Migration Strategy: AccuFund to Custom Fund Accounting System

Migrating from an established system like AccuFund to a custom-built solution requires careful planning. This document outlines a comprehensive approach to ensure a successful transition.

1. Data Migration Strategy

Assessment and Extraction:

- Export all financial data from AccuFund (charts of accounts, funds, transactions, budgets)
- Extract vendor/customer information, recurring transactions, and historical records
- Document custom reports and specialized configurations

Data Transformation:

- Map AccuFund chart of accounts structure to new system
- Transform fund definitions and attributes
- Convert historical journal entries while maintaining audit trails
- Adapt budget structures to the new database schema

Data Validation:

- Run parallel financial reports in both systems to verify accuracy
- Implement cross-checking reconciliation processes
- Verify opening balances match closing balances from AccuFund

2. Technical Implementation

Database Setup:

- Implement the PostgreSQL schema we designed
- Create import utilities for AccuFund data
- Build data verification and integrity checks

Integration Requirements:

- Identify any third-party systems AccuFund interfaces with (banking, payroll, etc.)
- Develop equivalent integration points in the new system
- Test data flow between systems

2.5 Inter-Entity Transfer Configuration (v8.8+)

The **Inter-Entity Transfer** feature, introduced in version 8.8 of the Fund Accounting System, automates cash movements between related legal entities by creating paired *Due To*/*Due From* journal entries. When migrating from AccuFund you must ensure the underlying account structure and historical transactions correctly map to this new capability.

Why It Matters for Multi-Entity Organizations

- Eliminates manual dual-entry work between entities.
- Provides an auditable `matching_transaction_id` that links both sides of the transfer.
- Keeps inter-company balances in sync via *Due To*/*Due From* accounts.

Required *Due To / Due From* Account Structure

- **Assets 19xx – "Due From <Other Entity>"**
- **Liabilities 29xx – "Due To <Other Entity>"**
- Create one pair per counter-party entity to simplify reconciliation.

Mapping AccuFund Accounts to the New System

- Locate AccuFund GL accounts used for inter-company settlements.
- Map each to the correct 19xx / 29xx account code in the new chart of accounts (or create new accounts if missing).

- Confirm each account is assigned to the correct entity in the import CSV.

Migration Considerations for Existing Inter-Entity Transactions

- When importing historical journal entries, populate `is_inter_entity`, `target_entity_id` and `matching_transaction_id` so legacy transfers appear as linked pairs.
- If AccuFund stored transfers as single multi-line entries, split them into two separate entries (one per entity) during transformation.

Validation Steps Post-Migration

- Run the "Inter-Entity Transfers" report and verify all historic transfers appear in matched pairs.
- Confirm each entity's *Due To* balance equals the counter-party's *Due From* balance.
- Perform a sample transfer in the new system to ensure automatic dual-entry posting functions as expected.

2.6 Bank Account Integration (v8.7+)

The **Bank Account Management** module—added in version 8.7 and enhanced in v8.8—allows direct tracking and optional online connections to operating, savings, and credit-card accounts. When migrating from AccuFund you should map every active AccuFund bank account to an equivalent record in the new system to preserve beginning balances, reconciliation history, and account metadata.

Identifying Bank Accounts in AccuFund

- Run the *Bank Accounts Listing* report (Administration → Banking).
- Export the report to CSV, capturing: Bank Name, Account Name, GL Account Code, Routing Number, Last Reconciled Date, and Current Balance.
- Note any accounts marked *Inactive*—these can be imported with `status = "Inactive"` so history is preserved but the account is hidden from daily workflows.

Mapping Bank Account Data to the New System

1. **Prepare CSV** – Create `bank_accounts.csv` with the following columns: `bank_name,account_name,account_number,routing_number,type,status,balance,connection_method,description`
2. **Type Mapping** – AccuFund "Checking" → "Checking", "Savings" → "Savings", "Credit Card" → "Credit Card".
3. **Connection Method** – Use `Manual` for all imports; you can switch to `Plaid` or `OFX`

later once online credentials are verified.
4. **Opening Balance** – Populate the `balance` field with the CSV export's *Current Balance* as of the migration cut-over date.
5. **Import** – Upload the CSV via *Settings* → *Data Import* → *Bank Accounts* and validate.

Best Practices for Setting Up Bank Accounts

- **Clear Naming** – Include the entity or fund in the *Account Name* (e.g., "TPF – Operating Checking").
- **One-to-One Mapping** – Ensure each GL cash account points to exactly one bank-account record to simplify reconciliation.
- **Reconcile Immediately** – After import, run the Reconciliation Wizard for the last closed period to confirm the opening balance.
- **Enable Online Sync Later** – Wait until after cut-over and internal verification before turning on Plaid/OFX to avoid duplicate transactions.

3. Process and Training

Process Documentation:

- Document current AccuFund workflows and identify changes in the new system
- Create standard operating procedures for core accounting functions
- Update approval workflows and internal controls

User Training:

- Develop role-based training materials
- Conduct hands-on training sessions for finance staff
- Provide specialized training for administrators and report developers

4. Phased Implementation Approach

Phase 1: Core Setup (2-3 months)

- Install infrastructure
- Configure chart of accounts and funds
- Migrate master data (vendor/customer records)
- Set up user accounts and permissions

Phase 2: Historical Data (1-2 months)

- Import historical transactions
- Validate financial reports against AccuFund
- Test period closings and reconciliations

Phase 3: Parallel Operations (1-2 months)

- Run both systems simultaneously
- Process transactions in both systems
- Compare outputs and resolve discrepancies

Phase 4: Cutover (1 week)

- Finalize balance transfers
- Complete final verifications
- Switch to the new system as primary

5. Common Challenges and Mitigation

Data Complexity:

- AccuFund likely contains years of historical transactions and customizations
- Solution: Detailed data mapping and validation processes

Feature Gaps:

- The custom solution may not immediately replicate all AccuFund features
- Solution: Prioritize critical functionality; phase additional features

User Resistance:

- Staff comfortable with AccuFund may resist change
- Solution: Early stakeholder involvement and comprehensive training

Reporting Differences:

- Custom reports in AccuFund need to be recreated
- Solution: Identify critical reports early and prioritize their development

6. Cost Considerations

- Development costs for migration utilities
- Staff time for data validation and testing
- Potential consulting assistance for complex data mapping
- Training and documentation development

A successful migration would typically take 4-6 months from planning to full implementation, depending on the complexity of your AccuFund implementation and the amount of historical data to be migrated.

Generated by Droid by FACTORY on