DOCUMENT 1: 3P3 FOUNDATIONS AND TABLE ANALYSIS

Technical Compass for FileMaker Implementation

For: Cyril Amegah - Technical Assessment THE BRIDGE
Date: August 10, 2025
Purpose: Practical guide to understand and implement 3P3 in FileMaker

---

PART A: 3P3 COMPASS - THE FUNDAMENTAL PRINCIPLE

🧭 THE HEART OF EVERYTHING: CONTROLLING THE ENTITY

3P3 Cardinal Principle:

> "3P3 is extremely simple - we want to CONTROL THE ENTITY. The entity is the product of ACTIONS and RESOURCES. From this ontological principle emanates all the philosophy."

In simple words for technicians:

- **ENTITY** = Anything we do in the company (client, product, task, order, etc.)
- **ACTIONS** = The movements that transform the entity (call client, assemble strands, send email)
- **RESOURCES** = Who/What is needed to perform actions (Marco, glue gun, die-cutting machine, time)

🔍 CONCRETE EXAMPLE - Understanding in 30 Seconds

ENTITY: 4-Panel Color Chart for LILA COSMETIC Client

ACTIONS that create it:

- TSK25001: Phone call to client
- PRJ25001: Project opening
- RCH25001: Sample strand request
- TEH25001: Technical sheet creation
- APR25001: Final approval

RESOURCES needed:

- Simona (commercial for phone call)
- Marco (operator for assembly)
- Dyeing machine (to color spools)
- Glue gun (for assembly)

RESULT: Finished chart worth 76€ for the client

⚡ WHY THIS IS REVOLUTIONARY

Traditional System (classic ERP):

- CLIENTS table separate from PRODUCTS table
- ORDERS table separate from ACTIVITIES table
- Business Intelligence to "put the pieces back together"
- **Result:** Increasing complexity, fragmented information

3P3 System:

- **ONE ENTITY** contains everything: what it is, how it's made, who makes it
- **HIERARCHICAL DNA** automatically connects everything
- **UNIFIED PROCESS** from first click to finished product
- **Result:** Natural simplicity, complete vision

PART B: RESOLVING SPECIFIC ARCHITECTURAL DOUBTS

🗃️ PROBLEM 1: LOG Table Schema - Eliminating Confusion

YOUR DOUBT (from emails):

> "Confusion between TaskID vs ProcessID - which one to use?"

DEFINITIVE SOLUTION:

CORRECT SCHEMA - Centralized LOG Table:

```
sql

LOG TABLE (Common existential attributes):
├── ProcessID (Primary Key) - DNA format: TSK25001, RCH25045, etc.
├── ProcessType (TSK, PRJ, RCH, TEH, APR, etc.)
├── ClientID (Foreign Key → CLIENTS table)
├── ResponsibleID (Foreign Key → USERS table)
├── CreatorID (Foreign Key → USERS table)
├── Status (TO_DO → IN_PROGRESS → DONE)
├── DateCreated, DateStarted, DateCompleted
├── ParentProcessID (for hierarchical DNA)
├── Description (free text)
└── Priority (High/Medium/Low)
```

SPECIFIC TABLES (1:1 relationship with LOG):

```
sql
```

```
TSK_PHONE_CALL TABLE:
├── ProcessID (FK → LOG.ProcessID)
├── PhoneNumber
├── CallDuration
├── CallOutcome
└── (only specific fields for phone calls)

TSK_EMAIL TABLE:
├── ProcessID (FK → LOG.ProcessID)
├── EmailAddress
├── Subject
├── Attachments
└── (only specific fields for emails)

RCH_SAMPLES TABLE:
├── ProcessID (FK → LOG.ProcessID)
├── SampleType
├── Quantity
├── Specifications
└── (only specific fields for sample requests)
```

SIMPLE RULE:

- **TaskID = COMPLETELY ELIMINATED** ❌
- **ProcessID = UNIQUE identifier** ✅
- **LOG = Common attributes** (dates, states, responsible parties)
- **Specific tables = Unique attributes** (phone number, email, etc.)

🔧 PROBLEM 2: Managing Common vs Specific Attributes

GUIDING PRINCIPLE:

> "If a field is useful for ALL processes → goes in LOG table"
>
> "If a field is useful only for ONE type of process → goes in specific table"

PRACTICAL EXAMPLES:

COMMON ATTRIBUTES (go in LOG):

- **Status:** All processes have states (TO_DO → IN_PROGRESS → DONE)
- **Responsible:** All processes have a responsible party
- **Date:** All processes have start/end dates
- **Client:** Most processes are connected to a client
- **Priority:** Useful for all processes

SPECIFIC ATTRIBUTES (go in dedicated tables):

- **PhoneNumber:** Only for TSK_PHONE_CALL
- **EmailSubject:** Only for TSK_EMAIL
- **SampleQuantity:** Only for RCH_SAMPLES
- **TechnicalSpecs:** Only for TEH_DATA_SHEET
- **ApprovalFile:** Only for APR_APPROVAL

⚙️ PROBLEM 3: Optimized Relationships for FileMaker Performance

RECOMMENDED ARCHITECTURE:

Relationship Graph Structure:

```
LOG (Master Table)
├── 1:1 → TSK_PHONE_CALL (ProcessID match)
├── 1:1 → TSK_EMAIL (ProcessID match)
├── 1:1 → PRJ_PROJECT (ProcessID match)
├── 1:1 → RCH_SAMPLES (ProcessID match)
├── 1:1 → TEH_DATA_SHEET (ProcessID match)
├── 1:1 → APR_APPROVAL (ProcessID match)
├── 1:∞ → LOG (Self-join for DNA hierarchy)
├── ∞:1 → CLIENTS (ClientID match)
└── ∞:1 → USERS (ResponsibleID match)
```

PERFORMANCE ADVANTAGES:

- **1:1 Join = Faster in FileMaker**
- **Single indexes** on ProcessID, Status, ClientID
- **Optimized found sets** for common filters
- **Fast portal loading** for related records

📋 PROBLEM 4: Concrete Implementation Examples

EXAMPLE A: TSK_PHONE_CALL Process

Scenario: Simona calls LILA COSMETIC client to confirm appointment

STEP 1 - Record in LOG:

```
ProcessID: TSK25001
ProcessType: TSK
ClientID: CLI_LILA_COSMETIC
ResponsibleID: USR_SIMONA
Status: IN_PROGRESS
DateCreated: 10/08/2025 14:30
Description: "Confirm visit appointment"
```

STEP 2 - Record in TSK_PHONE_CALL:

```
ProcessID: TSK25001 (FK)
PhoneNumber: +40-21-555-0123
CallDuration: 8 minutes
CallOutcome: "Appointment confirmed for 15/08"
```

STEP 3 - Update LOG when finished:

```
Status: DONE
DateCompleted: 10/08/2025 14:38
```

EXAMPLE B: RCH_SAMPLES Process

Scenario: Giuseppe registers LILA COSMETIC sample request

STEP 1 - Record in LOG:

```
ProcessID: RCH25002
ProcessType: RCH
ClientID: CLI_LILA_COSMETIC
ResponsibleID: USR_GIUSEPPE
Status: TO_DO
DateCreated: 10/08/2025 15:00
Description: "Request 120 color strand samples"
ParentProcessID: TSK25001
```

STEP 2 - Record in RCH_SAMPLES:

```
ProcessID: RCH25002 (FK)
SampleType: "Hair strands"
Quantity: 120
Specifications: "4-panel chart, blonde-brown colors"
DeliveryMethod: "Express courier"
```

EXAMPLE C: TEH_DATA_SHEET Process

Scenario: Marco creates technical sheet for LILA product

STEP 1 - Record in LOG:

```
ProcessID: TEH25003
ProcessType: TEH
ClientID: CLI_LILA_COSMETIC
ResponsibleID: USR_MARCO
Status: TO_DO
DateCreated: 10/08/2025 16:00
Description: "Technical sheet LILA 4-panel chart"
ParentProcessID: RCH25002
```

STEP 2 - Record in TEH_DATA_SHEET:

```
ProcessID: TEH25003 (FK)
ProductCode: "CART_LILA_4ANTE_120"
TechnicalSpecs: "Dimensions 15x20cm, 120 strands, 4 pages"
Materials: "Spools BOB001, BOB002, 200gr paper"
EstimatedTime: "3 hours production"
```

PART C: SIMPLIFIED DATABASE SCHEMA TO START

🚀 PHASE 1 IMPLEMENTATION - Core Tables

For THE BRIDGE, start with these 6 tables:

1. LOG (Master Process Table)

```
sql
```

```
Fields:
- ProcessID (Text, 50, Primary Key, Indexed)
- ProcessType (Text, 10, Indexed, Dropdown: TSK|PRJ|RCH|TEH|APR)
- ClientID (Text, 20, Foreign Key)
- ResponsibleID (Text, 10, Foreign Key)
- CreatorID (Text, 10, Foreign Key)
- Status (Text, 20, Dropdown: TO_DO|IN_PROGRESS|DONE)
- DateCreated (Timestamp, Auto-enter)
- DateStarted (Timestamp)
- DateCompleted (Timestamp)
- ParentProcessID (Text, 50, for DNA hierarchy)
- Description (Text, 500 chars)
- Priority (Text, 10, Dropdown: High|Medium|Low)
```

2. TSK_PHONE_CALL (Task Process Table)

```sql
Fields:
- ProcessID (Text, 50, Primary Key = Foreign Key to LOG)
- PhoneNumber (Text)
- CallDuration (Number, minutes)
- CallOutcome (Text, 1000 chars)
- CallDirection (Text, Dropdown: Incoming|Outgoing)
```

3. PRJ_PROJECT (Project Process Table)

```sql
Fields:
- ProcessID (Text, 50, Primary Key = Foreign Key to LOG)
- ProjectName (Text, 100 chars)
- ProjectType (Text, Dropdown: STANDARD|CUSTOM|PROTOTYPE)
- EstimatedValue (Number, currency)
- Deadline (Date)
- ProjectStatus (Text, Dropdown: PLANNING|ACTIVE|ON_HOLD|COMPLETED)
```

4. RCH_SAMPLES (Research Process Table)

```sql
```

```
Fields:
- ProcessID (Text, 50, Primary Key = Foreign Key to LOG)
- SampleType (Text, Dropdown: CIOCCHE|STAMPATI|ACCESSORI)
- Quantity (Number)
- Specifications (Text, 2000 chars)
- DeliveryMethod (Text, Dropdown: COURIER|POST|PICKUP)
- ClientSample (Container, image)
```

5. TEH_DATA_SHEET (Technical Process Table)

```sql
Fields:
- ProcessID (Text, 50, Primary Key = Foreign Key to LOG)
- ProductCode (Text, 30)
- TechnicalSpecs (Text, 3000 chars)
- Materials (Text, 1000 chars)
- EstimatedTime (Number, hours)
- Drawings (Container, PDF/images)
```

6. APR_APPROVAL (Approval Process Table)

```sql
Fields:
- ProcessID (Text, 50, Primary Key = Foreign Key to LOG)
- ApprovalType (Text, Dropdown: CLIENT|INTERNAL|TECHNICAL)
- ApprovalFile (Container, PDF/image)
- ApprovalDeadline (Date)
- ApprovalStatus (Text, Dropdown: PENDING|APPROVED|REJECTED|REVISION)
- ApprovalNotes (Text, 2000 chars)
```

🔗 RELATIONSHIPS for FileMaker

```
LOG::ProcessID = TSK_PHONE_CALL::ProcessID (1:1)
LOG::ProcessID = PRJ_PROJECT::ProcessID (1:1)
LOG::ProcessID = RCH_SAMPLES::ProcessID (1:1)
LOG::ProcessID = TEH_DATA_SHEET::ProcessID (1:1)
LOG::ProcessID = APR_APPROVAL::ProcessID (1:1)
LOG::ClientID = CLIENTS::ClientID (∞:1)
LOG::ResponsibleID = USERS::UserID (∞:1)
LOG::ParentProcessID = LOG::ProcessID (∞:1, self-join for hierarchy)
```

PART D: FUNDAMENTAL OPERATIONAL PRINCIPLES

✅ DO - What to Do

1. **Single Source of Truth:** Every ProcessID exists only in LOG table
2. **1:1 Relationships:** Every specific process connects to LOG with same ProcessID
3. **Universal Status:** All processes use TO_DO → IN_PROGRESS → DONE
4. **DNA Consistency:** Every ProcessID follows format: PRXYYNNN
5. **Time Tracking:** Always timestamp DateCreated/Started/Completed in LOG

❌ DON'T - What NOT to Do

1. **DON'T duplicate fields:** If it exists in LOG, don't put it in specific tables
2. **DON'T use TaskID:** Only ProcessID as unique identifier
3. **DON'T bypass LOG:** Every process MUST have record in LOG
4. **DON'T custom statuses:** Always use standard 3P3 states
5. **DON'T scattered logic:** All business logic on processes goes through LOG

🎯 SUCCESS CRITERIA

When implementation is correct:

- You can see all ongoing processes from a single screen (LOG)
- You can filter by client and see their entire history
- You can drill-down into any process for specific details
- You never have information duplications
- Performance stays fast even with 1000+ processes

PART E: PRACTICAL NEXT STEPS

📅 IMPLEMENTATION ROADMAP

WEEK 1: ☐ Create 6 core tables (LOG + 5 specific) ☐ Configure base relationships ☐ Test manual data entry

WEEK 2: ☐ Base layout for LOG (process list) ☐ Specific layouts for TSK, PRJ, RCH, TEH, APR ☐ Script for auto-generation ProcessID (format PRXYYNNN)

WEEK 3: ☐ Portal integration between LOG and specific tables ☐ Filters for Status, Client, Responsible ☐ Test workflow TSK→PRJ→RCH→TEH→APR

WEEK 4: ☐ Operational dashboard ☐ Base reporting ☐ Team training and feedback

🤝 CONTINUOUS SUPPORT

For specific technical doubts:

1. Always test first on 10-20 records
2. Performance problems → check indexing
3. Relationship problems → verify 1:1 cardinality
4. UI problems → simplify before adding complexity

Remember: 3P3 is simple in concept, the difficulty is resisting the temptation to complicate!

CONCLUSION

This document gives you the **philosophical compass** (controlling the entity) and **concrete technical solutions** (LOG schema + specific tables) to implement 3P3 in FileMaker.

The secret of success: Start simple, test frequently, and always remember that we are "controlling entities through actions and resources".

Next document: Hierarchical DNA and Automatic Generation Algorithms

---

KOOL TOOL SRL - Craiova, România
"Everything is a Process - including understanding 3P3!"