PL/SQL to Emery Hooks Conversion – Requirements Document

none

1. Background

Emery (Groovy-based DSL) was introduced to localize workflow logic, improve XML processing, and enable upgrade safety.

2. Goals

- **Upgrade-safety**
- **Modularity**: Avoid cross-module DB access; ensure isolation.
- **Performance**: Replace XML-heavy PL/SQL logic with faster POJO-based processing.
- **Security**: Limit functionality to avoid unrestricted DB access.
- **Maintainability**: Support hook chaining and config-based customization.

3. Advantages

- **Performance Boost**: Significantly faster execution—large XML files that take 5+ minutes in PL/SQL process in seconds with Emery.
- **Workflow Isolation**: Hooks are strictly scoped to individual workflows or transitions, reducing cross-module interference.
- **Hook Chaining**: Enables clean separation between out-of-the-box (OOTB) logic and custom extensions.
- **Simplified Deployment**: Emery scripts are config-based and do not require recompiling database objects.
- **Upgrade-Safe**: Hook modularity allows smoother upgrade paths with minimal impact on custom logic.
- **Database Agnostic**: Emery maintains flexibility for future platform shifts away from Oracle.
- **Code Isolation**: Hooks are contained within their owning application—no cross-application access to components.

4. Limitations

- **No** `Operator support e.g. LIKE` **/Procedure Calls**: Certain SQL ops and DB calls not supported.
- **Minimal IDE Support**: Lacks debugging, autocomplete, or validation.
- **Case Sensitivity**: Errors due to variable case mismatches.
- **Learning Curve**: Requires Groovy/Emery knowledge.

5. Key Concerns

- **High Effort**: Manual conversion is time intensive.
- **Partial Migration Risk**: Dual logic (PL/SQL + Emery) may add upgrade/test complexity.
- **Tooling Gap**: Syntax validation, test harnesses, and editor enhancements needed.
- **ROI Doubt**: Need to validate value vs. effort.

6. Use Cases

1. **Generic Conversion**: Convert PL/SQL file → Emery hook.
    1. Convert a PL/SQL file/proc into an equivalent Emery hook.
    2. Useful for bulk conversions or prototyping.
    3. Initial tool versions may support limited syntax and flag unsupported parts for manual review.
2. **Context-Aware Conversion**:
    - Users can:
        - Select: `Module → Workflows → Workflow` to define a new hook.
        - Select: `Module → Workflows → Workflow → Transition / Submit Action / Custom Function` for targeted hook editing.
        - Fetch: From instance file system, load `config.json` of the selected workflow and list all configured hooks (PL/SQL and Emery) in order.
        - Choose: A specific PL/SQL hook for conversion or an existing Emery hook for improvement or correction.
    - Conversion scope is limited to generating Emery hooks only.
3. **Editor Support**:
    - Provide a modern code editor with:
        - Autocomplete for Emery DSL
        - Syntax highlighting and validation (including case-sensitivity, bracket matching, etc.)
        - Inline unit testing support
        - Log tracing for debugging
    - Note: This functionality supports Emery hook creation and editing only (no PL/SQL generation).

---

7. Action Items

1 1 incomplete : Identify unconverted PL/SQL and reasons by talking to other product members. 2 2 incomplete : Fetch exceptions from LCNC documentation.

---

8. Final Decision

The decision is to proceed with converting PL/SQL to Emery Hooks.

> Note: The Admin UX will support generation of Emery Hooks only. PL/SQL authoring will not be allowed through this interface.

Decision from: ,

Reference

1. Emery - Team Platform - Confluence
2. Domain Specific Language (Emery) - Cookbook
3. DSL Based Groovy Hooks (Emery) - Team Platform - Confluence
4. Emery Client APIs - Team Platform - Confluence
5. Emery- Assignee, Parameter and Rule configurations - Team Platform - Confluence
6. Workflow - Emery conversion - Team Platform - Confluence
7. Emery FAQ - Team Platform - Confluence
8. Emery Overview - Technical Publications Team - Confluence
9. Reference from Diff Tool: https://metricstream.atlassian.net/wiki/spaces/AR/pages/57790553/Upgrade+Tooling+-+Steps+to+follow+in+the+projects#:~:text=b)-,PLSQL%20to%20Emery%20Code%20Assistant,-%3A