

2. Nueva Arquitectura (Basada en Dominios)

```

graph TB
    subgraph Fuentes["Fuentes Externas"]
        SITA["Red SITA  
Telex Messages"]
        AENA["AENA CDM"]
        SFTP["SFTP"]
        CKI["Check-in"]
        NIMBUS["Nimbus"]
    end

    subgraph InServices["Servicios IN"]
        RECEIVE_SITA["Recibe SITA"]
        RECEIVE_NIMBUS["Recibe Nimbus"]
        RECEIVE_CDM["Recibe CDM"]
        RECEIVE_GAUD["Recibe GAUD"]
        RECEIVE_SSIM["Recibe SSIM"]
    end

    subgraph Parsers["Parsers (con BD propia)"]
        subgraph TELEXP["Telex Parser"]
            TELEXP_P["TELEX_P  
21 tipos de mensajes"]
            TELEXP_MSG_DB["TELEX_MSG_DB  
raw messages"]
            TELEXP_DB["TELEX_DB  
parsed data"]
        end

        subgraph NIMBUSP["Nimbus Parser"]
            NIMBUS_P["NIMBUS_P"]
            NIMBUS_MSG_DB["NIMBUS_MSG_DB  
raw messages"]
            NIMBUS_DB["NIMBUS_DB  
parsed data"]
        end

        subgraph AENAP["AENA Parser"]
            AENA_P["AENA_P  
CDM"]
            AENA_MSG_DB["AENA_MSG_DB  
raw messages"]
            AENA_DB["AENA_DB  
parsed data"]
        end

        subgraph CKIP["CKI Parser"]
        end
    end

```

```

        CKI_P["cki-parser
GAUD"]
        CKI_MSG_DB["fh_cki_messages
raw messages"]
        CKI_DB["fh_cki
parsed data"]
        end
        subgraph SSIMP["SSIM Parser"]
            SSIM_P["ssim-parser
Schedules"]
            SSIM_MSG_DB["fh_ssim_messages
raw messages"]
            SSIM_DB["fh_ssim
parsed data"]
            end
        end

        subgraph OrchQueue["Orchestrator Queue"]
            ORCH_Q["orchestrator_queue.fifo
Orden FIFO garantizado por SQS"]
            end

        subgraph Orchestrator["Flight Orchestrator"]
            ORCH["Flight Orchestrator
-----
* Recibe datos parseados
* Extrae FUID + 6 campos
* Entrada específica por tipo
* Flexible matching
* Precedence management
* Domain routing"]
            ORCH_DB["fh_orchestrator
-----
flights (identifiers)
message_log (audit)"]
            end

        subgraph DomainQueues["Domain Event Queues (FIFO) – 13 Dominios"]
            Q_RES["resources_events.fifo"]
            Q_TIME["timeline_events.fifo"]
            Q_DELAY["delays_events.fifo"]
            Q_CREW["crew_events.fifo"]
            Q_ALERT["alerts_events.fifo"]
            Q_PAX["passengers_events.fifo"]
            Q_BAG["baggage_events.fifo"]
            Q_FUEL["fuel_events.fifo"]
            Q_AIR["aircraft_events.fifo"]
            Q_SCH["schedules_events.fifo"]
            Q_ONWARD["onward_events.fifo"]
            Q_CODE["codeshare_events.fifo"]
            end

        subgraph Domains["Dominios de Negocio (13 Dominios)"]
            subgraph Resources["Resources Domain"]

```

```
    RES_SVC["Resources Service"]
    RES_DB(["fh_resource"])
end

subgraph Timeline["Timeline Domain"]
    TIME_SVC["Timeline Service"]
    TIME_DB(["fh_timeline"])
end

subgraph Delays["Delays Domain"]
    DELAY_SVC["Delays Service"]
    DELAY_DB(["fh_delay"])
end

subgraph Crew["Crew Domain"]
    CREW_SVC["Crew Service"]
    CREW_DB(["fh_crew"])
end

subgraph Alerts["Alerts Domain"]
    ALERT_SVC["Alerts Service"]
    ALERT_DB(["fh_alert"])
end

subgraph Passengers["Passengers Domain"]
    PAX_SVC["Passengers Service"]
    PAX_DB(["fh_pax"])
end

subgraph Baggage["Baggage Domain"]
    BAG_SVC["Baggage Service"]
    BAG_DB(["fh_bag"])
end

subgraph Fuel["Fuel Domain"]
    FUEL_SVC["Fuel Service"]
    FUEL_DB(["fh_fuel"])
end

subgraph Aircraft["Aircraft Domain"]
    AIR_SVC["Aircraft Service"]
    AIR_DB(["fh_aircraft"])
end

subgraph Schedules["Schedules Domain"]
    SCH_SVC["Schedules Service"]
    SCH_DB(["fh_schedule"])
end

subgraph OnwardFlights["Onward Flights Domain"]
    ONWARD_SVC["Onward Flights Service"]
    ONWARD_DB(["fh_onward"])
end
```

```

        subgraph Codeshare["Codeshare Domain"]
            CODE_SVC["Codeshare Service"]
            CODE_DB[("fh_codeshare")]
        end
    end

    subgraph Publisher["Event Publisher"]
        PUB_SVC["Event Publisher
    -----
    * FUID → External ID
    * 6 campos únicos
    * Payload adaptation
    * EventBridge publishing"]
        PUB_CACHE[("Redis Cache
    Flight mappings")]
    end

    subgraph External["Sistemas Externos"]
        SNS["AWS EventBridge
    flight-events-bus"]
        CONSUMERS["Consumers
    Mobile Apps
    Web
    Partners"]
    end

    SITA --> TELEX_IN
    NIMBUS --> NIMBUS_IN
    AENA --> AENA_IN
    CKI --> CKI_IN
    SFTP --> SSIM_IN

    TELEX_IN --> TELEX_MSG_DB
    TELEX_MSG_DB --> TELEX_IN
    TELEX_IN --> TELEX_P
    TELEX_P --> TELEX_MSG_DB
    TELEX_MSG_DB --> TELEX_P
    TELEX_P --> TELEX_DB
    TELEX_DB --> TELEX_P

    NIMBUS_IN --> NIMBUS_MSG_DB
    NIMBUS_MSG_DB --> NIMBUS_IN
    NIMBUS_IN --> NIMBUS_P
    NIMBUS_P --> NIMBUS_MSG_DB
    NIMBUS_MSG_DB --> NIMBUS_P
    NIMBUS_P --> NIMBUS_DB
    NIMBUS_DB --> NIMBUS_P

    AENA_IN --> AENA_MSG_DB
    AENA_MSG_DB --> AENA_IN
    AENA_IN --> AENA_P
    AENA_P --> AENA_MSG_DB
    AENA_MSG_DB --> AENA_P
    AENA_P --> AENA_DB

```

AENA_DB --> AENA_P

CKI_IN --> CKI_MSG_DB

CKI_MSG_DB --> CKI_IN

CKI_IN --> CKI_P

CKI_P --> CKI_MSG_DB

CKI_MSG_DB --> CKI_P

CKI_P --> CKI_DB

CKI_DB --> CKI_P

SSIM_IN --> SSIM_MSG_DB

SSIM_MSG_DB --> SSIM_IN

SSIM_IN --> SSIM_P

SSIM_P --> SSIM_MSG_DB

SSIM_MSG_DB --> SSIM_P

SSIM_P --> SSIM_DB

SSIM_DB --> SSIM_P

TELEX_P --> ORCH_Q

NIMBUS_P --> ORCH_Q

AENA_P --> ORCH_Q

CKI_P --> ORCH_Q

SSIM_P --> ORCH_Q

ORCH_Q --> ORCH

ORCH --> ORCH_DB

ORCH_DB --> ORCH

ORCH --> Q_RES

ORCH --> Q_TIME

ORCH --> Q_DELAY

ORCH --> Q_CREW

ORCH --> Q_ALERT

ORCH --> Q_PAX

ORCH --> Q_BAG

ORCH --> Q_FUEL

ORCH --> Q_AIR

ORCH --> Q_SCH

ORCH --> Q_ONWARD

ORCH --> Q_CODE

Q_RES --> RES_SVC

RES_SVC --> RES_DB

RES_DB --> RES_SVC

Q_TIME --> TIME_SVC

TIME_SVC --> TIME_DB

TIME_DB --> TIME_SVC

Q_DELAY --> DELAY_SVC

DELAY_SVC --> DELAY_DB

DELAY_DB --> DELAY_SVC

Q_CREW --> CREW_SVC

CREW_SVC --> CREW_DB
CREW_DB --> CREW_SVC

Q_ALERT --> ALERT_SVC
ALERT_SVC --> ALERT_DB
ALERT_DB --> ALERT_SVC

Q_PAX --> PAX_SVC
PAX_SVC --> PAX_DB
PAX_DB --> PAX_SVC

Q_BAG --> BAG_SVC
BAG_SVC --> BAG_DB
BAG_DB --> BAG_SVC

Q_FUEL --> FUEL_SVC
FUEL_SVC --> FUEL_DB
FUEL_DB --> FUEL_SVC

Q_AIR --> AIR_SVC
AIR_SVC --> AIR_DB
AIR_DB --> AIR_SVC

Q_SCH --> SCH_SVC
SCH_SVC --> SCH_DB
SCH_DB --> SCH_SVC

Q_ONWARD --> ONWARD_SVC
ONWARD_SVC --> ONWARD_DB
ONWARD_DB --> ONWARD_SVC

Q_CODE --> CODE_SVC
CODE_SVC --> CODE_DB
CODE_DB --> CODE_SVC

RES_SVC --> PUB_SVC
TIME_SVC --> PUB_SVC
DELAY_SVC --> PUB_SVC
CREW_SVC --> PUB_SVC
ALERT_SVC --> PUB_SVC
PAX_SVC --> PUB_SVC
BAG_SVC --> PUB_SVC
FUEL_SVC --> PUB_SVC
AIR_SVC --> PUB_SVC
SCH_SVC --> PUB_SVC
ONWARD_SVC --> PUB_SVC
CODE_SVC --> PUB_SVC

PUB_SVC --> PUB_CACHE
PUB_CACHE --> PUB_SVC
ORCH_DB -. -> |Lookup FUID| PUB_SVC

PUB_SVC --> SNS
SNS --> CONSUMERS

Beneficios Clave:

- **✓ Arquitectura IN → Parser → Orchestrator:** Flujo claro y separado (telex-in → telex-parser, nimbus-in → nimbus-parser, etc.)
- **✓ Servicios IN dedicados:** Cada fuente tiene su servicio de ingesta (telex-in, nimbus-in, aena-in, cki-in, ssim-in)
- **✓ Parsers con BD propia:** Cada parser guarda en su base de datos (fh_telex, fh_nimbus, fh_aena, fh_cki, fh_ssim)
- **✓ Telex Parser unificado:** Un solo parser procesa los 21+ tipos de mensajes telex
- **✓ Auditoría completa:** Todos los mensajes parseados se guardan antes de enviar al orchestrator
- **✓ Cola única del orchestrator:** `orchestrator_queue.fifo` recibe de TODOS los parsers
- **✓ Flight Orchestrator:** Extrae FUID + 6 campos con entrada específica por tipo de dato
- **✓ 13 dominios granulares:** Resources, Timeline, Delays, Crew, Alerts, Passengers, Baggage, Fuel, Aircraft, Schedules, Onward Flights, Codeshare
- **✓ Dominios independientes** con sus propias tablas o bases de datos
- **✓ FUID único** (ULID) para uso interno
- **✓ 6 campos de identificación** se guardan en cada tabla de dominio
- **✓ Event Publisher** usa solo los 6 campos (NO publica FUID a EventBridge)
- **✓ Prefijos consistentes:** `fh_resource_`, `fh_timeline_`, `fh_delay_`, `fh_crew_`, `fh_alert_`, `fh_pax_`, `fh_bag_`, `fh_fuel_`, etc.
- **✓ Escalado independiente** por parser y por dominio
- **✓ Deploys independientes**
- **✓ Matching flexible** (IATA/ICAO)

3. Flujo de Datos: Mensaje MVT Completo

```
sequenceDiagram
    participant SRC as SITA Network
    participant TIN as telex-in
    participant TPARSER as telex-parser
    participant TELMSGDB as fh_telex_messages DB
    participant TELDB as fh_telex DB
    participant ORCHQ as orchestrator_queue.fifo
    participant ORC as Flight Orchestrator
    participant DB as fh_orchestrator
    participant Q1 as passengers_events.fifo
    participant Q2 as fuel_events.fifo
    participant Q3 as onward_events.fifo
    participant PAX as Passengers Service
    participant FUEL as Fuel Service
    participant ONWARD as Onward Flights Service
    participant PUB as Event Publisher
    participant EB as AWS EventBridge
```

SRC->>TIN: Telex Message (raw)

Note over TIN: Recibe mensaje raw

```

    TIN-->>TEMSGDB: Guarda mensaje raw
en fh_telex_messages
    TELMSGDB-->>TIN: Message ID
    TIN-->>TPARSER: Notifica nuevo mensaje

    Note over TPARSER: 1. Lee mensaje de fh_telex_messages
    TELMSGDB-->>TPARSER: Mensaje raw
    Note over TPARSER: 2. Detecta tipo/subtipo
usando regex patterns
type='MVT', subtype='AA'
    Note over TPARSER: 3. Parsea el mensaje MVT-AA
Extrae todos los campos
    TPARSER-->>TELDDB: Guarda datos parseados
en fh_telex
    TELDB-->>TPARSER: UUID generado
    Note over TPARSER: 4. Envuelve datos parseados
en CloudEvents
    TPARSER-->>ORCHQ: CloudEvent
{type:'MVT', subtype:'AA',
parsedData: {...}}

    ORCHQ-->>ORC: Consume (orden FIFO garantizado por SQS)

    Note over ORC: Recibe datos parseados
Extrae FUID + 6 campos
según tipo de mensaje MVT
    ORC-->>DB: Verificar/Guardar FUID + 6 campos
    DB-->>ORC: FUID confirmado

    Note over ORC: Extract domain data from MVT
    Note over ORC: MVT contains:
* Passengers: 180
* Fuel: 12500 kg
* Times: ATD 08:30

    ORC-->>Q1: PassengerEvent (FUID + 6 campos + data)
    ORC-->>Q2: FuelEvent (FUID + 6 campos + data)
    ORC-->>Q3: OnwardFlightEvent (FUID + 6 campos + data)

    Q1-->>PAX: Consume
    Note over PAX: UPSERT passenger_summary
SET fuid = '01HQZ8X9...',
operationDate = '2025-01-14',
flightDesignator = '347',
airlineDesignator = 'IB',
...
    PAX-->>PUB: passengers.updated (FUID + 6 campos)

    Q2-->>FUEL: Consume
    Note over FUEL: UPSERT fuel_summary
Guarda FUID + 6 campos + data
    FUEL-->>PUB: fuel.updated (FUID + 6 campos)

    Q3-->>ONWARD: Consume

```


Note over ONWARD: UPSERT fh_onward_flight
 Guarda FUID + 6 campos + data
 ONWARD->>PUB: onward_flight.updated (FUID + 6 campos)

Note over PUB: Los 6 campos ya vienen
 en cada evento de dominio.
 FUID NO se publica a EventBridge

Note over PUB: Build external payload
 SOLO con los 6 campos
 (sin FUID)

PUB->>EB: Publish external event
 Note over EB: Topic: flight.updated
 Payload includes:
 flightIdentifier (6 campos)
 passengers, fuel, times
 (NO incluye FUID)

4. Identificadores: FUID + 6 Campos de Identificación

```
graph TB
    subgraph TelexParsing4["TELEX PARSING"]
        TIN4["telex-in"]
        Recibe SITA[""]
        TPARSER4["telex-parser"]
        -----
        Detecta tipo/subtipo (21+)
        Parsea mensaje
        Guarda en fh_telex_messages y fh_telex[""]
        TELEX_MSG_DB4["fh_telex_messages"]
        raw[""]
        TELEX_DB4["fh_telex"]
        parsed[""]
        end
    end

    subgraph NimbusParsing4["NIMBUS PARSING"]
        NIN4["nimbus-in"]
        Recibe Nimbus[""]
        NPARSER4["nimbus-parser"]
        Parsea fuel data[""]
        NIMBUS_MSG_DB4["fh_nimbus_messages"]
        raw[""]
        NIMBUS_DB4["fh_nimbus"]
        parsed[""]
        end
    end

    subgraph AENAParsing4["AENA PARSING"]
        AIN4["aena-in"]
        Recibe AENA[""]
    end
```

```

        AENA_P4["aena-parser
Parsea AENA data"]
        AENA_MSG_DB4["fh_aena_messages
raw"]]
        AENA_DB4["fh_aena
parsed"]]
        end

        subgraph CKIParsing4["CKI PARSING"]
            CIN4["cki-in
Recibe CKI"]
            CKI_P4["cki-parser
Parsea CKI data"]
            CKI_MSG_DB4["fh_cki_messages
raw"]]
            CKI_DB4["fh_cki
parsed"]]
            end

        subgraph SSIMParsing4["SSIM PARSING"]
            SSIM_S3_4["S3 Bucket
SSIM files"]
            SSIM_P4["ssim-parser
Lee de S3
Parsea schedules"]
            SSIM_MSG_DB4["fh_ssim_messages
raw"]]
            SSIM_DB4["fh_ssim
parsed"]]
            end

        subgraph OrchQueue4["ORCHESTRATOR QUEUE"]
            ORCHQ4["orchestrator_queue.fifo
Recibe de TODOS los parsers"]
            end

        subgraph Orchestrator["ORCHESTRATOR (Extracción)"]
            ORCH_EXT["Flight Orchestrator
-----
Extrae del mensaje:
* FUID (de DB o nuevo ULID)
* operationDate
* flightDesignator
* operationalSuffix
* airlineDesignator
* departureAirport
* departureNumber

Entrada específica por tipo"]
            end

        subgraph Internal["USO INTERNO (Dominios)"]
            FUID["FUID (ULID)
-----

```

01HQZ8X9Y1K2M3N4P5Q6R7S8T9

- * Único en todo el sistema
- * Inmutable
- * 26 caracteres
- * Ordenable
- * Extraído por Orchestrator"]

CAMPOS["6 Campos de Identificación

 operationDate: 2025-01-14
 flightDesignator: 347
 operationalSuffix: (empty)
 airlineDesignator: IB
 departureAirport: MAD
 departureNumber: 1

- * Se guardan en CADA tabla
- * Permiten queries sin joins"]

ORCH_INT["Flight Orchestrator
 extrae y envía FUID + 6 campos"]

RES_INT["Resources Service
 guarda FUID + 6 campos"]

TIME_INT["Timeline Service
 guarda FUID + 6 campos"]

PAX_INT["Passengers Service
 guarda FUID + 6 campos"]

FUEL_INT["Fuel Service
 guarda FUID + 6 campos"]

FUID --> ORCH_INT

CAMPOS --> ORCH_INT

ORCH_INT --> RES_INT

ORCH_INT --> TIME_INT

ORCH_INT --> PAX_INT

ORCH_INT --> FUEL_INT

end

subgraph External["PUBLICACIÓN EXTERNA"]

PUB_MAP["Event Publisher

 Los 6 campos YA VIENEN
 en cada evento de dominio.
 Solo necesita formatear."]

EXT_ID["External ID (6 campos)

 airlineDesignator: IB
 flightDesignator: 347
 operationalSuffix: (empty)
 departureAirport: MAD
 operationDate: 2025-01-14
 departureNumber: 1

- * NO incluye FUID
- * Estándar aeronáutico
- * Compatible con externos"]

EB_EXT["AWS EventBridge

flight-events-bus

Consumidores se suscriben:

- * Mobile Apps
- * Web Dashboard
- * Partner Systems"]

EXT_ID --> EB_EXT
end

TIN4 --> TELEX_MSG_DB4
TELEX_MSG_DB4 --> TIN4
TIN4 --> TPARSER4
TPARSER4 --> TELEX_MSG_DB4
TELEX_MSG_DB4 --> TPARSER4
TPARSER4 --> TELEX_DB4
TELEX_DB4 --> TPARSER4

NIN4 --> NIMBUS_MSG_DB4
NIMBUS_MSG_DB4 --> NIN4
NIN4 --> NPARSER4
NPARSER4 --> NIMBUS_MSG_DB4
NIMBUS_MSG_DB4 --> NPARSER4
NPARSER4 --> NIMBUS_DB4
NIMBUS_DB4 --> NPARSER4

AIN4 --> AENA_MSG_DB4
AENA_MSG_DB4 --> AIN4
AIN4 --> AENA_P4
AENA_P4 --> AENA_MSG_DB4
AENA_MSG_DB4 --> AENA_P4
AENA_P4 --> AENA_DB4
AENA_DB4 --> AENA_P4

CIN4 --> CKI_MSG_DB4
CKI_MSG_DB4 --> CIN4
CIN4 --> CKI_P4
CKI_P4 --> CKI_MSG_DB4
CKI_MSG_DB4 --> CKI_P4
CKI_P4 --> CKI_DB4
CKI_DB4 --> CKI_P4

SSIM_S3_4 --> SSIM_P4
SSIM_P4 --> SSIM_MSG_DB4
SSIM_MSG_DB4 --> SSIM_P4
SSIM_P4 --> SSIM_DB4
SSIM_DB4 --> SSIM_P4

```

TPARSER4 --> ORCHQ4
NPARSER4 --> ORCHQ4
AENA_P4 --> ORCHQ4
CKI_P4 --> ORCHQ4
SSIM_P4 --> ORCHQ4

ORCHQ4 --> ORCH_EXT
ORCH_EXT --> ORCH_INT

RES_INT --> PUB_MAP
TIME_INT --> PUB_MAP
PAX_INT --> PUB_MAP
FUEL_INT --> PUB_MAP

PUB_MAP --> EXT_ID

```

Comparación: Tabla flight_departure_info

Arquitectura Actual (Monolítica)

```

graph TB
    subgraph Monolito["flight_departure_info (100+ campos mezclados)"]
        TABLA["Una sola tabla con TODO mezclado  
-----"]

        PAX_FIELDS["PASAJEROS:  
total_passengers  
checked_in_passengers  
boarded_passengers  
adults  
children  
infants"]

        BAG_FIELDS["EQUIPAJE:  
baggage_pieces  
baggage_weight  
cargo_weight  
mail_weight"]

        FUEL_FIELDS["COMBUSTIBLE:  
fuel_uplift  
fuel_planned  
fuel_remaining  
fuel_density"]

        TIME_FIELDS["TIEMPOS:  
std, etd, atd  
sta, eta, ata  
gate_departure"]
    end

```

```

gate_arrival"]

        CREW_FIELDS["TRIPULACIÓN:
crew_count
cockpit_crew
cabin_crew"]

        OPS_FIELDS["OPERACIONES:
flight_status
delay_code
cancellation_reason"]

        TABLA --> PAX_FIELDS
        TABLA --> BAG_FIELDS
        TABLA --> FUEL_FIELDS
        TABLA --> TIME_FIELDS
        TABLA --> CREW_FIELDS
        TABLA --> OPS_FIELDS
    end

    PROBLEM["❌ PROBLEMAS:
* Imposible escalar dominios
* Deploys todo-o-nada
* Ownership difuso
* Queries complejas
* Locking contention"]

```

Nueva Arquitectura (Separada por Dominios)

```

graph TB
    subgraph Nueva["13 Dominios Separados"]
        subgraph D1["Resources Domain"]
            RES_TAB["gate_assignments
            stand_assignments
            belt_assignments
            -----
            fuid (PK)
            + 6 campos"]
        end

        subgraph D2["Timeline Domain"]
            TIME_TAB["flight_times
            -----
            fuid (PK)
            std, etd, atd
            sta, eta, ata
            + 6 campos"]
        end

        subgraph D3["Delays Domain"]
            DELAY_TAB["flight_delays

```

```
delay_codes
-----
fuid (PK)
delay_code
delay_minutes
+ 6 campos"]
    end

    subgraph D4["Crew Domain"]
        CREW_TAB["crew_manifest
    -----
fuid (PK)
total_crew
cockpit
cabin
+ 6 campos"]
    end

    subgraph D5["Alerts Domain"]
        ALERT_TAB["flight_alerts
    -----
fuid (PK)
alert_type
severity
+ 6 campos"]
    end

    subgraph D6["Passengers Domain"]
        PAX_TAB["passenger_summary
    -----
fuid (PK)
total_passengers
checked_in
boarded
+ 6 campos"]
    end

    subgraph D7["Baggage Domain"]
        BAG_TAB["baggage_summary
    -----
fuid (PK)
pieces
weight
+ 6 campos"]
    end

    subgraph D8["Fuel Domain"]
        FUEL_TAB["fuel_summary
    -----
fuid (PK)
uplift
planned
+ 6 campos"]
    end
```

```

        subgraph D9["Aircraft Domain"]
            AIR_TAB["aircraft_info
-----
fuid (PK)
tail_number
aircraft_type
+ 6 campos"]
        end

        subgraph D10["Schedules Domain"]
            SCH_TAB["flight_schedules
-----
fuid (PK)
scheduled_times
+ 6 campos"]
        end

        subgraph D11["Onward Flights Domain"]
            ONWARD_TAB["onward_flights
-----
fuid (PK)
connection_type
+ 6 campos"]
        end

        subgraph D12["Codeshare Domain"]
            CODE_TAB["codeshare_flights
-----
fuid (PK)
operating_carrier
+ 6 campos"]
        end

        subgraph D13["Event Publisher"]
            PUB["Convierte FUID
a 6 campos externos
y publica"]
        end

        end

        BENEFITS["✅ BENEFICIOS:
* Escalado independiente
* Deploy por dominio
* Ownership claro
* Queries simples
* Sin contention
* 13 dominios especializados"]

```

6. Los 6 Campos de Identificación en Cada Tabla

Cada tabla de dominio guarda estos 6 campos para permitir queries directas sin joins:

```
interface FlightIdentifiers {
  operationDate: Date; // 2025-01-14
  flightDesignator: string; // "347"
  operationalSuffix: string; // "" o "A", "B"
  airlineDesignator: string; // "IB"
  departureAirport: string; // "MAD"
  departureNumber: number; // 1, 2, 3... (turnarounds)
}
```

Ejemplo en tabla **passenger_summary**:

```
CREATE TABLE passenger_summary (
  id UUID PRIMARY KEY,
  fuid VARCHAR(26) NOT NULL,

  -- Los 6 campos de identificación
  operation_date DATE NOT NULL,
  flight_designator VARCHAR(10) NOT NULL,
  operational_suffix VARCHAR(3) DEFAULT '',
  airline_designator VARCHAR(3) NOT NULL,
  departure_airport VARCHAR(3) NOT NULL,
  departure_number INTEGER NOT NULL DEFAULT 1,

  -- Datos de pasajeros
  total_passengers INTEGER,
  checked_in_passengers INTEGER,
  boarded_passengers INTEGER,
  -- ...

  -- Índices
  INDEX idx_fuid (fuid),
  INDEX idx_flight_id (airline_designator, flight_designator,
    operation_date, departure_airport)
);
```

Beneficios:

- ☒ Queries sin joins: `SELECT * FROM passenger_summary WHERE airline_designator='IB' AND flight_designator='347' AND operation_date='2025-01-14'`
- ☒ Cada dominio es independiente
- ☒ Event Publisher recibe los 6 campos directamente en cada evento

7. Onward Flights y departureNumber

```

graph LR
    subgraph Scenario["Escenario: Vuelo IB347 despegue 2 veces (Return to Base)"]
        E1["Primer Despegue  
08:30 MAD→BCN"]
        E2["Return to Base  
10:00 BCN→MAD  
(problemas técnicos)"]
        E3["Segundo Despegue  
14:00 MAD→BCN"]
    end

    subgraph OrchDB["orchestrator.flights"]
        F1["FUID: 01HQZ8X9..."]
        -----
        airline_designator: IB
        flight_designator: 347
        operation_date: 2025-01-14
        departure_airport: MAD
        departure_number: 1
        active: true"]

        F2["FUID: 01HQZ8X9... (mismo)"]
        -----
        airline_designator: IB
        flight_designator: 347
        operation_date: 2025-01-14
        departure_airport: MAD
        departure_number: 2
        active: true
        fuid_flight_principal: 01HQZ8X9..."]
    end

    subgraph OnwardFlights["fh_onward_flight"]
        OF1["id: uuid-1"]
        -----
        inbound_fuid: 01HQZ8X9...
        onward_airline: IB
        onward_flight: 347
        onward_date: 2025-01-14
        connection_type: RETURN_TO_BASE
        turnaround_time: 330 min"]
    end

    subgraph Events["Eventos Publicados a EventBridge"]
        EV1["Event 1:
        flightIdentifier:
        departureNumber: 1
        eventType: flight.departed"]

        EV2["Event 2:
        flightIdentifier:
        departureNumber: 2
        eventType: flight.departed"]
    end

```

```
end

E1 --> F1
F1 --> EV1
F1 --> OF1

E2 --> F2
F2 --> EV2

F1 -. -> |mismo vuelo| F2
```

Explicación:

- El FUID permanece **igual** (mismo vuelo)
- El **departure_number** se **incrementa** (1 → 2)
- El dominio **Onward Flights** registra la relación entre el vuelo entrante y el siguiente
- Los eventos externos incluyen **departureNumber** para diferenciar
- El campo **fuid_flight_principal** referencia al vuelo principal

Resumen de Componentes

| Componente | Responsabilidad | Base de Datos | Identificador Usado |
|---------------|---|--------------------|---------------------------|
| telex-in | Recibir y guardar mensajes raw desde SITA | fh_telex_messages | Mensaje raw |
| nimbus-in | Recibir y guardar mensajes raw desde Nimbus | fh_nimbus_messages | Mensaje raw |
| aena-in | Recibir y guardar mensajes CDM | fh_aena_messages | |
| cki-in | Recibir y guardar mensajes GAUD | fh_cki_messages | |
| ssim-in | Recibir y guardar archivos SSIM | fh_ssim_messages | Mensaje raw |
| Telex Parser | Lee de fh_telex_messages, parsea y guarda en fh_telex | fh_telex | Datos parseados solamente |
| Nimbus Parser | Lee de fh_nimbus_messages, parsea y guarda en fh_nimbus | fh_nimbus | Datos parseados solamente |

| Componente | Responsabilidad | Base de Datos | Identificador Usado |
|----------------------|---|---|----------------------------------|
| AENA Parser | Lee de fh_aena_messages, parsea y guarda en fh_aena | fh_aena | Datos parseados solamente |
| CKI Parser | Lee de fh_cki_messages, parsea y guarda en fh_cki | fh_cki | Datos parseados solamente |
| SSIM Parser | Lee de fh_ssim_messages, parsea y guarda en fh_ssim | fh_ssim | Datos parseados solamente |
| Orchestrator Queue | Cola FIFO para todos los parsers | N/A | CloudEvents (orden FIFO por SQS) |
| Flight Orchestrator | Extraer FUID + 6 campos, routing, precedencias | fh_orchestrator | FUID + 6 campos |
| Domain Services (13) | Lógica de negocio por dominio | fh_resource, fh_timeline, fh_delay, fh_crew, fh_alert, fh_pax, fh_bag, fh_fuel, fh_aircraft, fh_schedule, fh_onward, fh_codeshare | FUID + 6 campos (guardan ambos) |
| Event Publisher | Formatear y publicar (sin FUID) | Redis Cache | Solo 6 campos (NO FUID) |
| Consumers Externos | Recibir eventos | N/A | External ID (6 campos, sin FUID) |

Clave:

- **Flujo IN → Parser → Orchestrator:** Separación clara de responsabilidades
- **Servicios IN:** Reciben y guardan mensajes raw en fh_[nombre]_messages, luego notifican al parser
- **Parsers:** Cada parser lee de fh_[nombre]messages, *parsea y guarda en fh[nombre]* antes de publicar al orchestrator
 - telex-in → fh_telex_messages (raw)
 - telex-parser → fh_telex (parsed) - 21+ tipos
 - nimbus-in → fh_nimbus_messages (raw)
 - nimbus-parser → fh_nimbus (parsed)
 - aena-in → fh_aena_messages (raw)
 - aena-parser → fh_aena (parsed)

- cki-in → fh_cki_messages (raw)
- cki-parser → fh_cki (parsed)
- ssim-in → fh_ssim_messages (raw)
- ssim-parser → fh_ssim (parsed)
- **Cola única del orchestrator:** `orchestrator_queue.fifo` recibe de TODOS los parsers
- **Flight Orchestrator** extrae FUID + 6 campos con entrada específica por tipo de dato
- **6 campos en cada tabla:** Cada dominio guarda FUID + 6 campos
- **Event Publisher** recibe los 6 campos ya en cada evento, NO publica FUID a EventBridge
- **Queries sin joins:** Los 6 campos permiten buscar vuelos en cualquier tabla de dominio
- **Escalabilidad por parser:** Cada parser escala según su carga específica
- **Auditoría completa:** Todos los mensajes parseados se guardan antes del orchestrator