HIGHTOWER.DIGITAL® · CREATE EXCELLENCE!!™

Technical Owner — Use Case Report

IT REQUIREMENTS USE CASE · TO1 · VERSION R1.A

Author: LW Atkinson (CEO/CTO), HighTower Digital Inc.

Date: 06/25/2026 · **Classification:** HDI Confidential

Methods: Agile Scrum + Lean + XP · **Product:** IT Requirements As-A-Service (MSP SLA #1–5)

File: TO1_UCR1a

4 Technical Owner (Technical Spec Document)

The Technical Owner is the **Technical Leader @ Enterprise** accountable for the architecture, the **non-functional requirements (NFRs)**, and the **Architecture Decision Records (ADRs)**. The Technical Spec is the ISO/IEC/IEEE 29148 **SRS (Software Requirements Specification)** layer: it specifies API schemas, data models, NFR budgets, and edge-case behavior so that, per HighTower.Digital's promise, there is **"zero room for developer interpretation."**

4.1 Validated Design Document

4.1.1 STORY INDEX

Technical enablers and their NFR linkage:

Story	Technical concern	NFR target
TE-01	Idempotent intake API	Reliability 99.9%
TE-02	Spec generation latency	Performance ≤ 2 s p95
TE-03	Real-time SLA stream	Latency ≤ 10 s refresh
TE-04	Horizontal scale	1M records/min ingest

4.1.2 STORY BACKLOG

Technical stories are interleaved with feature stories so architecture keeps pace (XP's *continuous design* / *simple design*).

4.1.3 STORY LIST (FEATURES / FUNCTIONS)

Each technical story carries an **ADR** capturing context, decision, and consequences.

4.2 Frontend Feature List

4.2.1 FEATURE CARD — REAL-TIME SLA STREAM (CLIENT)

FRONTEND · FC-FE-TO-01

Feature: Real-time SLA countdown via WebSocket (React/TS) — meets the ≤10 s refresh NFR for real-time data systems.

(1) Header code

```
import { useEffect, useState } from "react";
```

(2) Function code

```
export function SlaCountdown({ ticketId }: { ticketId: string }) {
  const [remaining, setRemaining] = useState<number>(0);
  useEffect(() => {
    const ws = new WebSocket(`wss://api.hightower.digital/sla/${ticketId}`);
    ws.onmessage = e => setRemaining(JSON.parse(e.data).secondsLeft);
    return () => ws.close();
  }, [ticketId]);
  return <span role="timer">{Math.max(0, remaining)}s to SLA breach</span>;
}
```

(3) Footer code

```
export { SlaCountdown };
```

(4) Use Case Example — A client watches the live countdown; the architect sees the same stream and prioritizes accordingly.

(5) Step-by-Step Example — Open ticket → subscribe to `wss://.../sla/{id}` → render seconds-left → auto-close on unmount.

4.3 Middleware Features List

4.3.1 FEATURE CARD — IDEMPOTENT INTAKE GATEWAY

MIDDLEWARE · FC-MW-TO-01

Feature: Idempotent intake gateway (Node.js) — uses an `Idempotency-Key` header so retries never create duplicates (reliability NFR).

(1) Header code

```
import { Request, Response, NextFunction } from "express";
const seen = new Map<string, string>(); // replace with Redis in prod
```

(2) Function code

```
export function idempotency(req: Request, res: Response, next: NextFunction) {
  const key = req.header("Idempotency-Key");
  if (!key) return res.status(400).json({ error: "Idempotency-Key required" });
  if (seen.has(key)) return res.status(200).json({ ticketId: seen.get(key) });
  res.locals.idempotencyKey = key;
  next();
}
```

(3) Footer code

```
export { idempotency };
```

(4) Use Case Example — A flaky network causes the client to retry; the gateway returns the original ticket ID instead of creating a second one.

(5) Step-by-Step Example — Read key → if seen, return cached result → else proceed and cache on success.

4.4 Backend Feature List

4.4.1 FEATURE CARD — STREAMING INGEST PIPELINE

BACKEND · FC-BE-TO-01

Feature: Streaming ingest pipeline (Python) — async batch ingest meeting the 1M records/min scalability NFR.

(1) Header code

```python
import asyncio
from typing import AsyncIterator
```

(2) Function code

```python
async def ingest(stream: AsyncIterator[dict], batch_size: int = 5000) -> int:
    batch, total = [], 0
    async for record in stream:
        batch.append(record)
        if len(batch) >= batch_size:
            await persist_batch(batch)   # bulk insert
            total += len(batch); batch.clear()
    if batch:
        await persist_batch(batch); total += len(batch)
    return total
```

(3) Footer code

```python
__all__ = ["ingest"]
```

(4) Use Case Example — A real-time data source streams events; the pipeline batches and bulk-persists to sustain throughput without data loss.

(5) Step-by-Step Example — Open stream → accumulate batch → bulk persist at threshold → flush remainder → return total.

4.5 Technical Requirements

4.5.1 BUSINESS CASES

BC-04: *"Idempotency and bulk ingest prevent duplicate-record incidents that previously cost ~6 engineer-hours/month in reconciliation."*

4.5.2 EDGE CASES

- Missing `Idempotency-Key` → 400 (forces correct client behavior).
- WebSocket drops mid-stream → client auto-reconnects with backoff.
- Batch persist partial failure → transactional rollback; record-level dead-letter queue.

4.5.3 USE CASES

UC-04 Guarantee Reliable Throughput: ingest stream → batch → persist transactionally → expose live SLA via WebSocket.

4.5.4 'TECHNICAL REQUIREMENTS' CARD

TECH REQ CARD · TR-TO-01

Title: NFR budget — spec generation latency **Statement:** *95th-percentile spec generation latency shall be ≤ 2 seconds at 1,000 concurrent requests; real-time SLA stream shall refresh ≤ 10 seconds.*
Priority: MUST · **Verification:** Load test (k6) + p95 dashboard

4.5.5 TECH REQUIREMENTS INDEX + PRIORITY

ID	Requirement	Priority	Method
TR-TO-01	Spec latency ≤2 s p95	MUST	XP CI perf gate
TR-TO-02	Idempotent intake API	MUST	Scrum/XP
TR-TO-03	1M records/min ingest	SHOULD	Lean flow
TR-TO-04	Real-time SLA stream ≤10 s	SHOULD	Scrum
TR-TO-05	ADR for every architecture decision	MUST	ADR standard

Appendix D — Glossary (Ubiquitous Language)

Term	Definition
Trouble-Ticket	Canonical term for a submitted requirement/incident entering the SLA pipeline.
Developer-Ready Spec	A specification with API schema, data model, acceptance criteria, and edge cases — zero open interpretation.
SLA Tier	Advisory (4 h), Partner (1 h), Emergency Owner (15 min) response thresholds.
RTM	Requirements Traceability Matrix linking need → story → test.
NFR	Non-Functional Requirement (performance, scalability, reliability, security, etc.).
ADR	Architecture Decision Record — context, decision, consequences.
Bounded Context	DDD boundary within which a domain model and its terms are consistent.
INVEST	Independent, Negotiable, Valuable, Estimable, Small, Testable.
WSJF	Weighted Shortest Job First — Cost of Delay ÷ Job Size.

Appendix E — References

1. ISO/IEC/IEEE 29148:2018 — *Systems and software engineering — Life cycle processes — Requirements engineering.* IEEE SA. https://standards.ieee.org/standard/29148-2018.html
2. ReqView — *ISO/IEC/IEEE 29148 Requirements Specification Templates (StRS/SyRS/SRS/BRS/OpsCon).* https://www.reqview.com/doc/iso-iec-ieee-29148-templates
3. Schwaber, K. & Sutherland, J. — *The 2020 Scrum Guide.* https://scrumguides.org/scrum-guide.html
4. IIBA — *A Guide to the Business Analysis Body of Knowledge (BABOK® Guide v3).* https://www.iiba.org/
5. Blank, S. — *Jobs-To-Be-Done as the Front End of Customer Discovery.* https://steveblank.com/2021/11/04/market-definition-its-the-front-end-of-customer-discovery/
6. Ulwick, T. — *Jobs to Be Done (JTBD): The Original Framework.* Strategyn. https://strategyn.com/jobs-to-be-done/
7. Fitzpatrick, R. — *The Mom Test* (customer discovery interview method).

8. AltexSoft — *Nonfunctional Requirements: Examples, Types and Best Practices.* https://www.altexsoft.com/blog/non-functional-requirements/
9. TestQuality — *Gherkin User Stories & Acceptance Criteria Guide (2026).* https://testquality.com/gherkin-user-stories-acceptance-criteria-guide
10. Boost / PlatinumEdge — *INVEST Criteria for Agile User Stories.* https://www.boost.co.nz/blog/2021/10/invest-criteria
11. Architecture Decision Records. https://adr.github.io/
12. KanbanZone — *Agile Frameworks Compared: Scrum, Kanban, Lean, XP (2024).* https://kanbanzone.com/2024/agile-frameworks-compared/
13. PPM Express — *From RICE to WSJF: Prioritization Techniques.* https://www.ppm.express/blog/13-prioritization-techniques
14. GetProductPeople — *Prioritization Techniques: RICE, MoSCoW, ICE & Kano.* https://www.getproductpeople.com/blog/prioritization-techniques-rice-moscow-ice-kano
15. HighTower.Digital — *MSP SLA #1+5 — IT Requirements As-A-Service.* https://www.hightower.digital/msp-sla-15

Document control. HDI Confidential · Author: LW Atkinson (CEO/CTO) · Version R1.A · 06/25/2026.