

OTEP — Open Tracking Event Protocol

Traduction de courtoisie — la version anglaise fait autorité.

Statut : Brouillon v0.1 · Une spécification ouverte et indépendante de tout fournisseur · Licence : ouverte (voir §10)

OTEP est un protocole ouvert et indépendant de tout fournisseur pour le cycle de vie de tout sujet traçable. Il définit un seul modèle d'événement et un seul vocabulaire de statut afin que les événements de suivi provenant de n'importe quelle source — livraison par flotte propre, transporteurs tiers, étiquettes de transporteur et au-delà — puissent être échangés, compris et projetés vers les standards internationaux sans réintégration pour chaque partie.

Ce document est la spécification : la structure, les champs, les tables de statut, la machine à états, les correspondances avec les standards externes et les règles de conformité pour construire une implémentation conforme. Il est indépendant de toute implémentation — il décrit le protocole, et non les rouages internes d'un fournisseur particulier. Les mots-clés RFC-2119 (MUST / SHOULD / MAY) sont normatifs.

1. Ce que résout OTEP

Le suivi est fragmenté : chaque transporteur nomme les champs et les codes de statut différemment, chaque canal de livraison rapporte selon sa propre forme, et la connexion aux standards mondiaux signifie réintégrer encore et encore. OTEP donne aux producteurs et aux consommateurs un langage commun unique : un producteur émet des événements OTEP une fois, et chaque consommateur OTEP les comprend et peut les projeter vers le standard dont il a besoin.

2. Principes de conception

1. Orienté événements. La chronologie des événements est la source de vérité ; le « statut actuel » est toujours une projection — le statut du dernier événement.
2. Quoi / Quand / Où / Pourquoi. Chaque événement est structuré autour de ces quatre dimensions — les mêmes dimensions partagées par GS1 EPCIS, IATA ONE Record et UN/CEFACT — de sorte que ces standards sont des projections de sortie d'un événement OTEP plutôt que des modèles parallèles.
3. Les sources sans liste ne sont pas un obstacle. Une source qui n'expose qu'un statut actuel (sans historique) est gérée en synthétisant un événement par changement observé (§5).
4. Additif. OTEP est exposé en parallèle de toute API de suivi existante ; son adoption n'exige jamais un changement cassant pour ce que les consommateurs utilisent déjà.

3. La chronologie OTEP

Une chronologie est une enveloppe portant le sujet suivi et une liste ordonnée d'événements.

```
{
  "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_sccc": "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 */ ]
}
```

L'événement OTEP

```
{
  // 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"
  }
}
```

Chaque champ sauf `subject` , `occurred_at` , `status_code` et `source` est optionnel — les sources partielles remplissent ce qu'elles ont. Les scans de nœud/hub définissent `location` sur l'installation de scan. La télémétrie GPS à haute fréquence n'est PAS un événement OTEP ; un événement est émis lors d'un changement de statut ou d'un scan de nœud.

4. Vocabulaire

4.1 Codes de statut

20 codes canoniques de cycle de vie. `phase` est toujours dérivable du code.

code	phase	terminal	POD	signification
information_submitted	pre_shipment			informations de commande reçues
booking_confirmed	pre_shipment			transporteur/réservation confirmé
awaiting_pickup	pre_shipment			prêt pour la collecte
out_for_pickup	pickup			en route pour la collecte
picked_up	pickup		✓	collecté chez l'expéditeur
pickup_failed	exception			tentative de collecte échouée
pickup_rescheduled	exception			la collecte sera retentée
received	inbound			reçu à l'installation
arrival_scan	inbound			scan d'arrivée au nœud
in_transit	transit			en mouvement

code	phase	terminal	POD	signification
package_outbound	transit			a quitté l'installation
removed_from_route	exception			retiré de la tournée
route_cancelled	exception			tournée annulée
out_for_delivery	out_for_delivery			dans le véhicule
delivered	delivered	✓	✓	livré au destinataire
delivery_failed	exception			tentative de livraison échouée
delivery_rescheduled	exception			sera retentée / relivrée
return_to_sender	return	✓		retour à l'origine
rejected_by_recipient	return	✓		destinataire a refusé
cancelled	return	✓		commande annulée

4.2 Phases

pre_shipment · preparing · pickup · inbound · in_custody · transit · out_for_delivery · delivered · exception · return . (preparing et in_custody sont réservés aux profils non-colis — §8.)

4.3 Types de source et types de temps

source.type : self_delivery · third_party_delivery · carrier_label . time_type : actual · estimated · scheduled (par défaut actual).

4.4 Motifs d'incident

Lorsqu'un événement est dans la phase exception , il SHOULD porter un incident_reason issu de ce vocabulaire normalisé :

- Transporteur: 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
- Détaillant/expéditeur: retailer_cancelled , retailer_incorrect_data , retailer_not_ready , retailer_incorrect_parcel , retailer_incorrect_dimensions , retailer_packaging_issue
- Destinataire: 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
- Douanes: customs_delay , customs_documentation , customs_duties_unpaid , customs_prohibited , customs_inspection
- Force majeure: weather_delay , natural_disaster , force_majeure
- Autre: parcel_being_researched , security_issue , regulatory_hold , unknown

5. Machine à états et sources à statut unique

Les phases avancent vers l'avant ; les exceptions interrompent et se résolvent en retour. Les états terminaux (delivered , return_to_sender , rejected_by_recipient , cancelled) clôturent le sujet.

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

Règles :

- Les consommateurs MUST ordonner les événements par occurred_at et MUST tolérer une arrivée dans le désordre.

- Les transitions vers un état terminal sont idempotentes ; les répétitions sont dédoublées.
- Après un statut terminal, un producteur MUST NOT émettre d'autres événements sauf un flux documenté de RMA / réouverture.
- Une source n'exposant qu'un statut actuel MUST synthétiser un événement par changement observé (avec une clé de déduplication stable) plutôt que d'omettre l'historique. Au fil du temps, les instantanés s'accumulent en une chronologie.

6. Correspondances avec les standards externes

Les quatre dimensions d'OTEP s'alignent champ par champ avec les principaux standards, de sorte que chacun est une projection de sortie d'un événement OTEP.

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

Les valeurs par code (EPCIS CBV `bizStep / disposition`, ONE Record `eventCode`, code UN/CEFACT) sont publiées dans le codebook lisible par machine. Au-delà de ces standards internationaux, une chronologie OTEP peut aussi être projetée vers des traces OpenTelemetry, des observations OGC SensorThings et des plateformes de commerce courantes (AfterShip, Shopify, Amazon, Walmart, BigCommerce, Magento, WooCommerce, Etsy).

Confiance : les valeurs EPCIS CBV sont des URN standard stables. Les valeurs de code ONE Record et UN/CEFACT sont les plus adaptées pour le dernier kilomètre et devraient être validées contre les listes de codes officielles avant tout usage externe. « Delivered to consignee » n'a pas de bizStep CBV exact — la correspondance la plus proche (`receiving + received`) est utilisée, ou une URN d'extension de vocabulaire utilisateur.

7. L'API OTEP

OTEP est consommé via une petite surface HTTP en lecture seule ; tous les points de terminaison sont publics.

Verbe	Chemin	Retourne
GET	<code>/api/v1/otep/trackings/{tracking_number}</code>	la chronologie pour un numéro de suivi
GET	<code>/api/v1/otep/trackings/{tracking_number}/events</code>	les événements uniquement
POST	<code>/api/v1/otep/trackings/batch</code>	plusieurs numéros de suivi en un seul appel
GET	<code>/api/v1/otep/trackings/{tracking_number}?view=public</code>	vue publique sécurisée : sans nom de chauffeur ni identifiant interne, limitée à l'entreprise du domaine, numéros de déménagement et d'entrepasage acceptés
GET	<code>/api/v1/client/otep/trackings/{identifiant}</code>	(jeton API) chronologie et détails de votre propre envoi : colis, déménagement, entrepasage
GET	<code>/api/v1/client/otep/locker-storage-rentals/{id}</code>	(jeton API) chronologie d'un dépôt en consigne, par id d'enregistrement
GET	<code>/oteptrack</code>	page publique de suivi : colis, déménagement, entrepasage ; détails après vérification
POST	<code>/api/v1/otep/validate</code>	vérification de conformité pour une chronologie postée (§9)

Une requête GraphQL exposant la même chronologie est également disponible.

Négociation de contenu

La même chronologie est sérialisée dans la représentation que vous demandez, via un paramètre de requête `?format=` ou un profil `Accept` :

Requête	Représentation
<code>?format=otep</code> (par défaut)	chronologie OTEP native
<code>?format=epcis</code>	GS1 EPCIS 2.0 (<code>ObjectEvent</code> s JSON-LD)
<code>?format=onerecord</code>	IATA ONE Record (<code>LogisticsEvent</code> s)
<code>?format=uncefact</code>	statut de transport UN/CEFACT
<code>?format=otlp</code>	traces OpenTelemetry
<code>?format=sensorthings</code>	observations OGC SensorThings
<code>?format=aftership</code> <code>shopify</code> <code>amazon</code> <code>walmart</code> <code>bigcommerce</code> <code>magento</code> <code>woocommerce</code> <code>etsy</code>	la forme de suivi/exécution de la plateforme

Les événements qui ne peuvent pas se voir attribuer un code dans une projection sont ignorés et comptabilisés (jamais abandonnés silencieusement).

8. Profils de domaine

OTEP est un protocole général, pas un protocole de colis. La couche protocole (enveloppe d'événement, colonne vertébrale des phases, machine à états) est universelle ; les codes de statut concrets appartiennent à un profil déclaré sur la chronologie via `profile`. Les codes du §4 sont le profil **parcel**. D'autres domaines — déménagement, livraison de nourriture, stockage et au-delà — ajoutent leurs propres jeux de codes sous leur profil, avec l'espace de noms `otep:<profile>:<code>`, chacun se rattachant à la même colonne vertébrale des phases. Ajouter un profil est une extension, pas un changement de protocole.

9. Spécification de conformité (normative)

Un producteur ou un consommateur est conforme à OTEP lorsque chaque événement qu'il émet ou accepte satisfait ces tables et règles.

9.1 Enveloppe de chronologie

Champ	Type	Req.	Contraintes
<code>otep_version</code>	string	MUST	semver, p. ex. <code>0.1</code>
<code>profile</code>	string	MUST	un profil enregistré
<code>subject</code>	object	MUST	§9.2
<code>current_status</code>	string null	SHOULD	un code de statut (§4)
<code>current_phase</code>	string null	SHOULD	MUST être égal à la phase de <code>current_status</code> si les deux sont présents
<code>delivered</code>	boolean	SHOULD	<code>true</code> ssi <code>current_status = delivered</code>
<code>events</code>	array	MUST	objets événement (§9.3), ordonnables par <code>occurred_at</code>

9.2 **subject**

Au moins UN de `tracking_number` / `order_id` / `package_id` MUST être présent.

Champ	Type	Contraintes
<code>tracking_number</code>	string	non vide
<code>order_id</code> / <code>package_id</code>	integer null	
<code>external_tracking_number</code>	string null	
<code>gs1_sscc</code>	string null	18 chiffres

Champ	Type	Contraintes
<code>piece_id</code>	string null	

9.3 Objet événement

#	Champ	Type	Req.	Contraintes
1	<code>occurred_at</code>	string	MUST	ISO-8601 avec offset
2	<code>recorded_at</code>	string null	SHOULD	ISO-8601
3	<code>time_type</code>	string	MAY (par défaut <code>actual</code>)	<code>actual</code> <code>estimated</code> <code>scheduled</code>
4	<code>status_code</code>	string null	MUST ¹	un code du §4.1
5	<code>phase</code>	string null	SHOULD	MUST être égal à la phase de <code>status_code</code>
6	<code>incident_reason</code>	string null	SHOULD ²	un motif du §4.4
7	<code>description</code>	string null	MAY	lisible par l'humain
8	<code>location</code>	object null	MAY	§9.4
9	<code>actor</code>	object null	MAY	{ <code>type</code> , <code>name?</code> , <code>phone?</code> } ; <code>type</code> ∈ { <code>driver</code> , <code>operator</code> , <code>carrier</code> , <code>system</code> }
10	<code>source</code>	object	MUST	§9.5
11	<code>pod</code>	object null	MAY ³	{ <code>photos[]</code> , <code>signature[]</code> , <code>recipient?</code> }

¹ Un événement codé MUST porter un code de statut du §4.1. Un scan brut que vous ne pouvez pas encore classer MAY définir `status_code = null`, mais MUST préserver le code natif dans `source.external_event_code` et MUST être comptabilisé, jamais abandonné. ² Un événement de phase `exception` SHOULD porter un `incident_reason`. ³ Les événements avec `status_code` ∈ {`delivered`, `picked_up`} SHOULD porter un `pod`.

9.4 location

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

9.5 source

Champ	Type	Req.	Contraintes
<code>type</code>	string	MUST	<code>self_delivery</code> <code>third_party_delivery</code> <code>carrier_label</code>
<code>provider_id</code>	integer null	MAY	
<code>carrier_code</code>	string null	MAY	
<code>external_event_code</code>	string null	SHOULD ⁴	votre code de statut brut
<code>raw</code>	object null	MAY	charge utile d'origine

⁴ MUST être présent quand `status_code` est null, afin que le code natif ne soit jamais perdu.

9.6 Règles

1. Temps. `occurred_at` MUST s'analyser comme ISO-8601. Normalisez les époques numériques et `.NET /Date(ms)/` à l'ingestion ; n'émettez pas ces formes.
2. Ordre / dédup. Les consommateurs MUST trier par `occurred_at`, tolérer une arrivée dans le désordre, et dédupliquer sur (`subject`, `status_code`, `occurred_at`).
3. Machine à états. Après un statut terminal, n'émettez pas d'autres événements sauf un RMA / réouverture documenté.
4. Aucune perte silencieuse. Les événements qui ne peuvent pas être codés MUST être comptabilisés, jamais abandonnés.
5. Dérivation. `phase`, `current_status`, `current_phase`, `delivered` sont des projections — s'ils sont présents, ils MUST être cohérents avec la chronologie des événements.

9.7 Niveaux de conformité et validation

- Niveau 1 — émet l'enveloppe (§9.1), des événements avec les champs requis (§9.3), des codes de statut valides (§4.1), des phases valides (§4.2), et respecte la machine à états (§5).
- Niveau 2 — émet en plus au moins une projection vers un standard externe (§6) et, le cas échéant, des codes spécifiques au profil (§8).

Vérifiez votre sortie en POSTant une chronologie à `POST /api/v1/otep/validate`. Traitez toute `errors` comme bloquante ; corrigez les `warnings`. Un JSON Schema lisible par machine et le codebook complet (chaque code de statut, phase et correspondance externe) sont publiés pour une validation hors ligne.

10. Ouverture et gouvernance

OTEP est une spécification ouverte, libre d'implémentation pour toute partie.

- Normatif vs informatif. Normatif : l'enveloppe d'événement, la colonne vertébrale des phases, le vocabulaire de statut, la machine à états et les correspondances de champs avec les standards externes. La manière dont un implémenteur relie OTEP à ses propres systèmes internes le concerne et sort du cadre de ce document.
- Identifiants stables. Les codes sont adressés sous la forme `otep:<profile>:<code>`, les phases sous la forme `otep:phase:<name>`. Une fois publiée dans une version diffusée, la signification d'un identifiant est immuable.
- Versionnage. Versionnage sémantique. Ajouter des codes/profils est un changement MINEUR (rétrocompatible) ; modifier la signification d'un code existant est un changement MAJEUR et SHOULD être évité. La version du protocole voyage avec chaque chronologie (`otep_version`).
- Extension. Les nouveaux profils et codes sont proposés par rapport à cette spécification plutôt que forkés, afin que les implémenteurs indépendants convergent. Les codes expérimentaux MAY utiliser un préfixe `x-` (`otep:parcel:x-my_code`) jusqu'à leur enregistrement.
- Licence. La spécification est destinée à être publiée sous une licence ouverte — à déterminer, en attente de validation.

11. Interopérabilité entre fournisseurs — apportez votre propre standard

OTEP invite les autres fournisseurs à apporter leur propre standard d'événements de suivi afin qu'OTEP puisse interopérer avec lui, dans les deux directions :

- Projeter OTEP → votre standard. Définissez une correspondance d'une chronologie OTEP vers votre format, en réutilisant le vocabulaire de statut OTEP. C'est une transformation pure — chronologie en entrée, votre structure en sortie — de sorte que la correspondance est écrite une fois et chaque producteur OTEP peut émettre votre format.
- Faire correspondre votre standard → OTEP. Fournissez une table de correspondance de votre vocabulaire de statut vers les codes OTEP (§4) plus, si nécessaire, un profil (§8). Vos codes bruts sont préservés dans `source.external_event_code` ; les codes non mis en correspondance sont comptabilisés, jamais abandonnés.

Même si vous ne pouvez pas adopter OTEP directement, vous êtes invité à ajouter un seul `otep_status` normalisé à vos propres réponses d'API et à partager vos codes d'événements de suivi pour la mise en correspondance. Proposez les correspondances par rapport à cette spécification (plutôt que de forker) pour que les implémenteurs convergent ; les nouveaux formats se branchent dans la même négociation de contenu `?format=` et ne cassent jamais un consommateur existant.