

Full-Stack Cost Plans System Flow

1) Planning cycle setup (Year is the boss)

1. Admin creates/opens a **Planning Year** (example: 2026).
2. System marks it as **Draft / Active / Closed**.
3. If closed, system locks editing for that year (audit-safe).

2) Project onboarding into a planning year

1. Projects already exist in **projects (WBSE master)**.
2. For a selected year, create **project_metadata (project-year context)**:
 - a. Division, Branch, Section/Country
 - b. Trust Fund, Grant, Earmarking, etc.
3. Assign finance ownership using **user_project_access** (My Projects).

3) Auto-initialize the project cost plan (templates)

1. Finance clicks: “**Initialize Cost Plan for Project-Year**”
2. System auto-creates baseline rows:
 - a. OPC items from `opc_templates`
 - b. Deployment scaffolding (by grade/category or by position)
 - c. Optional: activity scaffolding
3. Everything starts consistent, not “Jane’s Excel version 4.”

4) Reference + Parameters selection

1. Select or attach the correct **salary_parameter_set** for:
 - a. Year + duty station/country (and later: version)
2. Set **project_parameters**:
 - a. Inflation
 - b. PSC rate (Umoja 155)
 - c. Defaults like currency assumptions if needed

5) Staffing planning (position-first, not vibes-first)

1. System maintains allowed positions via **project_positions**.
2. Finance builds a **staffing_plan** per project-year:
 - a. Position ID, category, grade, duty station
 - b. Funding dates or coverage period
3. Later (integration): system syncs position details from **Umoja** into the position layer.

6) Deployment allocation (time dimension)

1. Finance sets allocations via:
 - a. Monthly allocations, or
 - b. Date range + percent rules (system expands into months)
2. System calculates **post-months** per position or per grade bucket.
3. Deployments can be split across projects where needed.

7) Non-staff costs entry (OPC)

1. Finance edits **opc_items**:
 - a. Quantity, unit cost, duration

- b. Umoja class, commitment item, object code
- 2. Allocate across quarters (Q1–Q4) or let system derive.
- 3. Items can be toggled for scenarios (what-if).

8) Activities and results framework (optional but supported)

- 1. Define **activities** (project-year).
- 2. Add **activity_costs**:
 - a. Quarterly allocations
 - b. Classifications for reporting
- 3. Enables reporting by “what we’re doing”, not just “what we’re spending”.

9) Scenario planning (baseline vs what-if)

- 1. Baseline scenario always exists.
- 2. Finance creates new scenarios:
 - a. “No Petrol”
 - b. “Cut travel by 30%”
 - c. “Shift budget to staffing”
- 3. Scenario changes stored as **scenario_overrides** (not destructive edits).
- 4. System can compare scenarios side-by-side.

10) Versioning and approvals (immutability)

- 1. Finance works in **Draft**.
- 2. When ready: create a **Version**:
 - a. Submitted
 - b. Approved
 - c. Released snapshot
- 3. Approved versions become immutable.
- 4. Audit trail stays clean and boring (which is the goal).

11) Recalculation engine (real logic, not spreadsheet praying)

- 1. Finance triggers **Recompute** (or system runs automatically on save later).
- 2. System runs deterministic calculations:
 - a. staff_costing
 - b. psc_calculations
 - c. summary_lines
 - d. rollup_totals
- 3. Every run is logged in **calculation_runs** with status + timestamp.

12) Reporting and Dashboards (multi-level + What-if Scenarios)

12.1 Year-first reporting (mandatory filter)

- 1. User selects **Planning Year** (ex: 2026).
- 2. System loads reporting context for that year only:
 - a. Baseline + available scenarios
 - b. Latest approved version (or draft if allowed)

12.2 Multi-level rollups (same numbers, different lenses)

Users can view totals and breakdowns at:

- **Project (WBSE)**

- **Section/Country**
- **Branch**
- **Division**
- **Organization-wide**

And slice by:

- **Umoja Class** (010, 155, etc.)
- **Category / Subcategory**
- **Commitment item / IMIS object code**
- **Cost type (staff vs non-staff)**
- **Quarter (Q1-Q4) and Annual**

12.3 Baseline vs Scenario comparison (core decision-support)

1. Baseline always exists (the official plan).
2. Users can select **1 or more scenarios** and compare:
 - a. Baseline vs Scenario A
 - b. Baseline vs Scenario A vs Scenario B
3. System shows deltas:
 - a. **Absolute difference (CHF/USD)**
 - b. **% change**
 - c. **Which category changed most**
 - d. **Which org level benefits most (division/branch/country)**

12.4 What-if scenario types supported

Finance users can create scenarios like:

A) Cost item toggles (simple switches)

- Exclude or reduce specific OPC items (ex: **petrol, travel, IT equipment**)
- Apply at:
 - project level
 - country level
 - division level

B) Category-level adjustments (bulk policy changes)

- “Cut all travel by 20% across Division X”
- “Reduce operating costs by 10% in Country Y”
- “Freeze recruitment for 6 months” (staffing impact)

C) Reallocation scenarios (trade-off planning)

- “Remove petrol, redirect savings to 2 new posts”
- “Shift budget from consultants to staffing”
- “Move activity funding from Q4 to Q2”

D) Parameter-based what-if

- Alternate inflation assumptions
- Different PSC rates (if policy changes)

- Alternate exchange rate assumptions (if included)

12.5 Scenario drill-down and explainability

For any scenario delta, the system must let the user drill into:

- **Which override caused it**
- The affected records (OPC items / staffing / deployment / activities)
- The override reason/comment (mandatory for governance)

So the dashboard isn't just "numbers changed", it's "numbers changed because *this* changed."

12.6 YoY comparison with scenarios (the "savings proof" view)

Users can compare:

- **2025 baseline vs 2026 baseline**
- **2026 baseline vs 2026 scenario (expected savings)**
- **2025 baseline vs 2026 scenario (policy impact across years)**

This enables reporting like:

- "We reduced petrol spend by X from last year"
- "Scenario A saves Y across Division 1"
- "Category Z increased despite overall savings"

12.7 Outputs and integration

Reporting must support:

- Dashboard tables + charts (web UI)
- Export:
 - CSV / Excel
 - Power BI-friendly dataset
- Scheduled extracts (later phase)

13) Security & access (so nobody edits random projects)

1. Finance users see **My Projects** by default.
2. Reviewers get read-only or approval permissions.
3. Admins manage years, templates, reference data, roles.

14) Audit + governance (so this survives UN reality)

1. Every change is logged in **audit_log**.
2. Imports logged in **data_import_jobs**.
3. Calculation runs logged in **calculation_runs**.
4. You can always answer: *who changed what, when, and why*.