Architecture Evaluation Tool - Sprint 0 Deliverable

Team Crackers

November 1, 2025

Title Page

Architecture Evaluation Tool

Sprint 0 Deliverable

Team: Crackers

Team Member	Email	GitHub Username	Role
Roukaya Mabrouk	r.mohammed@innopolis.university	RoukayaZaki	Documentation & Delivery Manager
Timur Kharin	t.kharin@innopolis.university	timur-harin	Front-end Development
Ilya Pechersky	i.pechersky@innopolis.university	IlyaPechersky	Core Logic Development

GitHub Repository: https://github.com/architecture-tools/add-evaluation-tool

Team Members and Contributions

Roukaya Mabrouk

Role: Documentation & Delivery Manager

Contributions:

1. Created interview script
2. Conducted customer interview (together with Ilya)

Timur Kharin

Role: Front-end Development

Contributions:

1. Created repository
2. Created future steps
3. Created deliverable

Ilya Pechersky

Role: Core Logic Development

Contributions:

1. Researched existing solutions
2. Conducted customer interview (together with Roukaya)

Qualitative Analysis

The qualitative analysis table is available in our Sprint 0 report.

Link: https://github.com/architecture-tools/add-evaluation-tool/blob/main/docs/sprints/sprint-0/report.md#qualitative-analysis

GitHub Repository

Project Repository: https://github.com/architecture-tools/add-evaluation-tool

The repository contains project documentation, Sprint 0 findings, customer interview materials, and the comprehensive report with qualitative analysis.

Report on What We've Learned

Overview

During Sprint 0, we researched existing architecture evaluation tools, conducted a customer interview, and defined our MVP vision.

Key Learnings

Market Gap

We identified a gap in the architecture tooling landscape. While methodologies like ADD exist and tools like PlantUML are popular, there's no end-to-end solution combining guided ADD workflows, matrix-based evaluation (NFR × components), and visual diffs between versions.

Technology Choices

- **PlantUML** provides a text-based DSL that works well with Git version control
- **ArchiMate macros** extend PlantUML for enterprise architecture concepts
- Text-based models stored in Git enable DevOps-friendly workflows

Customer Requirements

From the interview, we identified:

1. 12 functional requirements (parsing, matrices, scoring, version management, visual diffs)
2. 2 non-functional requirements (usability)
3. Need for web-based application accessible via browser

MVP Priorities

1. Core evaluation matrix functionality (primary differentiator)
2. PlantUML parsing and component extraction (foundation)
3. Intuitive web interface (accessibility)
4. Version comparison with visual diff (tracking evolution)
5. Scoring calculation and quality tracking (feedback)

Implementation Challenges

Key risks identified:

1. Semantic alignment with ArchiMate and ADD methodologies
2. Consistent abstraction levels in component extraction
3. Model evolution handling in large repositories
4. Structural diff computation

Conclusion

Sprint 0 established a clear understanding of the problem space, identified our value proposition, and defined requirements. We're now prepared to move forward with technical architecture design and proof-of-concept development.