 Hz (one reading per second) |
| Sensors per site | 30-45 machines typically |
| Communication protocol | LoRaWAN or MQTT |
| Daily data volume | ~2.6 million readings per site per day |
| Latency requirement | <1 second (immediate status visibility required) |

**Key Metrics Calculated:**

* Overall Equipment Effectiveness (OEE): Industry-standard metric combining availability, performance, and quality (target: >85% world-class)
* Availability: Percentage of scheduled time the machine is operational
* Performance: Actual production rate vs ideal production rate
* Utilisation: Percentage of time machines are actively producing
* Mean Time Between Failures (MTBF): Reliability metric
* Mean Time To Repair (MTTR): Maintenance efficiency metric

### Use Case 2: Air Quality Management

**Business Objective:** Monitor environmental conditions in manufacturing facilities to ensure worker health and safety, regulatory compliance, and optimal working conditions.

**How It Works:**

Environmental sensors positioned throughout the facility continuously monitor air quality parameters:

* CO₂ levels: Carbon dioxide concentration (target: <1000 ppm for good ventilation)
* VOCs: Volatile Organic Compounds from paints, solvents, adhesives
* Particulate Matter: PM1, PM2.5, PM4, PM10 (from grinding, cutting, welding)
* Temperature: Workspace thermal comfort
* Humidity: Moisture levels affecting comfort and processes
* Atmospheric Pressure: Baseline environmental measurement

**Technical Specifications:**

|  |  |
| --- | --- |
| Specification | Value |
| Data frequency | 1-minute intervals |
| Sensors per site | 10-15 sensors (distributed across facility) |
| Communication protocol | MQTT over WiFi/Ethernet |
| Daily data volume | ~200,000 readings per site per day |
| Alert requirement | <10 seconds for dangerous conditions (e.g., CO₂ >5000 ppm) |

**Regulatory Compliance:**

The platform must support compliance with:

* UK Health and Safety Executive (HSE) Workplace Exposure Limits (WELs)
* European Union Occupational Safety and Health Directives
* ISO 45001 Occupational Health and Safety Management
* COSHH (Control of Substances Hazardous to Health) Regulations

### Use Case 3: Energy Monitoring and Optimisation

**Business Objective:** Track electrical consumption to identify energy waste, reduce costs, and meet sustainability goals. Manufacturing typically represents 30-50% of operational costs for energy-intensive facilities.

**How It Works:**

Energy monitoring devices (installed at circuit breaker panels or on individual equipment) measure electrical parameters in real-time:

* Voltage: Electrical potential (V) - indicates power quality
* Current: Electrical flow (A) - indicates load
* Power Factor: Efficiency of electricity usage (target: >0.95)
* Real Power: Actual energy consumed (kW)
* Apparent Power: Total power drawn (kVA)
* Cumulative Consumption: Total kilowatt-hours (kWh) over time
* Cost Estimate: Energy cost based on tariff rates

**Technical Specifications:**

|  |  |
| --- | --- |
| Specification | Value |
| Data frequency | 15-second intervals |
| Monitors per site | 10-20 monitoring points (circuits + individual equipment) |
| Communication protocol | Modbus TCP or MQTT |
| Daily data volume | ~600,000 readings per site per day |
| Accuracy requirement | ±1% for billing-grade monitoring |

**Key Analytics:**

* Baseline Consumption: Establish normal usage patterns
* Peak Demand Analysis: Identify when and where peak usage occurs (affects tariffs)
* Power Factor Correction Opportunities: Improve efficiency and reduce reactive power charges
* Equipment Efficiency Comparison: Compare energy use across similar machines
* Cost Allocation: Distribute energy costs across departments or products
* Carbon Footprint Calculation: Convert kWh to CO₂ emissions for sustainability reporting

### Use Case 4: Job Location Tracking

**Business Objective:** Provide real-time visibility of work-in-progress (WIP) locations throughout the factory floor. Prevents lost jobs, reduces search time, and enables accurate delivery commitments.

**How It Works:**

Jobs are tagged with RFID tags or printed barcodes. As jobs move through the manufacturing process, workers or automated scanners record each movement:

* Job Start: When work begins on an order
* Station Arrival: Job reaches a workstation (e.g., "Welding Station 3")
* Station Completion: Work at that station finishes
* Quality Check: Inspection points
* Job Completion: Final product ready for shipping
* Exception Events: Holds, rework required, quality failures

**Technical Specifications:**

|  |  |
| --- | --- |
| Specification | Value |
| Event type | Discrete events (not continuous time-series) |
| Event frequency | 500-2,000 scans per site per day |
| Scanners per site | 5-15 RFID readers or barcode scanners |
| Communication protocol | HTTP REST API or MQTT |
| Data structure | Event-based with rich metadata (job ID, location, operator, timestamp) |

**Key Capabilities:**

* Real-Time Location: "Where is Job #12345 right now?"
* Job History: Complete audit trail of movements
* Dwell Time Analysis: How long jobs spend at each station (identifies bottlenecks)
* Exception Alerting: Jobs stuck at one location beyond expected time
* Delivery Estimates: Predict completion time based on current location and historical data
* Throughput Metrics: Jobs completed per hour/day by station

**Important Architectural Consideration:**

Job tracking events have fundamentally different characteristics than continuous sensor data. They are discrete, irregular, and event-driven rather than time-series. This impacts architecture choice:

* Options A, B, D: Handle job tracking naturally alongside time-series data
* Option C (SiteWise): SiteWise is designed for continuous time-series, not discrete events. Requires separate Timestream database, increasing complexity.

# 3. Architecture Option A: Flink-Based AWS-Heavy Architecture

## 3.1 Overview and Summary

|  |  |
| --- | --- |
| Characteristic | Details |
| Monthly Cost (30 tenants) | £2,500 - 4,200 |
| Cost Per Tenant | £83 - 140 |
| Complexity Level | Very High (15+ AWS services) |
| Implementation Timeline | 24 weeks |
| Team Size & Skills | 3-4 engineers (Flink, Java/Scala, AWS, data engineering) |
| Best Suited For | Sub-second latency legally required; advanced ML needed |

Option A represents the most technically sophisticated architecture, leveraging Apache Flink for advanced stream processing. This approach provides unparalleled real-time capabilities with sub-second latency but comes with significant operational complexity.

## 3.2 Architecture Diagram

*[INSERT DIAGRAM: SMDH-Option-A-Flink-Architecture.drawio]*

## 3.3 Core Components

### Ingestion Layer

**AWS IoT Core:** MQTT broker receiving 1Hz sensor data from 30-45 machines per site. Handles device authentication via X.509 certificates. Provides device shadows for configuration.

**IoT Rules Engine:** Routes messages based on content. Enriches data with tenant metadata. Filters invalid messages. SQL-based routing rules.

**Kinesis Data Firehose:** Buffers data in 10MB or 60-second batches. Compresses to Parquet format. Invokes Lambda for transformation. Delivers to S3.

**Lambda Transformation:** Validates sensor ranges, converts units, enriches with lookup data. Python or Node.js. Scales automatically.

**S3 Data Lake:** Stores Parquet files partitioned by tenant/date. Lifecycle policies for archival. Enables historical analysis and recovery.

### Stream Processing Layer (Apache Flink)

**Why Flink?**

Apache Flink is an advanced distributed stream processing framework. Unlike simpler alternatives, Flink provides:

* Sub-second latency: Process and respond to events in <1 second
* Exactly-once semantics: Guarantees no data loss or duplication
* Complex event processing: Detect patterns across multiple streams
* Stateful processing: Remember context across events (e.g., calculate 5-minute rolling averages)
* Advanced windowing: Tumbling, sliding, session windows for aggregations
* Backpressure handling: Gracefully manages data spikes

**Flink on EMR:** Runs on 3-5 node cluster. Auto-scaling based on data velocity. Checkpoints state to S3 every 5 minutes.

**RocksDB State Backend:** Embedded database storing intermediate state. Enables fault tolerance. Recovers from checkpoints if nodes fail.

**Flink Jobs:** Custom Java/Scala code for: real-time aggregations, anomaly detection, pattern matching, multi-tenant data isolation.

### Data Platform Layer (Snowflake)

**Snowpipe Auto-Ingestion:** Detects new S3 files via events. Loads into Snowflake within seconds. Serverless—no warehouse management.

**Raw Tables:** VARIANT columns store semi-structured JSON. Partitioned by tenant\_id. Clustered on timestamp for query performance.

**Snowflake Streams:** Change Data Capture tracking new/changed rows. Enables incremental processing. Multiple consumers can read independently.

**Snowflake Tasks:** Scheduled SQL transformations. DAG-based dependencies. Runs hourly aggregations, data quality checks, archival.

**Curated Views:** Row-Level Security enforces tenant isolation. Pre-calculated aggregations. Analytics-ready datasets.

### Application Layer

**API Gateway + Lambda:** REST APIs for data access. Tenant authentication via Cognito. Query Snowflake and return JSON.

**ECS Fargate Web App:** React frontend hosted on Fargate. Auto-scaling. CI/CD via CodePipeline.

**QuickSight Dashboards:** Embedded analytics. Row-Level Security passes through from Snowflake. Pay-per-session pricing.

**SageMaker ML:** Custom model training (predictive maintenance, anomaly detection). GPU-accelerated. Model hosting on auto-scaling endpoints.

## 3.4 Data Flow

Real-time path (critical alerts):

1. Sensor → IoT Core (100ms)
2. IoT Rules → Kinesis Streams (200ms)
3. Flink processes and detects alert condition (500ms)
4. SNS notification sent (200ms)
5. Total: <2 seconds end-to-end

Batch path (historical analytics):

1. Kinesis Firehose buffers 60 seconds
2. Lambda transforms batch
3. S3 receives Parquet files
4. Snowpipe loads into Snowflake (<5 min)
5. Snowflake Tasks aggregate hourly
6. QuickSight dashboards refresh

## 3.5 Strengths

**Sub-Second Latency:** Achieves <1s for critical data paths. Essential if regulatory requirements mandate immediate response.

**Advanced Stream Processing:** Complex event processing, pattern detection, sophisticated windowing not possible with simpler systems.

**Unlimited Flexibility:** Custom Java/Scala code enables any transformation or ML algorithm.

**Proven at Scale:** Flink powers streaming at Netflix, Uber, Alibaba processing billions of events daily.

## 3.6 Limitations

**Very High Complexity:** Managing 15+ services, Flink cluster tuning, distributed debugging.

**Flink Expertise Required:** Rare and expensive skill set. Steep learning curve for Java/Scala.

**Longest Timeline:** 24 weeks to production. Complex distributed systems take time to build and test.

**Operational Overhead:** EMR cluster management, checkpointing monitoring, state recovery testing.

## 3.7 When to Choose

Select Option A ONLY if:

* Sub-second latency is legally or regulatorily mandated (not just preferred)
* Complex event processing is essential for the business case
* Team has Flink expertise or budget for specialists
* Willing to accept highest complexity and 24-week timeline

# 4. Architecture Option B: Snowflake-Leveraged Architecture (RECOMMENDED)

## 4.1 Overview and Summary

|  |  |
| --- | --- |
| Characteristic | Details |
| Monthly Cost (30 tenants) | £2,170 - 3,450 (CHEAPEST) |
| Cost Per Tenant | £72 - 115 |
| Complexity Level | Low (5 core services) |
| Implementation Timeline | 20 weeks (FASTEST) |
| Team Size & Skills | 2-3 engineers (SQL, Snowflake, basic AWS) |
| Best Suited For | Most scenarios - default recommendation |

**🏆 RECOMMENDED OPTION**

Option B centralises data processing in Snowflake, dramatically simplifying the architecture compared to Option A. By leveraging Snowflake's native streaming, task orchestration, and SQL-first approach, this option provides the fastest time-to-market and lowest operational complexity whilst meeting all functional requirements.

## 4.2 Architecture Diagram

*[INSERT DIAGRAM: SMDH-Option-B-Snowflake-Architecture.drawio]*

## 4.3 Core Components

### Simplified Ingestion

**AWS IoT Core:** Same as Option A—MQTT broker with X.509 authentication.

**Kinesis Data Streams:** Lightweight buffering (not Firehose). Streams directly to Snowflake. 10-20 shards partitioned by tenant\_id.

**Snowpipe Streaming:** Snowflake's streaming ingestion service. Loads data in 5-10 seconds. Eliminates S3 staging and Lambda transformation complexity.

### Unified Data Platform (Snowflake Does It All)

**Key Simplification:**

In Option B, Snowflake replaces multiple Option A services:

* Replaces S3 Data Lake → Snowflake internal storage
* Replaces Lambda + Flink → Snowflake Streams and Tasks
* Replaces AWS Glue → Snowflake native schema evolution
* Replaces separate ETL orchestration → Snowflake Task DAGs

**Raw Tables with VARIANT:** JSON data stored in VARIANT columns. Schema-flexible. Automatic compression (75-85% reduction). Partitioned by tenant\_id.

**Snowflake Streams (CDC):** Tracks inserts/updates/deletes. Enables incremental processing. Zero-copy—doesn't duplicate data.

**Snowflake Tasks:** SQL-based transformations run on schedule or when streams have data. DAG orchestration. Serverless compute.

**Dynamic Tables:** Continuously updated aggregations. Declarative SQL definitions. Incremental refresh automatically.

**Row-Level Security (RLS):** Native multi-tenancy. Policies enforce tenant isolation at database level. Cannot be bypassed by applications.

### Machine Learning (Snowflake Cortex)

Snowflake Cortex ML enables SQL-based machine learning:

* CREATE MODEL predict\_failure AS SELECT \* FROM training\_data;
* SELECT PREDICT\_FAILURE(sensor\_data) FROM current\_readings;
* Supports: Classification, regression, time-series forecasting
* Limitation: Less flexible than SageMaker but covers 80% of use cases

## 4.4 Data Flow

End-to-end flow (simplified):

1. Sensor → IoT Core (100ms)
2. Kinesis Streams (200ms)
3. Snowpipe Streaming → Raw table (5-10 seconds)
4. Snowflake Stream detects new rows immediately
5. Task processes (runs every 1 min) → Curated views
6. Dashboard refreshes (total: 60-65 seconds)

**Alert Latency Solution:**

Default 60s latency violates <10s alert requirement. FIX: Add Lambda fast-path for alerts (+£100-150/month, +2 weeks). Fast-path achieves <5s alert latency whilst keeping batch processing in Snowflake.

## 4.5 Strengths

**Lowest Cost:** £2,170-3,450/month. Cheapest option by 15-20%. Auto-suspend when idle.

**Lowest Complexity:** Only 5 core services vs 15+ in Option A. Single platform for most processing.

**Fastest Timeline:** 20 weeks to production. SQL-first development accelerates delivery.

**SQL-First Development:** Familiar to most engineers. Easier to hire than Flink specialists.

**Native Multi-Tenancy:** Row-Level Security enforced at database level. Production-proven.

**Unified Governance:** All data in one platform. Single place for audits, lineage, retention policies.

## 4.6 Limitations

**Alert Latency Requires Fix:** 60s default. Add Lambda fast-path for <5s alerts (+£100-150/month).

**Snowflake Licensing:** Separate from AWS bill. Requires contract negotiation. Some organisations prefer AWS-native only (see Option D).

**Not Sub-Second:** Snowpipe Streaming is 5-10s, not <1s. If sub-second legally required, choose Option A.

## 4.7 When to Choose

Select Option B if:

* Cost and simplicity are priorities (most SME scenarios)
* Team has SQL skills or can train quickly
* 60s dashboard latency acceptable, or willing to add Lambda fast-path
* Want fastest time to market (20 weeks)
* Snowflake licensing not a blocker

**✅ RECOMMENDED for 80% of scenarios**

# 5. Architecture Option C: AWS IoT SiteWise-Based

For complete details on Option C (SiteWise), see smdh-architecture-option-c-sitewise.md. Key summary:

## 5.1 Overview

|  |  |
| --- | --- |
| Monthly Cost | £6,334 (CORRECTED) |
| Complexity | Medium-High (8-10 services) |
| Timeline | 28 weeks |
| Best For | SiteWise expertise; <10 tenants |
| Key Limitation | NOT designed for multi-tenant SaaS |

## 5.2 Architecture Diagram

*[INSERT DIAGRAM: SMDH-Option-C-SiteWise-Architecture.drawio]*

Strengths: Managed asset modelling, OPC-UA/Modbus support, real-time latency

Limitations: Tag-based multi-tenancy risk, dual storage (SiteWise + Timestream), custom dashboards required

# 6. Architecture Option D: AWS-Native (Timestream + Grafana)

For complete details on Option D, see smdh-architecture-option-d-aws-native.md. Key summary:

## 6.1 Overview

|  |  |
| --- | --- |
| Monthly Cost | £3,965 |
| Complexity | Medium (7-8 services) |
| Timeline | 22 weeks |
| Best For | AWS-native mandate |
| Key Strength | Fully AWS-native, Grafana dashboards |

## 6.2 Architecture Diagram

*[INSERT DIAGRAM: SMDH-Option-D-Timestream-Architecture.drawio]*

Strengths: Fully AWS-native, unified storage, native partition multi-tenancy, Grafana

Limitations: 15% more expensive than Option B, custom asset modelling required

# 7. Architecture Comparison

## 7.1 Cost Comparison

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Option | Monthly (30 tenants) | Per Tenant | Annual TCO | Within Budget? | Rank |
| B (Snowflake) | £2,170-3,450 | £72-115 | £346K-521K | ✅ Yes | 1st (Cheapest) |
| A (Flink) | £2,500-4,200 | £83-140 | £525K-720K | ✅ Yes | 2nd |
| D (Timestream) | £3,965 | £132 | £374K-538K | ✅ Yes | 3rd |
| C (SiteWise) | £6,334 | £211 | £566K-715K | ✅ Yes (corrected) | 4th (Most expensive) |

## 7.2 Complexity Comparison

|  |  |  |  |
| --- | --- | --- | --- |
| Option | Number of Services | Primary Skills Required | Complexity Rank |
| B (Snowflake) | 5 services | SQL, Snowflake | 1st (Simplest) |
| D (Timestream) | 7-8 services | AWS-native, SQL, Grafana | 2nd |
| C (SiteWise) | 8-10 services | SiteWise, AWS IoT, React | 3rd |
| A (Flink) | 15+ services | Flink, Java/Scala, AWS | 4th (Most complex) |

## 7.3 Performance and Latency

|  |  |  |  |
| --- | --- | --- | --- |
| Option | Real-Time Latency | Alert Latency | Dashboard Updates |
| A (Flink) | <1 second | <5 seconds ✅ | <5 seconds |
| B (Snowflake) | 5-10 seconds | 60s (fixable to <5s) | 60-65 seconds |
| C (SiteWise) | <1 second | <10 seconds ✅ | <5 seconds |
| D (Timestream) | <5 seconds | <5 seconds ✅ | <10 seconds |

## 7.4 Multi-Tenancy Security

|  |  |  |
| --- | --- | --- |
| Option | Multi-Tenancy Approach | Security Assessment |
| B (Snowflake) | Row-Level Security (RLS) | ✅ Excellent - Database enforced |
| A (Flink) | Snowflake RLS | ✅ Excellent - Database enforced |
| D (Timestream) | Partition Keys | ✅ Good - Native physical isolation |
| C (SiteWise) | Tag-based filtering | ⚠️ Risky - Application-layer |

## 7.5 Decision Framework

Use this decision tree to select the appropriate option:

1. Is AWS-native mandatory (no Snowflake)?

→ YES: Choose Option D (Timestream + Grafana)

→ NO: Continue to question 2

2. Is sub-second latency legally required for ALL data?

→ YES: Choose Option A (Flink)

→ NO: Continue to question 3

3. Is cost and simplicity your top priority?

→ YES: Choose Option B (Snowflake) ✅ RECOMMENDED

→ NO: Continue to question 4

4. Do you have deep SiteWise expertise AND <10 tenants?

→ YES: Consider Option C (with caution)

→ NO: Choose Option B (Snowflake) ✅ RECOMMENDED

DEFAULT: Choose Option B for most scenarios