OTEP — Open Tracking Event Protocol

Übersetzung zur besseren Verständlichkeit — maßgeblich ist die englische Fassung.

> Status: Entwurf v0.1 · Eine offene, herstellerneutrale Spezifikation · Lizenz: offen (siehe §10)

OTEP ist ein offenes, herstellerneutrales Protokoll für den Lebenszyklus jedes verfolgbaren Objekts. Es definiert ein Ereignismodell und ein Statusvokabular, sodass Tracking-Ereignisse aus jeder Quelle — eigene Flottenzustellung, Drittanbieter-Kuriere, Frachtführer-Labels und darüber hinaus — ausgetauscht, verstanden und auf internationale Standards projiziert werden können, ohne für jede Partei neu integrieren zu müssen.

Dieses Dokument ist die Spezifikation: die Struktur, die Felder, die Statustabellen, der Zustands- automat, die Zuordnungen zu externen Standards und die Konformitätsregeln für den Aufbau einer konformen Implementierung. Es ist implementierungsunabhängig — es beschreibt das Protokoll, nicht die internen Abläufe eines bestimmten Herstellers. RFC-2119-Schlüsselwörter (MUST / SHOULD / MAY) sind normativ.

1. Was OTEP löst

Tracking ist fragmentiert: jeder Frachtführer benennt Felder und Statuscodes anders, jeder Zustellkanal meldet in seiner eigenen Form, und die Anbindung an globale Standards bedeutet immer wieder eine erneute Integration. OTEP gibt Produzenten und Konsumenten eine einzige gemeinsame Sprache: ein Produzent gibt OTEP-Ereignisse einmal aus, und jeder OTEP-Konsument versteht sie und kann sie auf den von ihm benötigten Standard projizieren.

2. Designprinzipien

1. Ereignisbasiert (event-sourced). Die Ereignis-Zeitleiste ist die Quelle der Wahrheit; der „aktuelle Status“ ist immer eine Projektion — der Status des jüngsten Ereignisses.
2. Was / Wann / Wo / Warum. Jedes Ereignis ist um diese vier Dimensionen herum aufgebaut — dieselben Dimensionen, die GS1 EPCIS, IATA ONE Record und UN/CEFACT gemeinsam haben — sodass diese Standards Ausgabeprojektionen eines OTEP-Ereignisses sind und keine parallelen Modelle.
3. Quellen ohne Liste sind kein Hindernis. Eine Quelle, die nur einen aktuellen Status (keine Historie) bereitstellt, wird durch Synthese eines Ereignisses pro beobachteter Änderung behandelt (§5).
4. Additiv. OTEP wird neben jeder bestehenden Tracking-API bereitgestellt; seine Einführung erfordert nie eine bahnbrechende Änderung an dem, was Konsumenten bereits verwenden.

3. Die OTEP-Zeitleiste

Eine Zeitleiste ist ein Umschlag, der das verfolgte Objekt und eine geordnete Liste von Ereignissen trägt.

```
{
  "otep_version": "0.1",
  "profile": "parcel",                  // domain profile (§8)
  "subject": {
    "tracking_number": "SR123...",      // ?1 identifier required
    "order_id": 12345,                  // optional
    "package_id": 67890,                // optional
    "external_tracking_number": "1Z...",// optional
    "gs1_sscc": "00...",                // optional — enables EPCIS epcList
    "piece_id": "..."                   // optional — enables ONE Record linkage
  },
  "current_status": "delivered",        // projection of the latest event
  "delivered": true,
  "events": [ /* OTEP events, §4 */ ]
}
```

Das OTEP-Ereignis

```
{
  // WHAT — carried once at the timeline level (subject above)

  // WHEN
  "occurred_at": "2026-06-10T09:30:00-04:00",  // event instant, ISO-8601 with offset
  "recorded_at": "2026-06-10T09:45:23-04:00",  // ingestion instant (optional)
  "time_type": "actual",                        // actual | estimated | scheduled

  // WHERE
  "location": {
    "name": "Toronto Hub",
    "code": "YYZ-2",
    "gln": "0614141000005",                     // GS1 GLN ? EPCIS SGLN
    "lat": 43.6777, "lng": -79.6248,
    "country": "CA"                             // ISO 3166-1 alpha-2
  },

  // WHY / WHAT HAPPENED
  "status_code": "out_for_delivery",            // an OTEP status code (§4)
  "phase": "out_for_delivery",                  // derived from status_code
  "incident_reason": null,                       // an OTEP incident reason (§4) when exception

  // WHO
  "actor": { "type": "driver", "name": "Jane D.", "phone": "+1..." },

  // PROVENANCE
  "source": {
    "type": "self_delivery",                    // self_delivery | third_party_delivery | carrier_label
    "provider_id": null,
    "carrier_code": null,
    "external_event_code": null,                 // your raw status code (preserve it)
    "raw": { /* original payload */ }
  },

  // PROOF
  "pod": {
    "photos": [ { "url": "...", "content_base64": null } ],
    "signature": [ { "url": "...", "content_base64": null } ],
    "recipient": "John Smith"
  }
}
```

Jedes Feld außer `subject`, `occurred_at`, `status_code` und `source` ist optional — partielle Quellen füllen aus, was sie haben. Node-/Hub-Scans setzen `location` auf die scannende Einrichtung. Hochfrequente GPS-Telemetrie ist KEIN OTEP-Ereignis; ein Ereignis wird bei einer Statusänderung oder einem Node-Scan ausgegeben.

4. Vokabular

4.1 Statuscodes

20 kanonische Lebenszyklus-Codes. `phase` ist immer aus dem Code ableitbar.

code	phase	terminal	POD	Bedeutung
`information_submitted`	pre_shipment			Bestellinformationen empfangen
`booking_confirmed`	pre_shipment			Frachtführer/Buchung bestätigt
`awaiting_pickup`	pre_shipment			abholbereit
`out_for_pickup`	pickup			unterwegs zur Abholung
`picked_up`	pickup		✓	beim Versender abgeholt
`pickup_failed`	exception			Abholversuch fehlgeschlagen
`pickup_rescheduled`	exception			Abholung wird erneut versucht
`received`	inbound			in der Einrichtung empfangen
`arrival_scan`	inbound			Ankunftsscan am Node
`in_transit`	transit			in Bewegung

code	phase	terminal	POD	Bedeutung
`package_outbound`	transit			Einrichtung verlassen
`removed_from_route`	exception			von der Route genommen
`route_cancelled`	exception			Route storniert
`out_for_delivery`	out_for_delivery			auf dem Fahrzeug
`delivered`	delivered	✓	✓	an Empfänger zugestellt
`delivery_failed`	exception			Zustellversuch fehlgeschlagen
`delivery_rescheduled`	exception			wird erneut versucht / erneut zugestellt
`return_to_sender`	return	✓		Rücksendung an Ursprung
`rejected_by_recipient`	return	✓		Empfänger verweigert
`cancelled`	return	✓		Bestellung storniert

4.2 Phasen

`pre_shipment` · `preparing` · `pickup` · `inbound` · `in_custody` · `transit` · `out_for_delivery` · `delivered` · `exception` · `return` . (`preparing` und `in_custody` sind für Nicht-Paket-Profile reserviert — §8.)

4.3 Quelltypen & Zeittypen

`source.type` : `self_delivery` · `third_party_delivery` · `carrier_label` . `time_type` : `actual` · `estimated` · `scheduled` (Standard `actual`).

4.4 Vorfallgründe (incident reasons)

Wenn sich ein Ereignis in der Phase `exception` befindet, SHOULD es einen `incident_reason` aus diesem normalisierten Vokabular tragen:

- Carrier: `carrier_damaged_parcel` , `carrier_sorting_error` , `carrier_address_not_found` , `carrier_parcel_lost` , `carrier_not_enough_time` , `carrier_vehicle_issue` , `carrier_capacity_exceeded` , `carrier_mechanical_delay`
- Retailer/shipper: `retailer_cancelled` , `retailer_incorrect_data` , `retailer_not_ready` , `retailer_incorrect_parcel` , `retailer_incorrect_dimensions` , `retailer_packaging_issue`
- Consignee: `consignee_refused` , `consignee_business_closed` , `consignee_not_available` , `consignee_not_home` , `consignee_cancelled` , `consignee_verification_failed` , `consignee_incorrect_address` , `consignee_access_restricted` , `consignee_safe_place_unavailable`
- Customs: `customs_delay` , `customs_documentation` , `customs_duties_unpaid` , `customs_prohibited` , `customs_inspection`
- Force majeure: `weather_delay` , `natural_disaster` , `force_majeure`
- Other: `parcel_being_researched` , `security_issue` , `regulatory_hold` , `unknown`

5. Zustandsautomat & Quellen nur mit Status

Phasen schreiten vorwärts; Ausnahmen unterbrechen und lösen sich wieder auf. Terminalzustände (`delivered` , `return_to_sender` , `rejected_by_recipient` , `cancelled`) schließen das Objekt ab.

```
pre_shipment ? preparing ? pickup ? inbound ? in_custody ? transit ? out_for_delivery ? delivered ?
                                      ????????????? exception ??????????
                                      ????????????? return / cancelled ?
```

Regeln:

- Konsumenten MUST Ereignisse nach `occurred_at` ordnen und MUST ein Eintreffen in falscher Reihenfolge tolerieren.

- Übergänge in einen Terminalzustand sind idempotent; Wiederholungen werden dedupliziert.
- Nach einem Terminalstatus MUST NOT ein Produzent weitere Ereignisse ausgeben, außer einem dokumentierten RMA-/Wiedereröffnungs-Ablauf.
- Eine Quelle, die nur einen aktuellen Status bereitstellt, MUST ein Ereignis pro beobachteter Änderung synthetisieren (mit einem stabilen Dedupe-Schlüssel), anstatt die Historie auszulassen. Mit der Zeit sammeln sich die Momentaufnahmen zu einer Zeitleiste an.

6. Zuordnungen zu externen Standards

Die vier Dimensionen von OTEP entsprechen Feld für Feld den wichtigsten Standards, sodass jeder eine Ausgabeprojektion eines OTEP-Ereignisses ist.

OTEP	GS1 EPCIS 2.0	IATA ONE Record	UN/CEFACT
`occurred_at` (+offset)	`eventTime` + `eventTimeZoneOffset`	`eventDate` + `eventTimeType=Actual`	Event Date/Time
`recorded_at`	`recordTime`	recordedAt	—
subject (sscc/piece)	`epcList` (SSCC URN)	`linkedObject` → Piece/Shipment	Consignment
`location` (gln/lat/lng)	`readPoint` / `bizLocation` (SGLN)	`recordedAtLocation` → Location	Location
`status_code`	`bizStep` + `disposition`	`eventCode`	Transport status code
`actor`	sourceList / extension	Actor / Party	Party
`incident_reason`	disposition / ErrorDeclaration	event remark	Status reason code

Werte pro Code (EPCIS CBV `bizStep` / `disposition`, ONE Record `eventCode`, UN/CEFACT-Code) werden im maschinenlesbaren Codebook veröffentlicht. Über diese internationalen Standards hinaus kann eine OTEP- Zeitleiste auch auf OpenTelemetry-Traces, OGC SensorThings-Beobachtungen und gängige Handelsplattformen (AfterShip, Shopify, Amazon, Walmart, BigCommerce, Magento, WooCommerce, Etsy) projiziert werden.

> Vertrauensgrad: EPCIS CBV-Werte sind stabile Standard-URNs. ONE Record- und UN/CEFACT-Codewerte sind eine bestmögliche Annäherung für die letzte Meile und sollten vor der externen Verwendung gegen die offiziellen Codelisten validiert werden. „Delivered to consignee “ hat keinen exakten CBV- `bizStep` — die nächstliegende Annäherung (`receiving` + `received`) wird verwendet, oder eine Erweiterungs-URN aus dem Benutzervokabular.

7. Die OTEP-API

OTEP wird über eine kleine, nur lesende HTTP-Schnittstelle konsumiert; alle Endpunkte sind öffentlich.

Verb	Path	Liefert
GET	`/api/v1/otep/trackings/{tracking_number}`	die Zeitleiste für eine Sendungsnummer
GET	`/api/v1/otep/trackings/{tracking_number}/events`	nur Ereignisse
POST	`/api/v1/otep/trackings/batch`	viele Sendungsnummern in einem Aufruf
POST	`/api/v1/otep/validate`	Konformitätsprüfung für eine übermittelte Zeitleiste (§9)

Eine GraphQL-Abfrage, die dieselbe Zeitleiste bereitstellt, ist ebenfalls verfügbar.

Inhaltsaushandlung (content negotiation)

Dieselbe Zeitleiste wird in die von Ihnen angeforderte Darstellung serialisiert, über einen `?format=` - Query-Parameter oder ein `Accept` -Profil:

Anfrage	Darstellung
`?format=otep` (default)	native OTEP-Zeitleiste

Anfrage	Darstellung
`?format=epcis`	GS1 EPCIS 2.0 (JSON-LD `ObjectEvent`s)
`?format=onerecord`	IATA ONE Record (`LogisticsEvent`s)
`?format=uncefact`	UN/CEFACT transport status
`?format=otlp`	OpenTelemetry-Traces
`?format=sensorthings`	OGC SensorThings-Beobachtungen
`?format=aftership` | `shopify` | `amazon` | `walmart` | `bigcommerce` | `magento` | `woocommerce` | `etsy`	die Tracking-/Fulfillment-Form der Plattform

Ereignisse, denen in einer Projektion kein Code zugewiesen werden kann, werden übersprungen und gezählt (niemals stillschweigend verworfen).

8. Domänenprofile

OTEP ist ein allgemeines Protokoll, kein Paketprotokoll. Die Protokollebene (Ereignisumschlag, Phasen- Rückgrat, Zustandsautomat) ist universell; konkrete Statuscodes gehören zu einem Profil, das auf der Zeitleiste über `profile` deklariert wird. Die Codes in §4 sind das **`parcel`**-Profil. Andere Domänen — Umzug, Essenslieferung, Lagerung und darüber hinaus — fügen unter ihrem Profil eigene Codesätze hinzu, mit dem Namensraum `otep:<profile>:<code>`, die jeweils auf dasselbe Phasen-Rückgrat abbilden. Das Hinzufügen eines Profils ist eine Erweiterung, keine Protokolländerung.

9. Konformitätsspezifikation (normativ)

Ein Produzent oder Konsument ist OTEP-konform, wenn jedes von ihm ausgegebene oder akzeptierte Ereignis diese Tabellen und Regeln erfüllt.

9.1 Zeitleisten-Umschlag

Feld	Typ	Erf.	Einschränkungen
`otep_version`	string	MUST	semver, z. B. `0.1`
`profile`	string	MUST	ein registriertes Profil
`subject`	object	MUST	§9.2
`current_status`	string|null	SHOULD	ein Statuscode (§4)
`current_phase`	string|null	SHOULD	MUST gleich der Phase von `current_status` sein, wenn beide vorhanden
`delivered`	boolean	SHOULD	`true` genau dann, wenn `current_status` = `delivered`
`events`	array	MUST	Ereignisobjekte (§9.3), nach `occurred_at` sortierbar

9.2 `subject`

Mindestens EINES von `tracking_number` / `order_id` / `package_id` MUST vorhanden sein.

Feld	Typ	Einschränkungen
`tracking_number`	string	nicht leer
`order_id` / `package_id`	integer|null	
`external_tracking_number`	string|null	
`gs1_sscc`	string|null	18 Ziffern
`piece_id`	string|null	

9.3 Ereignisobjekt

#	Feld	Typ	Erf.	Einschränkungen
1	`occurred_at`	string	MUST	ISO-8601 mit Offset
2	`recorded_at`	string|null	SHOULD	ISO-8601
3	`time_type`	string	MAY (default `actual`)	`actual` | `estimated` | `scheduled`
4	`status_code`	string|null	MUST[1]	ein Code in §4.1
5	`phase`	string|null	SHOULD	MUST gleich der Phase von `status_code` sein
6	`incident_reason`	string|null	SHOULD[2]	ein Grund in §4.4
7	`description`	string|null	MAY	menschenlesbar
8	`location`	object|null	MAY	§9.4
9	`actor`	object|null	MAY	`{ type, name?, phone? }`; type ∈ {driver, operator, carrier, system}
10	`source`	object	MUST	§9.5
11	`pod`	object|null	MAY[3]	`{ photos[], signature[], recipient? }`

[1] Ein codiertes Ereignis MUST einen Statuscode aus §4.1 tragen. Ein Rohscan, den Sie noch nicht klassifizieren können, MAY `status_code = null` setzen, MUST aber den nativen Code in `source.external_event_code` bewahren und MUST gezählt werden, niemals verworfen. [2] Ein Ereignis der Phase `exception` SHOULD einen `incident_reason` tragen. [3] Ereignisse mit `status_code` ∈ {`delivered`, `picked_up`} SHOULD ein `pod` tragen.

9.4 `location`

`name` (string) · `code` (string) · `gln` (GS1 GLN, 13 Ziffern) · `lat` / `lng` (WGS-84) · `country` (ISO 3166-1 alpha-2). Alle optional.

9.5 `source`

Feld	Typ	Erf.	Einschränkungen
`type`	string	MUST	`self_delivery` | `third_party_delivery` | `carrier_label`
`provider_id`	integer|null	MAY	
`carrier_code`	string|null	MAY	
`external_event_code`	string|null	SHOULD[4]	Ihr Roh-Statuscode
`raw`	object|null	MAY	ursprüngliche Nutzlast

[4] MUST vorhanden sein, wenn `status_code` null ist, damit der native Code niemals verloren geht.

9.6 Regeln

1. Zeit. `occurred_at` MUST als ISO-8601 parsbar sein. Normalisieren Sie numerische Epochs und `.NET /Date(ms)/` bei der Aufnahme; geben Sie diese Formen nicht aus.
2. Ordnung / Dedup. Konsumenten MUST nach `occurred_at` sortieren, ein Eintreffen in falscher Reihenfolge tolerieren und auf (`subject`, `status_code`, `occurred_at`) deduplizieren.
3. Zustandsautomat. Nach einem Terminalstatus keine weiteren Ereignisse ausgeben, außer einem dokumentierten RMA / Wiedereröffnung.
4. Kein stiller Verlust. Ereignisse, die nicht codiert werden können, MUST gezählt, niemals verworfen werden.
5. Ableitung. `phase`, `current_status`, `current_phase`, `delivered` sind Projektionen — wenn vorhanden, MUST sie mit der Ereignis-Zeitleiste konsistent sein.

9.7 Konformitätsstufen & Validierung

- Level 1 — gibt den Umschlag (§9.1), Ereignisse mit den erforderlichen Feldern (§9.3), gültige Status- codes (§4.1), gültige Phasen (§4.2) aus und beachtet den Zustandsautomaten (§5).

- Level 2 — gibt zusätzlich mindestens eine Projektion auf einen externen Standard (§6) und, wo zutreffend, profilspezifische Codes (§8) aus.

Überprüfen Sie Ihre Ausgabe, indem Sie eine Zeitleiste an `POST /api/v1/otep/validate` senden. Behandeln Sie alle `errors` als blockierend; beheben Sie `warnings`. Ein maschinenlesbares JSON Schema und das vollständige Codebook (jeder Statuscode, jede Phase und jede externe Zuordnung) werden für die Offline-Validierung veröffentlicht.

10. Offenheit & Governance

OTEP ist eine offene Spezifikation, frei für jede Partei zur Implementierung.

- Normativ vs. informativ. Normativ: der Ereignisumschlag, das Phasen-Rückgrat, das Statusvokabular, der Zustandsautomat und die Feldzuordnungen zu externen Standards. Wie ein Implementierer OTEP an seine eigenen internen Systeme bindet, ist seine eigene Angelegenheit und liegt hier außerhalb des Umfangs.
- Stabile Bezeichner. Codes werden als `otep:<profile>:<code>` adressiert, Phasen als `otep:phase:<name>`. Sobald ein Bezeichner in einer veröffentlichten Version publiziert ist, ist seine Bedeutung unveränderlich.
- Versionierung. Semantische Versionierung. Das Hinzufügen von Codes/Profilen ist eine MINOR-Änderung (abwärtskompatibel); die Änderung der Bedeutung eines bestehenden Codes ist eine MAJOR-Änderung und SHOULD vermieden werden. Die Protokollversion reist mit jeder Zeitleiste (`otep_version`).
- Erweiterung. Neue Profile und Codes werden gegen diese Spezifikation vorgeschlagen, statt geforkt zu werden, sodass unabhängige Implementierer konvergieren. Experimentelle Codes MAY ein `x-`-Präfix verwenden (`otep:parcel:x-my_code`), bis sie registriert sind.
- Lizenz. Die Spezifikation soll unter einer offenen Lizenz veröffentlicht werden — noch offen, vorbehaltlich der Freigabe.

11. Hersteller-Interoperabilität — bringen Sie Ihren eigenen Standard mit

OTEP heißt andere Hersteller willkommen, ihren eigenen Tracking-Ereignis-Standard mitzubringen, damit OTEP damit interoperieren kann, in beide Richtungen:

- OTEP → Ihren Standard projizieren. Definieren Sie eine Zuordnung von einer OTEP-Zeitleiste auf Ihr Format, unter Wiederverwendung des OTEP-Statusvokabulars. Es ist eine reine Transformation — Zeitleiste hinein, Ihre Struktur hinaus — sodass die Zuordnung einmal geschrieben wird und jeder OTEP-Produzent Ihr Format ausgeben kann.
- Ihren Standard → OTEP abbilden. Stellen Sie eine Querverbindung von Ihrem Statusvokabular auf die OTEP- Codes (§4) bereit, plus, falls nötig, ein Profil (§8). Ihre Rohcodes werden in `source.external_event_code` bewahrt; nicht zugeordnete Codes werden gezählt, niemals verworfen.

Selbst wenn Sie OTEP nicht direkt übernehmen können, sind Sie willkommen, einen einzigen normalisierten `otep_status` zu Ihren eigenen API-Antworten hinzuzufügen und Ihre Tracking-Ereignis-Codes für die Querverbindungs-Zuordnung zu teilen. Schlagen Sie Zuordnungen gegen diese Spezifikation vor (statt zu forken), damit Implementierer konvergieren; neue Formate fügen sich in dieselbe `?format=`-Inhaltsaushandlung ein und brechen niemals einen bestehenden Konsumenten.