



ASSESSORADU DE SOS TRASPORTOS
ASSESSORATO DEI TRASPORTI

Evoluzione e gestione del Sistema Account Based Ticketing per il Trasporto Pubblico Locale
in Regione Autonoma della Sardegna

CUI S80002870923202500621 - CUP E71C25000020002

Allegato 3

Specifiche di Interfacciamento ai servizi di Sardegna Mobilità



Specifiche di Interfacciamento ai Servizi API

1. Introduzione

Il presente documento definisce le specifiche tecniche per l'interfacciamento con i servizi API esposti dalla piattaforma. Queste API forniscono l'accesso a un insieme di funzionalità relative alla mobilità, includendo la ricerca di informazioni sul trasporto pubblico, la pianificazione di viaggi, la consultazione di orari statici e in tempo reale, la gestione di dati tariffari, l'accesso a servizi geospaziali e ai dati Netex.

Tutti i parametri delle richieste, se non diversamente specificato come parametri di percorso (path parameters), devono essere trasmessi come parametri di query (query string parameters).

2. Modalità di Accesso e Autenticazione

2.1 URL Base

L'URL base per l'accesso alle API è il seguente: <https://api.sardegnamobilita.it>

Per l'accesso alle API Netex l'URL base è invece il seguente: <https://rap.sardegnamobilita.it>

Tutti i percorsi API specificati di seguito sono relativi a tali URL.

Esempio: Se l'URL base è <https://api.sardegnamobilita.it>, l'endpoint `/v1/stops/search/{query}` sarà accessibile a <https://api.sardegnamobilita.it/v1/stops/search/{query}>.

2.2 Autenticazione

L'accesso alla maggior parte delle API è protetto e richiede un'autenticazione basata su chiave API e verifica dell'origine della richiesta. I client devono includere i seguenti header HTTP nelle loro richieste:

- **x-api-key:** Chiave API univoca fornita al client autorizzato.
- **Origin:** Header standard HTTP che indica l'origine della richiesta (es. <https://www.clientdomain.com>).
- **x-origin:** Header personalizzato che deve essere inviato, specialmente in contesti di integrazione tramite iframe. Indica l'origine della pagina principale che ospita l'iframe.

Il sistema verificherà la validità della x-api-key e l'ammissibilità degli Origin e x-origin specificati per il servizio API invocato, restituendo in risposta gli opportuni header CORS basati sull'origine della richiesta verificata.

2.3 Formato dei Dati

- **Richieste:**
 - Parametri per richieste GET: forniti tramite la query string.



- Corpo per richieste POST: tipicamente in formato application/json o application/x-www-form-urlencoded. Il content type specifico è indicato per ogni endpoint POST.
- Se non diversamente specificato, il parametro {routerId} è “sardegna”.

- **Risposte:**

- Il formato delle risposte è generalmente application/json con Content-Type: application/json. Eccezioni sono indicate.
- In caso di successo, il codice di stato HTTP sarà 200 OK o simile (es. 201 Created).
- In caso di errore, verranno restituiti codici di stato HTTP appropriati (es. 400 Bad Request, 401 Unauthorized, 403 Forbidden, 404 Not Found, 500 Internal Server Error) e un corpo JSON con dettagli dell'errore ove possibile.

2.4 Gestione CORS (Cross-Origin Resource Sharing)

Il server API implementa CORS per consentire richieste da origini differenti. Le risposte includeranno i seguenti header CORS, a seconda della configurazione e dell'autorizzazione:

- Access-Control-Allow-Origin: Indica le origini autorizzate a effettuare la richiesta.
- Access-Control-Allow-Methods: Metodi HTTP permessi (es. GET, POST, OPTIONS).
- Access-Control-Allow-Headers: Header permessi nella richiesta (es. x-api-key, x-origin, Content-Type).

Le richieste OPTIONS (pre-flight) sono gestite per negoziare i permessi CORS.

3. Descrizione dei Servizi API (Endpoint)

Di seguito è riportato l'elenco degli endpoint API disponibili, raggruppati per funzionalità.

3.1 Gestione Fermate

- **GET /v1/stops/search/{query}**
 - Descrizione: Alias di /v1/search/{query} (specifico per ricerca fermate, ma implementazione identica).
 - Parametri Path:
 - query: (Stringa, obbligatorio, min 3 caratteri, max 50 caratteri) Termine di ricerca.
 - Risposta (200 OK): JSON contenente un array data di fermate e un array dataNom di risultati da Nominatim.
 - {
 - "data": [



ASSESSORADU DE SOS TRASPORTOS
ASSESSORATO DEI TRASPORTI

- {
- "id": "string", // stop_id
- "description": "string", // stop_name
- "lat": "number",
- "lon": "number",
- "type": "string", // stop_type
- "comune": "string" // stop_comune
- }
-],
- "dataNom": [- // Oggetti risultato da Nominatim
-]
- }

Note: Se il termine di ricerca non è valido o è troppo corto/lungo, restituisce {"data":{}}.

- **GET /v1/stops/info/{stopId}**

- Descrizione: Richiede informazioni dettagliate su una specifica fermata dall'anagrafica.
- Parametri Path:
 - stopId: (Stringa, obbligatorio) Identificativo della fermata (case-insensitive).
- Risposta (200 OK): JSON con i dettagli della fermata.
- // Esempio di struttura, i campi esatti dipendono dalla tabella 'anagrafica.anagrafica_stops'
- {
- "stop_id": "string",
- "stop_name": "string",
- "stop_lat": "number",
- "stop_lon": "number",



ASSESSORADU DE SOS TRASPORTOS
ASSESSORATO DEI TRASPORTI

- "id_zona": "string", // Alias di zone_id
- // ...altri campi dall'anagrafica
- }

- Risposta (404 Not Found): Se la fermata non è trovata.

```
{ "success": false, "message": "Fermata non trovata" }
```

- **GET /v1/stops/services/{stopId}**

- Descrizione: Richiede i servizi (linee e direzioni) disponibili presso una specifica fermata.
- Parametri Path:
 - stopId: (Stringa, obbligatorio) Identificativo della fermata (case-insensitive).
- Risposta (200 OK): JSON array di oggetti, ognuno rappresentante un servizio/linea.
- [
○ {
○ "stop_id": "string",
○ "stop_name": "string",
○ "stop_lat": "number",
○ "stop_lon": "number",
○ "agency_id": "string",
○ "agency_name": "string",
○ "route_id": "string",
○ "route_short_name": "string",
○ "route_long_name": "string",
○ "direction_stop_id": "string",
○ "direction_stop_name": "string",
○ "service_ids": ["string"] // Array di service_id (calendari) attivi per questa combinazione



ASSESSORADU DE SOS TRASPORTOS
ASSESSORATO DEI TRASPORTI

- }
- // ...altri servizi
-]
- Risposta (404 Not Found): Se la fermata non è trovata o non ha servizi associati.

- **GET /v1/stops/nearby**

- Descrizione: Richiede le fermate vicine a una data coordinata geografica, ordinate per distanza (massimo 10 risultati).
- Parametri Query:
 - lon: (Numero, obbligatorio) Longitudine del punto di ricerca.
 - lat: (Numero, obbligatorio) Latitudine del punto di ricerca.
 - dist: (Intero, obbligatorio) Distanza massima in metri per la ricerca.
- Risposta (200 OK): JSON array di oggetti fermata, con un campo aggiuntivo distanza.
- [
- {
- // Campi dalla tabella 'anagrafica.anagrafica_stops'
- "stop_id": "string",
- "stop_name": "string",
- // ...
- "distanza": "number" // Distanza in metri dalla coordinata fornita
- }
- // ...altre fermate vicine
-]
- Risposta (400 Bad Request): Se i parametri non sono validi.

- **GET /v1/stops/getSchedules/{stopId}**

- Descrizione: Richiede i prossimi orari programmati (statici GTFS, massimo 5) per una specifica fermata.



ASSESSORADU DE SOS TRASPORTOS
ASSESSORATO DEI TRASPORTI

- Parametri Path:
 - stopId: (Stringa, obbligatorio) Identificativo della fermata.
- Risposta (200 OK): JSON array di schedulazioni. Ogni oggetto rappresenta un passaggio.
- // Esempio struttura, basata su 'gtfs.stop_routes_times_view'
- [
 - {
 - "agency_id": "string",
 - "route_id": "string",
 - "trip_id": "string",
 - "stop_id": "string",
 - "stop_name": "string",
 - "departure_time": "string" // HH:MM:SS
 - // ...altri campi dalla vista
 - }
-]

- **GET /v1/stops/getSchedulesRT/{stopId}**
- **GET /v1/stops/getSchedulesRT/{comune}/{stopId}**
- **GET /v1/stops/getSchedulesRT/{comune}/{tipo}/{stopId}**
 - Descrizione: Richiede i prossimi orari (massimo 10) per una fermata, combinando dati statici GTFS con dati in tempo reale (RT), se disponibili.
 - Parametri Path:
 - stopId: (Stringa, obbligatorio) Identificativo della fermata.
 - comune (Stringa, opzionale): Filtra per comune (case-insensitive, es. "CAGLIARI").
 - tipo (Stringa, opzionale): Filtra per tipo (Fermata, Fermata Metropolitana, Stazione, Porto, Aeroporto).
 - Parametri Query (alternativa a path param comune):



- comune (Stringa, opzionale): Filtra per comune.
- Risposta (200 OK): JSON array di schedulazioni. Gli oggetti possono avere campi aggiuntivi per RT:
 - [
 - {
 - // Campi da 'gtfs.stop_routes_times_view'
 - "agency_id": "string",
 - "trip_id": "string",
 - "stop_id": "string",
 - "departure_time": "string", // HH:MM:SS (schedulato)
 - // Campi RT aggiuntivi (se disponibili)
 - "departure_time_rt": "string", // HH:MM:SS (reale o previsto)
 - "delay": "number" // Ritardo in secondi (da GTFS-RT)
 - // ...altri campi
 - }
 -]

3.2 Tariffe

- **GET /v1/tariffe/zona/{lat}/{lon}**
 - Descrizione: Determina la zona tariffaria basata su coordinate geografiche (latitudine e longitudine).
 - Parametri Path:
 - lat: (Numero, obbligatorio) Latitudine.
 - lon: (Numero, obbligatorio) Longitudine.
 - Risposta (200 OK): JSON con ID e nome della zona.
 - {
 - "ID": "string", // id_zona
 - "NOME": "string" // zona_tariffaria



- }

Oppure {} se nessuna zona trovata.

- **GET /v1/tariffe/zona/{coordinate}**

- Descrizione: Determina la zona tariffaria basata su una stringa di coordinate.
- Parametri Path:
 - coordinate: (Stringa, obbligatorio) Stringa "lat,lon".
- Risposta (200 OK): Come sopra.

- **GET /v1/tariffe/tariffe/{origine}/{destinazione}**

- Descrizione: Calcola le tariffe tra una zona di origine e una di destinazione.
- Parametri Path:
 - origine: (Stringa, obbligatorio) ID della zona di origine.
 - destinazione: (Stringa, obbligatorio) ID della zona di destinazione.
- Risposta (200 OK): JSON con i costi per tipo di biglietto, tratta e distanza.
- {
- "BIGLIETTO_CORSA_SEMPLICE": "string", // Costo formattato es. "1,30"
- // ...altri tipi di biglietto (codice)
- "tratta": "string",
- "distanza": "number"
- }

Oppure {} se nessuna tariffa trovata.

- **GET /v1/tariffe/tariffe/{orig_dest}**

- Descrizione: Calcola le tariffe, con origine e destinazione in una singola stringa.
- Parametri Path:
 - orig_dest: (Stringa, obbligatorio) Stringa "id_origine,id_destinazione".
- Risposta (200 OK): Come sopra.



3.3 Pianificazione Viaggi (OpenTripPlanner - OTP)

- **GET /v1/routers/{routerId}/getShape/{tripId}**

- Descrizione: Richiede la geometria (shape) di uno specifico viaggio (trip) GTFS.
- Parametri Path:
 - routerId: (Stringa) Identificativo del router.
 - tripId: (Stringa, obbligatorio) Identificativo del viaggio (trip_id GTFS).
- Risposta (200 OK): JSON array contenente un singolo oggetto con la geometria (es. GeoJSON) dalla tabella gtfs.shapes_geometry.

- **GET /v1/routers/{routerId}/plan**

- Descrizione: Pianifica un viaggio utilizzando il motore OTP.
- Parametri Path:
 - routerId: (Stringa, obbligatorio) Identificativo del router (per selezionare la configurazione OTP corretta).
- Parametri Query: Standard OTP, i principali sono:
 - fromPlace: (Stringa, obbligatorio) "lat,lon" del punto di partenza.
 - toPlace: (Stringa, obbligatorio) "lat,lon" del punto di arrivo.
 - mode: (Stringa, obbligatorio) Modalità di trasporto (es. TRANSIT,WALK, CAR, BICYCLE, AIRPLANE, FERRY). Il sistema adatta questo parametro in base alla localizzazione (Sardegna/extra-Sardegna).
 - date: (Stringa, opzionale) Data del viaggio "MM-DD-YYYY".
 - time: (Stringa, opzionale) Ora del viaggio "HH:mm".
 - arriveBy: (Booleano, opzionale, default false) Se true, date e time si riferiscono all'arrivo.
 - numItineraries: (Intero, opzionale, default da configurazione) Numero di itinerari da restituire.
 - optimize: (Stringa, opzionale, default da configurazione) Criterio di ottimizzazione (es. QUICK, TRANSFERS).
 - maxWalkDistance: (Numero, opzionale, default da configurazione) Distanza massima a piedi in metri.



ASSESSORADU DE SOS TRASPORTOS
ASSESSORATO DEI TRASPORTI

- walkReluctance: (Numero, opzionale, default da configurazione) Fattore di "riluttanza" a camminare.
- Altri parametri OTP standard...
- Risposta (200 OK, Content-Type: application/json): JSON con il piano di viaggio restituito da OTP, con eventuale post-processing.
- {
- "plan": {
- "date": "number", // Timestamp
- "from": { /* ... */ },
- "to": { /* ... */ },
- "itineraries": [
- {
- "duration": "number", // secondi
- "startTime": "number", // Timestamp
- "endTime": "number", // Timestamp
- "legs": [
- {
- "mode": "string", // WALK, BUS, RAIL, AIRPLANE, etc.
- "startTime": "number",
- "endTime": "number",
- "distance": "number", // metri
- // ...altri dettagli della leg (route, trip, from, to, geometry)
- }
-]
- // ...altri dettagli itinerario
- }
-]
- }



- // ...altre informazioni (requestParameters, debugOutput, error)
- }

- Risposta (in caso di errore OTP o di chiamata): {} o JSON di errore da OTP.

- **GET /v1/routers/{routerId}/index/trips/{tripId}/stops**

- Descrizione: Recupera l'elenco delle fermate per un specifico tripId indicizzato da OTP.
- Parametri Path:
 - routerId: (Stringa, obbligatorio) Identificativo del router.
 - tripId: (Stringa, obbligatorio) Identificativo del viaggio OTP.
- Risposta (200 OK, Content-Type: application/json): JSON array di fermate restituito da OTP.

3.4 Servizi Geospaziali (GeoServer)

- **GET /v1/geoserver/sardegna/wms**

- Descrizione: Proxy per richieste WMS (Web Map Service) a un'istanza GeoServer configurata.
- Parametri Query: Parametri standard WMS, ad esempio:
 - SERVICE=WMS (Obbligatorio)
 - REQUEST (Obbligatorio, es. GetMap, GetCapabilities, GetFeatureInfo)
 - VERSION (Obbligatorio, es. 1.1.1, 1.3.0)
 - LAYERS (Stringa, per GetMap, GetFeatureInfo)
 - STYLES (Stringa, opzionale)
 - CRS o SRS (Stringa, per GetMap)
 - BBOX (Stringa, per GetMap)
 - WIDTH (Intero, per GetMap)
 - HEIGHT (Intero, per GetMap)
 - FORMAT (Stringa, es. image/png, application/json per GetFeatureInfo)
 - Altri parametri WMS specifici per la richiesta...
- Risposta: Dipende dalla richiesta WMS.



ASSESSORADU DE SOS TRASPORTOS
ASSESSORATO DEI TRASPORTI

- GetMap: Immagine (es. Content-Type: image/png).
- GetCapabilities: Documento XML (Content-Type: application/vnd.ogc.wms_xml o text/xml).
- GetFeatureInfo: Dipende dal INFO_FORMAT richiesto (es. text/html, application/json).

3.5 Servizi Netex

- **GET /netex/api/v1/xsdzip**
 - Descrizione: Restituisce gli schemi XSD per il formato Netex, compressi in un file ZIP.
 - Risposta (Content-Type: application/zip o simile): File ZIP contenente gli XSD.
- **GET /netex/api/v1/convertedNetex**
 - Descrizione: Fornisce informazioni su dati Netex che sono stati convertiti.
 - Parametri Query: Probabilmente version, datasetId per filtrare.
 - Risposta (application/json): JSON con informazioni sulle conversioni.
- **GET /netex/api/v1/downloadVersion**
 - Descrizione: Permette il download di una specifica versione di dati Netex.
 - Parametri Query: Probabilmente versionId o simile.
 - Risposta: File Netex (XML o ZIP).

4. Risposte di Errore Comuni

Oltre agli errori specifici per endpoint:

- 400 Bad Request: Parametri mancanti o malformati.
 - 401 Unauthorized: Chiave API mancante, non valida, o origine non autorizzata.
 - 403 Forbidden: Chiave API valida ma non autorizzata per la risorsa richiesta.
 - 404 Not Found: Risorsa non trovata (es. fermata inesistente, file non trovato).
 - 500 Internal Server Error: Errore generico del server durante l'elaborazione della richiesta.
- Le risposte di errore (specialmente 500) possono includere un corpo JSON del tipo:

```
{ "success": false, "message": "Descrizione dell'errore" }
```

```
// o
```

```
{ "error": true, "data": { "message": "Descrizione dell'errore" } }
```

```
// o
```



{ "message": "Descrizione dell'errore" }

5. Note Aggiuntive

- Eventuali limiti di utilizzo (rate limiting) saranno comunicati separatamente.
- Le presenti specifiche sono soggette a revisioni. Eventuali modifiche saranno comunicate con adeguato preavviso.