HIGHTOWER.DIGITAL® · CREATE EXCELLENCE!!™

Product Owner — Use Case Report

IT REQUIREMENTS USE CASE · PO1 · 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: PO1_UCR1a

3 Product Owner (Product Spec Document)

Per the **2020 Scrum Guide**, the Product Owner is accountable for **maximizing the value** of the product and for **Product Backlog management**: developing and communicating the Product Goal, creating and ordering backlog items, and ensuring the backlog is transparent and understood. The Product Spec turns the VOC backlog into a value-ordered, release-mapped plan. HighTower.Digital fills this role as a **Team Leader @ Dev, Exec, Client** under the Partner/Emergency tiers.

3.1 Validated Design Document

3.1.1 STORY INDEX

The PO maintains a single, ordered Product Backlog. Index excerpt for the reference product:

Rank	Story	Product Goal contribution	WSJF
1	ST-03 Route intake to architect	Enables the SLA promise	9.4
2	ST-04 Generate spec from intake	Core value engine	8.8
3	ST-01 Submit a requirement	Entry point	7.1
4	ST-02 Live SLA countdown	Trust/visibility	4.2
5	ST-05 AI criteria suggestion	Delighter	2.0

3.1.2 STORY BACKLOG

Ordered by WSJF (Appendix C). The PO continuously refines the backlog so the top items are **"Ready"** (small, clear, testable) for the next Sprint Planning.

3.1.3 STORY LIST (FEATURES / FUNCTIONS)

The PO owns the **Definition of Ready** and **Definition of Done**, ensuring each story carries acceptance criteria, NFR links, and a value hypothesis before commitment.

3.2 Frontend Feature List

The PO prioritizes a **Release Roadmap View** so executives can see value delivery against the Product Goal.

3.2.1 FEATURE CARD — ROADMAP / VALUE BOARD

FRONTEND · FC-FE-PO-01

Feature: Release Roadmap & Value Board (React/TS)

(1) Header code

```
import { useRoadmap } from "../hooks/useRoadmap";
interface ReleaseColumn { id: string; goal: string; stories: string[]; }
```

(2) Function code

```
export function ValueBoard() {
  const { columns } = useRoadmap();            // ordered by WSJF
  return (
    <div className="board">
      {columns.map((c: ReleaseColumn) => (
        <section key={c.id} aria-label={c.goal}>
          <h4>{c.goal}</h4>
          <ol>{c.stories.map(s => <li key={s}>{s}</li>)}</ol>
        </section>
      ))}
    </div>
  );
}
```

(3) Footer code

```
export { ValueBoard } from "./ValueBoard";
```

(4) Use Case Example — An executive opens the Value Board to confirm the next release advances the stated Product Goal.

(5) Step-by-Step Example 1. PO orders backlog by WSJF. 2. Roadmap groups stories into release columns by Product Goal. 3. Exec reviews; PO captures feedback as new backlog items.

3.3 Middleware Features List

3.3.1 FEATURE CARD — BACKLOG ORDERING SERVICE

MIDDLEWARE · FC-MW-PO-01

Feature: Backlog Ordering Service (Node.js) — computes WSJF and returns the ordered backlog.

(1) Header code

```
interface BacklogItem { id: string; costOfDelay: number; jobSize: number; }
```

(2) Function code

```
export function orderByWsjf(items: BacklogItem[]): BacklogItem[] {
  return [...items].sort(
    (a, b) => (b.costOfDelay / b.jobSize) - (a.costOfDelay / a.jobSize)
  );
}
```

(3) Footer code

```
export { orderByWsjf };
```

(4) Use Case Example — The Value Board requests the ordered backlog; the service returns items sorted by WSJF.

(5) Step-by-Step Example — Fetch items → compute CoD/JobSize → sort descending → return to frontend.

3.4 Backend Feature List

3.4.1 FEATURE CARD — VALUE HYPOTHESIS TRACKER

BACKEND · FC-BE-PO-01

Feature: Value Hypothesis Tracker (Python) — records the measurable hypothesis behind each story and its validated outcome (Lean Build–Measure–Learn).

(1) Header code

```python
from dataclasses import dataclass
@dataclass
class Hypothesis:
    story_id: str
    metric: str
    target: float
    actual: float | None = None
```

(2) Function code

```python
def is_validated(h: Hypothesis) -> bool:
    return h.actual is not None and h.actual >= h.target
```

(3) Footer code

```python
__all__ = ["Hypothesis", "is_validated"]
```

(4) Use Case Example — After a release, the PO records the actual metric; validated hypotheses justify continued investment.

(5) Step-by-Step Example — Define hypothesis at commit → ship → measure → mark validated/ invalidated → feed next backlog ordering.

3.5 Technical Requirements

3.5.1 BUSINESS CASES

BC-03: *"WSJF ordering increases delivered value per sprint by prioritizing high Cost-of-Delay, low-effort items first."*

3.5.2 EDGE CASES

- Two items with identical WSJF → tie-break by stakeholder priority then FIFO.
- Story with no value hypothesis → blocked from "Ready."
- Job size = 0 → rejected (prevents divide-by-zero in WSJF).

3.5.3 USE CASES

UC-03 Order & Communicate Backlog: PO refines → orders by WSJF → publishes roadmap → captures exec feedback.

3.5.4 'TECHNICAL REQUIREMENTS' CARD

TECH REQ CARD · TR-PO-01

Title: Definition of Ready enforcement **Statement:** *No story shall be eligible for Sprint Planning unless it has acceptance criteria, a value hypothesis, and an estimate ≤ 8 points.* **Priority:** MUST · **Verification:** Backlog policy check + PO sign-off

3.5.5 TECH REQUIREMENTS INDEX + PRIORITY

ID	Requirement	Priority	Method
TR-PO-01	Definition of Ready enforcement	MUST	Scrum
TR-PO-02	WSJF backlog ordering	MUST	SAFe/Lean
TR-PO-03	Value hypothesis per story	SHOULD	Lean
TR-PO-04	Release roadmap view	SHOULD	Scrum
TR-PO-05	Forecast/burnup analytics	COULD	Scrum metrics

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.