Process Management System - Complete Technical Brief

SYSTEM PHILOSOPHY

Fundamental Concept

A process is simultaneously the final result and all the actions necessary to achieve it. Every business activity is viewed as a work cycle with beginning, sequence of actions, and end.

Three Entities Theory

Everything in the system is an **ENTITY**, classifiable as:

- **RAW MATERIAL**: Elementary entity (basic process)
- **SEMI-FINISHED**: Composite entity (intermediate process)
- **FINISHED PRODUCT**: Complex entity (final process)

Master Process

The root process of the entire system is **"Company Management"** - from which all other processes derive through specialization and composition.

ADVANCED LOGICAL STRUCTURE

Table-Process-Instance-Task Model

TABLE = Process Name (e.g., "Customer Registration")
INSTANCE = Specific Process (e.g., Customer "Mario Rossi" - CLI25001)
TASK = Activity Recording (who, when, how created the instance)

Architectural Principles

1. **Each table represents ONE process** with its interface
2. **Each instance is a materialization** of that process
3. **Each instance is born with integrated TASK attributes** (responsible, date, status, time)
4. **Processes can be composite** through relationships between tables

Process Hierarchy

- **MAIN PROCESS**: General category (e.g., TASK, REQUEST, PROJECT)
- **PROCESS**: Macro-activity (e.g., CRM, SHOWCASE, PRODUCTION)
- **SUB-PROCESS**: Specific activity (e.g., PHONE, EMAIL, MEETING)

- **INSTANCE**: Specific materialization of the process
- **TASK**: Activity recording attributes

CODING SYSTEM AND RELATIONSHIPS

Coding Standards

- **Base Format**: [3 LETTERS][YEAR][PROGRESSIVE NUMBER]
- **Examples**: PRJ25001, RCH25711, TSK2382
- **The 3 letters identify the process type**
- **2-digit year + progressive number for unique instance**

Hierarchical Coding System (Process DNA)

Automatic Path to track parent-child relationships:

- **Format**: [PARENT_PROCESS][CHILD_PROCESS]
- **Examples**:
  - PRJ25001\RCH25045 (Request 1 of Project 25001)
  - PRJ25001\RCH25046 (Request 2 of Project 25001)
  - PRJ25001\RCH25045\OFR25123 (Offer of Request 1)

Process DNA Advantages

- **Immediate traceability**: Genealogy is in the code itself
- **Powerful search**: PRJ25001* finds all children of the project
- **Logical navigation**: Path reconstructs hierarchical structure
- **Automatic validation**: Integrity control of relationships via code

Automatic Child Process Generation

A process can automatically generate other processes

- TASK "Prepare offer" → automatically generates:
  - TASK "Create bill of materials"
  - TASK "Prepare technical sheet"
  - TASK "Send document to customer"

Relationships and Dependencies

- **Containment**: Project → Requests → Activities → Sub-activities
- **Sequential dependency**: Process A complete → Process B starts
- **Logical dependency**: Approvals, resources, time constraints
- **Cross traceability**: Different types of processes that influence each other

LIFE CYCLE MANAGEMENT

Process States

- **Standard**: To do, In progress, Done
- **Type-specific**: To ship, Shipped, To correct, Under approval

Operational Flexibility

The system supports **complete flexibility** compared to the ideal plan:

- ✅ **Skip planned TASKs**
- ✅ **Add unplanned TASKs**
- ✅ **Modify execution order**
- ✅ **Create TASKs "on the fly" during execution**

Creation Modes

1. **Planned**: Creation of process instances with predefined template
2. **Dynamic**: TASK creation during execution (in progress)
3. **Hybrid**: Combination of planning and dynamic adaptation

DATA ARCHITECTURE

Separate TASK Architecture (Recommended)

Each process instance is connected to a dedicated TASK through 1:1 relationship

Process Tables (e.g., PROJECTS, REQUESTS, etc.)

Code: PRJ25001
Name: "Cosmetics SpA Project"
Client: "XYZ Company"
[Process-specific attributes...]

Centralized TASK Table

Task_ID: TSK2382
Process_Code: PRJ25001
Process_DNA: PRJ25001\RCH25045\OFR25091
Process_Type: "PROJECT"
Responsible: "Mario Rossi"
Start_Date: 27/05/25 17:01
End_Date: [empty if in progress]
Status: "In progress"
Time_Spent: 120 min
Notes: "Complex project, requires further investigation"

Separated Architecture Advantages

- **Scalability**: Facilitated aggregate queries on TASKs

- **Evolution**: New TASK attributes without modifying all tables
- **Reporting**: Centralized analytics on times and performance
- **Process DNA**: Optimal management of hierarchical code
- **History**: Centralized change tracking

Detailed Relationships and Dependencies Structure

A) Structural Relationships (Containment)

Define parent-child hierarchy of processes:

PROJECT contains REQUESTS
REQUEST contains ACTIVITIES
ACTIVITY contains SUB-ACTIVITIES

DNA Coding: PRJ25001 → PRJ25001\RCH25045 → PRJ25001\RCH25045\OFR25091

B) Temporal Relationships (Sequence)

Define mandatory execution order:

TASK A complete → TASK B can start
TASK B complete → TASK C can start

Example: "Prepare offer" → "Send to customer" → "Follow up feedback"

C) Logical Relationships (Conditional Dependencies)

Define prerequisites for advancement:

APPROVAL → Unlocks other processes
RESOURCE_AVAILABLE → Enables dependent processes
DOCUMENT_COMPLETE → Allows next phase

Example: "Contract signed" → Enables "Start production"

D) Cross Relationships (Cross-Process)

Connect different types of processes that influence each other:

SUPPLIER_ORDER depends on PROJECT_APPROVAL
CUSTOMER_INVOICE depends on PRODUCT_SHIPMENT

Example: ORD25067 awaits approval from PRJ25001\APP25123

Data Structure for Relationships

PROCESS_RELATIONSHIPS Table

Relationship_ID: REL001
Parent_Process: PRJ25001
Child_Process: PRJ25001\RCH25045
Relationship_Type: "CONTAINMENT"
Sequence_Order: 1

Relationship_Status: "ACTIVE"
Creation_Date: 27/05/25

DEPENDENCIES Table

Dependency_ID: DEP001
Blocking_Process: PRJ25001\APP25123
Blocked_Process: ORD25067
Dependency_Type: "APPROVAL"
Unlock_Condition: "Status = APPROVED"
Dependency_Status: "WAITING"
Priority: "HIGH"

Automatic Dependency Management

- **Integrity control**: Verify prerequisites before starting processes
- **Automatic notifications**: Alert when dependencies are satisfied
- **Cascade update**: Propagate status changes along the chain
- **DNA validation**: Check consistency of hierarchical code

ADVANCED FILTERS AND SEARCH SYSTEM

Google Sheets-Style Multiple Filters

Implementation already tested in TASK/PROJECTS/REQUESTS system:

- Filters for each column (User, Date, Client, Process, Status)
- Combined and simultaneous filters
- Autocompletion based on existing data
- Quick reset and user configuration saving

Requirements for All Process Lists

- **Universal filters**: Same system for every table/list
- **Performance**: Instant results even on large volumes
- **Personalization**: Saving favorite filters per user
- **Hierarchical searches**: DNA pattern support (PRJ25001*)

Specific Required Searches

- All processes of a specific project
- Activities per responsible and period
- Processes blocked by dependencies
- Progress states per category

INTEGRATED DAILY REPORTS SYSTEM

TASK-Attendance-Reports Integration

Objective: Unified system integrating processes, attendance, and daily reporting

System Components:

- **QR Attendance System**: Entry/exit with QR code on iPad (existing)
- **TASK Recording**: Each process generates a TASK with automatic timing
- **Automatic Report**: Daily summary for each collaborator

Operational Flow:

1. Collaborator enters → Scans QR → System records attendance
2. Works on processes → Each action generates TASK with timing
3. End of day → System generates automatic activity report
4. Management → Aggregated view of workload per person/project

Report Functions

- **Daily View**: Activity summary for each user
- **Automatic Timesheet**: Time dedicated per process/project
- **Workload Analysis**: Dashboard for management
- **Attendance Integration**: Correlation between hours/actual activities

Fundamental UI/UX Principle

"Simplicity is the maximum complexity" - The system must be technically powerful but extremely simple to use.

Simplicity Examples from Current Implementation:

- Intuitive Google Sheets-style filters (see PROJECT MANAGER)
- Clean interfaces focused on essential data
- Breadcrumb navigation for user orientation
- Hidden process codes (only internal DNA system)

User Navigation and Tracking

Breadcrumb Navigation to track user path:

- Structure: Main_Process > Sub_Process > Current_Operation
- Dedicated table to store session paths
- Advantages: Workflow debugging, usage pattern analysis, intuitive navigation

USE CASE SCENARIOS

Scenario 1: Employee Hiring

Master Process: "Human Resources Management"
↓
Instance: COL25001 "Mario Rossi Hiring"
↓
Automatically generated TASKs:
- Employee record registration
- Document collection (ID, diplomas, etc.)
- Contract preparation
- Contract signing
- Payroll system insertion

Scenario 2: Customer Project with Process DNA

Master Process: "Order Management"
↓
Instance: PRJ25001 "Cosmetics SpA Project"
↓
Contains (with DNA):
- PRJ25001\RCH25045 (Quote Request)
- PRJ25001\RCH25046 (Order Request)
↓
RCH25045 generates:
- PRJ25001\RCH25045\DTB25089 (Bill of materials)
- PRJ25001\RCH25045\STC25090 (Technical sheet)
- PRJ25001\RCH25045\OFR25091 (Quote document)

Scenario 3: Cross-Process Dependency Management

Situation: Production blocked waiting for project approval

Main Process: PROD25089 "Face Cream Batch Production"
Dependency: PRJ25001\APP25123 "Cosmetics Project Approval"

DEPENDENCIES Table:
- Dependency_ID: DEP089
- Blocking_Process: PRJ25001\APP25123
- Blocked_Process: PROD25089
- Condition: "Status = APPROVED"
- Status: "WAITING"

Automatic Workflow:
1. System checks PRJ25001\APP25123 status
2. If "APPROVED" → Notify PROD25089 responsible
3. Automatically unlock production
4. Update dependency status → "RESOLVED"

Search Functions with Process DNA

- PRJ25001* → All processes of project 25001
- *\RCH* → All requests of all projects

- PRJ25001\RCH25045* → Everything derived from specific request
- *\APP* → All approvals in all projects
- Dependency queries: Find all processes blocked by a specific process

ASPECTS TO DEVELOP IN SECOND PHASE

Performance and Scalability (High Priority)

Issue: Managing tens of thousands of annual processes with 5-10 active users

Strategies to Define with Technicians:

- **Intelligent archiving**: When and how to archive completed processes
- **Volume management**: Purging logic based on importance and time
- **Query optimization**: Performance on complex hierarchical searches
- **DNA indexing**: Data structures for efficient wildcard searches

Process Flexibility Categorization

To develop after complete technical understanding:

Framework to Define:

- **CRITICAL Processes**: Non-modifiable, require approvals
- **STANDARD Processes**: Modifiable with motivation and tracking
- **FLEXIBLE Processes**: Completely adaptable in real-time
- **Decision Matrix**: Who can modify what and when

Governance Rules:

- Approval workflow for exceptions
- Complete log of all process modifications
- Automatic notifications for dependency impacts
- Intelligent rollback for corrections

TECHNICAL OBJECTIVES

Core Functions

1. **Intelligent coding** that maintains hierarchical relationships
2. **Advanced filter system** Google Sheets-style for all lists
3. **Automatic daily reports** integrated with QR attendance system
4. **Ultra-simple UI** that hides technical complexity from users
5. **Breadcrumb navigation** to track user paths in processes
6. **Automatic generation** of child processes according to templates
7. **Controlled flexibility** in variant management
8. **Complete traceability** of all dependencies

Technical Constraints

- System based on existing FileMaker
- Maintain human readability of codes
- Support planning and dynamic execution
- Handle exceptions without compromising integrity
- Optimal performance on complex hierarchical structures

IMPLEMENTATION QUESTIONS

Immediate Implementation

1. **Filter architecture**: How to replicate Google Sheets system on all tables?
2. **Attendance integration**: API/connection between QR system and FileMaker TASKs?
3. **DNA performance**: Optimal indexing for hierarchical searches?
4. **Breadcrumb storage**: Data structure for user path tracking?
5. **Responsive UI**: Interface adaptation for ease of use?

Development Phase

6. **Archiving strategy**: Automatic criteria for large volume management?
7. **Flexibility governance**: Framework for process categorization?
8. **System integration**: Connections with other business tools?
9. **Backup/Recovery**: Strategies for critical data protection?
10. **User scalability**: Preparation for team growth?

EXPECTED RESULT

A system that implements the **"Everything is a Process"** philosophy with gradual approach:

Phase 1 - Operational Base (Immediate)

- **Define** processes with limited hierarchical DNA (max 4-5 levels)
- **Implement** advanced filter system on all lists
- **Integrate** separate TASKs with automatic daily reports
- **Connect** QR attendance system with process tracking
- **Create** ultra-simple interfaces that hide complexity

Phase 2 - Optimization and Governance (Subsequent)

- **Optimize** performance for large data volumes
- **Define** process flexibility categorization
- **Implement** intelligent archiving
- **Refine** automation and advanced dependencies

Immediate Added Value

- **Complete traceability**: Every business action becomes traceable
- **Automatic reports**: End of manual timesheet management
- **Powerful filters**: Instant search on any criteria
- **Ease of use**: Intuitive interfaces for all users
- **Attendance integration**: Single system for time and activities

The system transforms business management from **"management of scattered activities"** to **"orchestration of intelligent and traceable processes"**, always maintaining simplicity as the guiding principle.

This document represents the philosophical and logical foundation for implementing the integrated process management system.

Luca Meggiolaro

Kool Tool