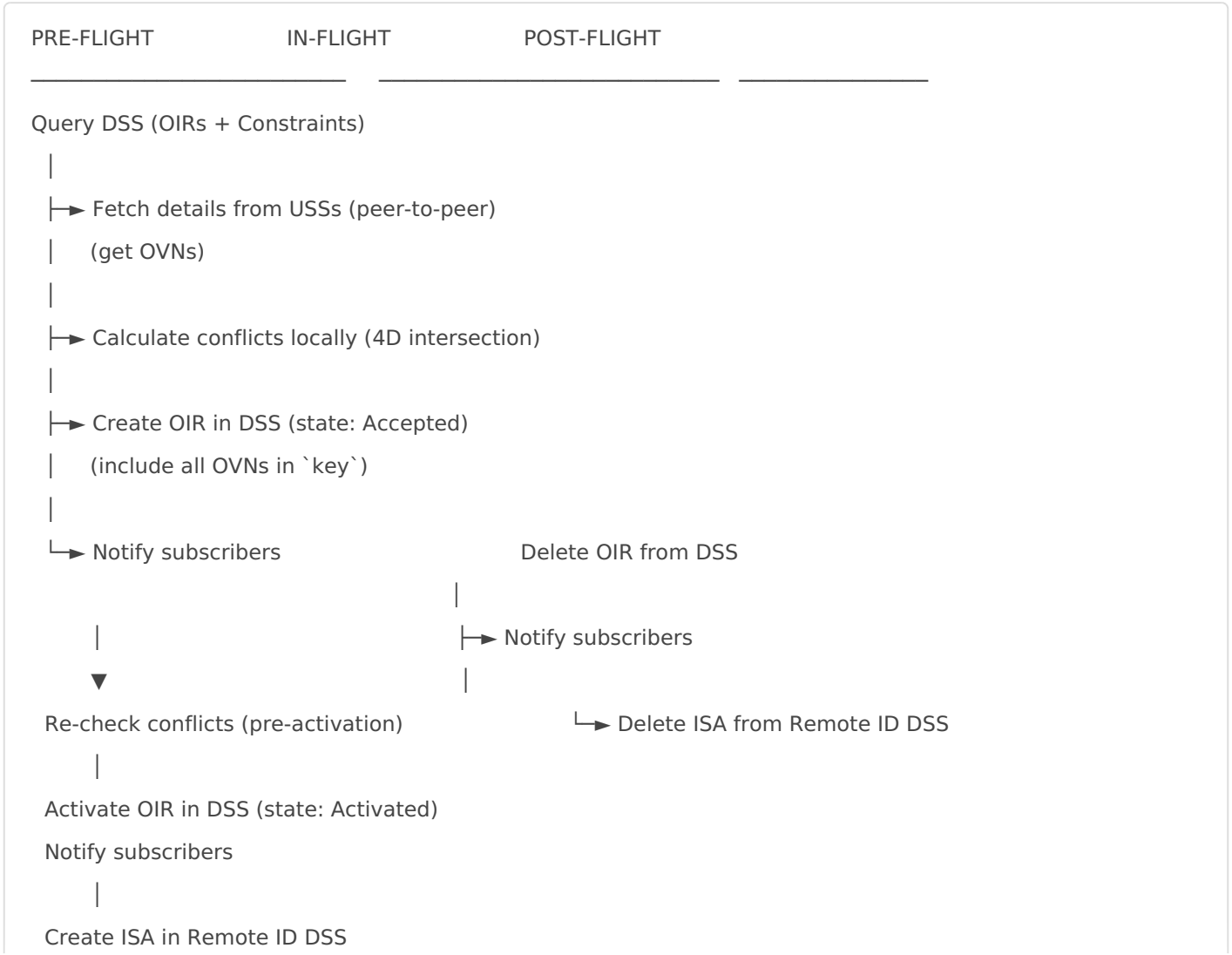
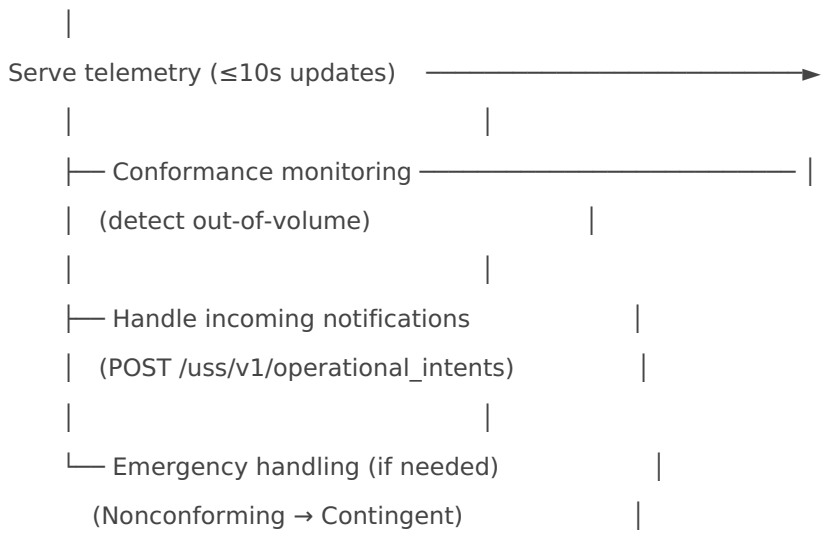
Chapter 5 — The Full Flight Lifecycle

This chapter walks through the complete technical lifecycle of a drone flight in BR-UTM — from pre-flight planning to post-flight cleanup. Every API call, data structure, and decision point is described in sequence.

5.1 Lifecycle Overview

A BR-UTM flight goes through the following stages:





5.2 Pre-Flight: Querying the Airspace

Before creating an OIR, your USS must understand the current state of the airspace in the intended operation area.

5.2.1 Query for Existing OIRs

Endpoint: `POST http://api.sandbox.brutm.dcta.mil.br/dss/dss/v1/operational_intent_references/query` **Scope:** `utm.strategic_coordination`

```
{
  "area_of_interest": {
    "volume": {
      "outline_polygon": {
        "vertices": [
          { "lat": -23.205, "lng": -45.878 },
          { "lat": -23.205, "lng": -45.872 },
          { "lat": -23.211, "lng": -45.872 },
          { "lat": -23.211, "lng": -45.878 }
        ]
      },
      "altitude_lower": { "value": 0, "reference": "W84", "units": "M" },
      "altitude_upper": { "value": 200, "reference": "W84", "units": "M" }
```

```
    },
    "time_start": { "value": "2026-07-01T10:00:00Z", "format": "RFC3339" },
    "time_end": { "value": "2026-07-01T11:00:00Z", "format": "RFC3339" }
  }
}
```

The response includes a list of `OperationalIntentReference` objects. Each contains the `uss_base_url` of the managing USS.

> **Important:** The DSS uses S2 cell indexing. It may return OIRs that do not precisely intersect your area. Your USS must perform the exact 4D intersection calculation locally after fetching the full volume details.

5.2.2 Fetch OIR Details from Peer USSs

For each OIR returned by the DSS that is **managed by another USS**, fetch the full details (including the OVN and exact volumes):

Endpoint (on the peer USS): `GET {uss_base_url}/uss/v1/operational_intents/{entityid}` **Scope:** `utm.strategic_coordination` **Token audience:** Domain of the peer USS (extracted from `uss_base_url`)

The response includes the full `OperationalIntentDetails` — volumes, off_nominal_volumes, priority, and the current OVN.

Collect the OVN from each response — you will need all of them when creating your OIR.

> **Note:** Even if local geometry calculation shows a particular OIR does not intersect with your planned volume, you **must still collect its OVN** and include it in the `key` array. The DSS guarantees airspace awareness at the S2 cell level, not the geometric level.

5.2.3 Query for Constraints

Endpoint: `POST http://api.sandbox.brutm.dcta.mil.br/dss/dss/v1/constraint_references/query` **Scope:** `utm.constraint_processing`

Same request format as the OIR query above. The response returns `ConstraintReference` objects.

5.2.4 Fetch Constraint Details from DECEA's USS

For each constraint returned, fetch the full details:

Endpoint (on DECEA's USS): `GET {constraint_uss_base_url}/uss/v1/constraints/{entityid}` **Scope:** `utm.constraint_processing`

Collect the OVN from each constraint. If your planned volume intersects a constraint, **you must not proceed** with creating the OIR in that area — deconflict or abandon the operation.

5.2.5 Local 4D Conflict Detection

After collecting all nearby OIR and Constraint volumes, perform local 4D intersection calculations:

- **Horizontal**: polygon or circle intersection in the lat/lng plane.
- **Vertical**: overlap between altitude ranges (`altitude_lower` and `altitude_upper`).
- **Temporal**: overlap between time windows (`time_start` and `time_end`).

A conflict exists only when **all three dimensions overlap simultaneously**.

If a conflict with another OIR is detected at the same priority (0), deconfliction strategies include:

- Adjusting the route or volume to avoid overlap.
- Adjusting the time window to fly when the conflicting OIR is not active.
- Suggesting the operator wait and retry.

5.3 Pre-Flight: Creating the OIR

Once you have resolved any conflicts and collected all OVNs, create the OIR in the DSS.

Endpoint: `PUT http://api.sandbox.brutm.dcta.mil.br/dss/dss/v1/operational_intent_references/{entityid}` **Scope:** `utm.strategic_coordination`

The `{entityid}` is a UUID generated by **your USS** — you own this identifier.

```
{
  "extents": [
    {
      "volume": {
```

```
      "outline_polygon": {
        "vertices": [
          { "lat": -23.207, "lng": -45.875 },
          { "lat": -23.208, "lng": -45.874 },
          { "lat": -23.209, "lng": -45.876 }
        ]
      },
      "altitude_lower": { "value": 0, "reference": "W84", "units": "M" },
      "altitude_upper": { "value": 120, "reference": "W84", "units": "M" }
    },
    "time_start": { "value": "2026-07-01T10:00:00Z", "format": "RFC3339" },
    "time_end": { "value": "2026-07-01T11:00:00Z", "format": "RFC3339" }
  }
 ],
 "key": ["<OVN_of_nearby_OIR_1>", "<OVN_of_constraint_1>"],
 "state": "Accepted",
 "uss_base_url": "https://uss.yourcompany.com/utm",
 "new_subscription": {
  "uss_base_url": "https://uss.yourcompany.com/utm",
  "notify_for_constraints": true
 },
 "flight_type": "BVLOS"
}
```

Key fields:

- `extents`: The bounding box of your operation (used for S2 indexing). This should cover all your planned volumes.
- `key`: **All OVNs** you collected in steps 5.2.2 and 5.2.4. Can be empty `[]` if the area is clear.
- `state`: Always `Accepted` at creation time.
- `uss_base_url`: Your publicly accessible base URL. Other USSs will use this to call you.
- `new_subscription`: Creates an implicit subscription for the operation's area, so you receive notifications about new OIRs/Constraints appearing in that area.
- `flight_type`: One of `VLOS`, `EVLOS`, `BVLOS`.

Response (HTTP 201):

```
{
 "operational_intent_reference": {
  "id": "2f8343be-6482-4d1b-a474-16847e01af1e",
  "manager": "your-uss-sub",
```

```
    "state": "Accepted",
    "ovn": "9d158f59-80b7-4c11-9c0c-8a2b4d936b2d",
    "uss_base_url": "https://uss.yourcompany.com/utm",
    "subscription_id": "78ea3fe8-...",
    ...
  },
  "subscribers": [
    {
      "subscriptions": [...],
      "uss_base_url": "https://other-uss.com/utm"
    }
  ]
}
```

Save the `ovn` from the response — you will need it for all subsequent updates.

5.4 Pre-Flight: Notifying Subscribers

The DSS response includes a `subscribers` list — the USSs that have subscriptions in your operation's area. You **must notify each of them** immediately after a successful DSS write.

Endpoint (on each subscriber USS): `POST {subscriber_uss_base_url}/uss/v1/operational_intents` **Scope:** `utm.strategic_coordination` **Token audience:** Domain of the subscriber USS

```
{
  "operational_intent_id": "2f8343be-6482-4d1b-a474-16847e01af1e",
  "operational_intent": {
    "reference": { ... },
    "details": {
      "volumes": [ ... ],
      "off_nominal_volumes": [],
      "priority": 0
    }
  },
  "subscriptions": [
    {
```

```
    "subscription_id": "78ea3fe8-...",
    "notification_index": 1
   }
  ]
}
```

Expected response: `204 No Content`

> **Notification timing:** You must send this notification within **5 seconds** in at least 95% of cases. If a subscriber USS is unreachable, proceed — their unavailability does not block your operation.

This same notification pattern applies for **every DSS write** — creation, updates, state changes, and deletion all require notifying the subscribers returned in that write's response.

5.5 Flight Activation

Immediately before the flight begins, activate the OIR. This is the moment the drone is cleared to launch.

5.5.1 Pre-Activation Conflict Re-Check

Before transitioning to `Activated`, perform a fresh conflict check (repeat steps 5.2.1–5.2.5). The airspace may have changed since you created the OIR. If a newly arrived OIR at the same or higher priority conflicts with your volume, **you must not activate** — deconflict first.

The rule: **first to activate wins**. If two USSs with overlapping OIRs at the same priority (0) both attempt activation simultaneously, it is the USS's responsibility to detect this and stand down if necessary.

5.5.2 Update OIR State to Activated

Endpoint: `PUT http://api.sandbox.brutm.dcta.mil.br/dss/dss/v1/operational_intent_references/{entityid}/{ovn}`
Scope: `utm.strategic_coordination`

```
{
  "extents": [ ... ],
```

```
  "key": ["<latest OVNs of all nearby entities>"],
  "state": "Activated",
  "uss_base_url": "https://uss.yourcompany.com/utm",
  "subscription_id": "78ea3fe8-...",
  "flight_type": "BVLOS"
}
```

> **Note:** The `{ovn}` in the URL must be the current OVN of your OIR (from the creation response or last update response). If the DSS returns `409 AirspaceConflictResponse`, it means new entities have appeared in the area whose OVNs you haven't acknowledged. Fetch their details, add their OVNs to `key`, and retry.

Response (HTTP 200): Returns the updated reference with a new OVN and a fresh `subscribers` list.

Notify all subscribers (step 5.4) with the `Activated` state.

5.5.3 Create the ISA (Remote ID)

Simultaneously with (or immediately after) OIR activation, register an **ISA** in the Remote ID DSS:

Endpoint: `PUT http://api.sandbox.brutm.dcta.mil.br/dss/dss/identification_service_areas/{isa_id}` **Scope:** `rid.service_provider`

```
{
  "extents": {
    "volume": {
      "outline_polygon": {
        "vertices": [
          { "lat": -23.205, "lng": -45.878 },
          { "lat": -23.205, "lng": -45.872 },
          { "lat": -23.211, "lng": -45.872 },
          { "lat": -23.211, "lng": -45.878 }
        ]
      },
      "altitude_lo": 0,
      "altitude_hi": 120
    },
    "time_start": "2026-07-01T10:00:00Z",
```

```
    "time_end": "2026-07-01T11:00:00Z"
  },
  "flights_url": "https://uss.yourcompany.com/uss/flights"
}
```

The `flights_url` tells display providers (like DECEA's monitoring tools) where to query your telemetry.

5.6 In-Flight: Serving Telemetry

Once the flight is activated and the ISA is registered, your USS must serve **live telemetry** for the drone via the Remote ID endpoints.

DECEA and other display providers will **poll** your telemetry endpoints. You must keep the data fresh.

Required update frequency: At least once every **10 seconds**.

Your USS must implement and serve:

`GET /uss/flights` — Basic flight list for a view area. **`GET /uss/flights/{id}/details`** — Detailed information for a specific flight.

See Chapter 6 — APIs to Implement for the full request/response specification.

5.7 In-Flight: Conformance Monitoring

Your USS must continuously monitor whether the drone is flying within its declared OIR volumes. This is called **conformance monitoring**.

- If the drone's reported position is inside the `volumes` of the Activated OIR → **nominal, no action needed**.
- If the drone's position is outside the `volumes` → trigger the Non-Conforming flow (Section 5.8).

5.8 In-Flight: Handling Incoming Notifications

While your flight is active, you may receive notifications from other USSs about changes in the airspace near you (new OIRs, constraint changes). These arrive as `POST` requests on your own USS:

- `POST /uss/v1/operational_intents` — A new or updated OIR in your subscription area.
- `POST /uss/v1/constraints` — A new or updated Constraint in your subscription area.

When you receive such a notification:

1. Update your internal airspace state.
2. Check whether the new/updated entity conflicts with your active OIR.
3. If a conflict exists and the new entity has **equal or higher priority**, you must deconflict — either modify your route (update the OIR) or, if the flight is active, trigger the Nonconforming flow.

This is how DECEA enforces ATM priority: DECEA's USS may create a high-priority Constraint or OIR in your area, and your USS must react within the 10-second conformance window.

5.9 Emergency Handling: Non-Conforming State

Trigger: The drone's telemetry position is outside the OIR's `volumes`. **Required response time:** Transition to `Nonconforming` **within 10 seconds** of detecting the deviation.

5.9.1 Calculate off_nominal_volumes

Compute a volume that covers the drone's current position and the deviation path. This is your "warning zone" for other operators. The exact calculation is at your USS's discretion — it should be large enough to represent the risk area but not unnecessarily large.

5.9.2 Update DSS (within 5 seconds of detection)

Endpoint: `PUT .../dss/v1/operational_intent_references/{entityid}/{ovn}`

```
{
  "extents": [ ... ],
  "state": "Nonconforming",
  "uss_base_url": "https://uss.yourcompany.com/utm",
  "subscription_id": "78ea3fe8-...",
  "flight_type": "BVLOS"
}
```

Note: `key` (OVNs) **may be omitted** in `Nonconforming` state — the DSS does not require proof of airspace awareness during emergencies.

5.9.3 Expose Telemetry Endpoint

In `Nonconforming` state, you must also expose live drone position at: `GET /uss/v1/operational_intents/{entityid}/telemetry`

This allows DECEA and other USSs to monitor the drone's actual position in real time.

5.9.4 Notify Subscribers

After the DSS update, notify all subscribers (as returned in the DSS response) with:

- The new `Nonconforming` state.
- The updated `off_nominal_volumes` included in the `details`.

5.9.5 Recovery

If the drone returns to its original volume:

1. Update the OIR back to `state: Activated`.
2. Clear `off_nominal_volumes` from the details.
3. Notify subscribers of the recovery.
4. Continue normal telemetry serving.

5.10 Emergency Handling: Contingent State

Trigger: The drone has been continuously in `Nonconforming` state for **more than 60 seconds**.

Transition to Contingent

Endpoint: `PUT .../dss/v1/operational_intent_references/{entityid}/{ovn}`

```
{
  "extents": [ ... ],
  "state": "Contingent",
  "uss_base_url": "https://uss.yourcompany.com/utm",
  "subscription_id": "78ea3fe8-...",
  "flight_type": "BVLOS"
}
```

In `Contingent` state:

- The **original** `volumes` **are no longer active** — only `off_nominal_volumes` define the warning zone.
- There is **no recovery path** — the operation must be terminated.
- The ISA **remains active** as long as the drone is still sharing telemetry.
- The telemetry endpoint `GET /uss/v1/operational_intents/{entityid}/telemetry` must remain available.
- Continue notifying subscribers of the Contingent state.

The USS must guide the operator to land the drone and then close the operation (Section 5.11).

5.11 End of Flight: Cleanup

When the flight concludes normally (or after a Contingent state is resolved), the USS must clean up all DSS records.

5.11.1 Delete the ISA

Endpoint: `DELETE /dss/identification_service_areas/{isa_id}/{version}` **Scope:** `rid.service_provider`

Delete the ISA from the Remote ID DSS. The `{version}` comes from the ISA creation response.

Notify the ISA subscribers if the DSS response includes any.

5.11.2 Delete the OIR

Endpoint: `DELETE /dss/v1/operational_intent_references/{entityid}/{ovn}` **Scope:** `utm.strategic_coordination`

Delete the OIR from the DSS entirely. There is no "Completed" state — deletion is how a flight is closed.

> **Note:** Each OIR represents a single flight. If the same drone flies a second mission (e.g., after a battery change), create a **new OIR with a new UUID** for that second flight.

5.11.3 Notify Subscribers of Deletion

The DSS delete response includes a `subscribers` list. Notify each subscriber:

Endpoint (on each subscriber): `POST {subscriber_uss_base_url}/uss/v1/operational_intents`

When sending a deletion notification, **omit the `operational_intent` field** from the body — its absence signals that the OIR has been deleted:

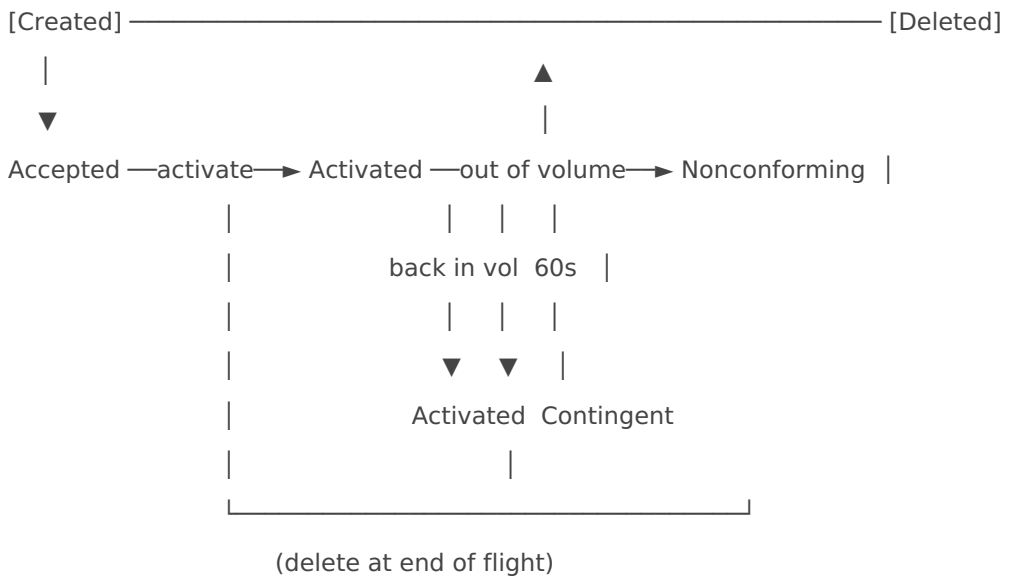
```
{
  "operational_intent_id": "2f8343be-6482-4d1b-a474-16847e01af1e",
  "subscriptions": [
    {
      "subscription_id": "78ea3fe8-...",
      "notification_index": 5
    }
  ]
}
```

5.12 Multi-Volume Operations

A single OIR can contain **multiple `Volume4D` objects** in its `volumes` array. This is the correct approach for complex flight paths. Example structure for a takeoff → route → landing mission:

```
"volumes": [
  {
   "volume": {
     "outline_circle": { "center": {...}, "radius": { "value": 50, "units": "M" }},
     "altitude_lower": { "value": 0, "reference": "W84", "units": "M" },
     "altitude_upper": { "value": 120, "reference": "W84", "units": "M" }
   },
   "time_start": { "value": "2026-07-01T10:00:00Z", "format": "RFC3339" },
   "time_end": { "value": "2026-07-01T10:05:00Z", "format": "RFC3339" }
  },
  {
   "volume": {
     "outline_polygon": { "vertices": [...] },
     "altitude_lower": { "value": 80, "reference": "W84", "units": "M" },
     "altitude_upper": { "value": 120, "reference": "W84", "units": "M" }
   },
   "time_start": { "value": "2026-07-01T10:05:00Z", "format": "RFC3339" },
   "time_end": { "value": "2026-07-01T10:55:00Z", "format": "RFC3339" }
  },
  {
   "volume": {
     "outline_circle": { "center": {...}, "radius": { "value": 50, "units": "M" }},
     "altitude_lower": { "value": 0, "reference": "W84", "units": "M" },
     "altitude_upper": { "value": 120, "reference": "W84", "units": "M" }
   },
   "time_start": { "value": "2026-07-01T10:55:00Z", "format": "RFC3339" },
   "time_end": { "value": "2026-07-01T11:00:00Z", "format": "RFC3339" }
  }
]
```

5.13 Complete State Transition Summary



Revision #1
Created 12 July 2026 20:09:18 by João Pedro Favoretti
Updated 14 July 2026 13:19:47 by João Pedro Favoretti