MDS Technical Workshop

Thank you for joining!

The Workshop will begin at 8:00 AM Pacific
MDS Technical Workshop
Uberconference Instructions

Questions

- Submit by chat window on the right of the Uberconference window
- Audio participants will be muted
- Email questions to ladot.innovation@lacity.org
Agenda

8:00  Welcome, Webinar Instructions
8:15  Introduction
8:30  Develop with LADOT

8:50  MDS: Provider
     Demos, API Spec Review, CUP Acceptance Testing
10:00 MDS: Agency
     City Platform, Workflows

11:15 Next Steps
WE ARE LADOT

Our Vision

In Los Angeles, all people have access to safe and affordable transportation choices that treat everyone with dignity and support vibrant, inclusive communities.
Introductions
What is MDS?
Contributors
History

- Open Source - CC0 License
- Released 1 week ahead of City Council Hearing
- SemVar- v0.0.1 just released
- MDS provides the tools to manage mobility at scale
Two APIs

**Provider API**

- Details about prior operations
- Way for “Agencies” to access information about Mobility as a Service Companies
- Required for CUP approval

**Agency API**

- Active Management API
- Required for Full Permit approval
- Longer onboarding processing
Goals: Setup APIs and Interchange before CUP

- All companies must provide staging + prod URLs in their application
- Contact: hunter.owens@lacity.org
- API must
  - Contain all trips
    - 1) Starting in the City of LA
    - 2) Ending in the City of LA
    - 3) With a observed Lat/Long in the routes data structure within the City of LA
    - 4) Where the as the crow flies line between start-point and end-point intersects the City of Los Angeles.
  - Contain and store 1 year of Data
  - Expose methods to allow subsetting by Date
Provider Demo
Trips Object

```
"data": [
{
"company_name": "Bat",
"device_type": "scooter",
"device_id": "f7a7f05d-7259-4686-af0a-4a131571ca3a",
"trip_id": "19e047e9db957851374",
"trip_distance": "007.819185831497",
"route": {
"type": "FeatureCollection",
"features": [
{
"type": "Feature",
"properties": {
"timestamp": 1533586649
},
"geometry": {
"type": "Point",
"coordinates": [
-118.33065461988966,
34.0281296425615
]
}
},
{
"type": "Feature",
"properties": {
"timestamp": 1533586821
},
"geometry": {
"type": "Point",
"coordinates": [
-118.3250978303528,
34.028220820062285
]
}
}
},
"accuracy": 5.0,
"trip_id": "116b2b6f-1683-47da-8d8b-693ccae91ecfe",
"parking_verification": "bot/co/images/on6x2ko",
"standard_cost": 2,
"actual_cost": 115
}
```

- vehicle_id vs device_id
- Route
- Parking Verification Link
Status Changes -

https://github.com/CityOfLosAngeles/mobility-data-specification/tree/master/provider#event-types
Provider Stack

**Provider API**
Up to date information about company, build by operator

**LA City Data Ingest**
Regular polls of data and stored in AWS

**Reporting, Monitoring and Compliance**
Automated reporting, compliance and measurements.
Dashboard Demo

Dockless Dashboard | Weekly Overview

August 04, 2018 - August 11, 2018

Filter by mobility provider: All, Bat, Lemon
Filter by vehicle type: All, Scooter, Bike

Trips Between Council Districts

Dockless Trips Per Company

Lemon
Bat

16 SEPTEMBER 12, 2018
Notifications and Alerts

- Hour Job Runs
- Over Cap Alerts
- 24 Hour Device in same location Alerts
- Other, TBD, Alerts
- This is a test phase
- Please provide an email address for us to send these notifications
Cap Measurement Methodology

3 Reasons Counting is the Hardest Thing in Data Science


- How do we check for cap compliance while devices move, by their nature
Cap Measurements

- Need to measure moving vehicles
- Need to measure multiple areas (equity cap)
- We make an availability view
- Cap: \( \frac{n(t)}{T} \)
  - \( n \) = number of devices
  - \( t \) = amount time on street
  - \( T \) = Time Delta

- **Implementation**
Q & A
BREAK
Summary

- CUP - Provider 0.1.0
- Starting a regular process of updates.
- PR, comments welcome.
- Ecosystem of MDS tooling
AGENCY API

A digital interface to a physical platform
Some Differences from the Provider API

- Agency API is meant to work within your workflows. The city is “in the loop”
- Service level driven
## LADOT Service Level EXAMPLES (DRAFT)

<table>
<thead>
<tr>
<th>Type of Measure</th>
<th>Example SLO Requirement</th>
<th>Measurement Period</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>The application will be available 99.95% of the time</td>
<td>Over a year</td>
</tr>
<tr>
<td>Service Desk Response</td>
<td>75% of help desk calls will be answered in less than a minute</td>
<td>Over a month</td>
</tr>
<tr>
<td></td>
<td>85% of help desk calls will be answered within two minutes</td>
<td></td>
</tr>
<tr>
<td></td>
<td>100% of help desk calls will be answered within three minutes</td>
<td></td>
</tr>
<tr>
<td>Incident Response Time</td>
<td>99% of severity 1 tickets will be resolved within three hours</td>
<td>Over a quarter</td>
</tr>
<tr>
<td></td>
<td>98% of severity 2 tickets will be resolved within eight hours</td>
<td></td>
</tr>
<tr>
<td></td>
<td>98% of severity 3 tickets will be resolved within three business days</td>
<td></td>
</tr>
<tr>
<td></td>
<td>98% of severity 4 tickets will be resolved within five business days</td>
<td></td>
</tr>
<tr>
<td>Response Time</td>
<td>85% of TCP replies within 1.5 seconds of receiving a request</td>
<td>Over a month</td>
</tr>
<tr>
<td></td>
<td>99.5% of TCP replies within 4 seconds of receiving a request</td>
<td></td>
</tr>
</tbody>
</table>
Access / Security

Two pieces of information are required to interact with the Agency API:

- `provider_id` - Issued *inside* MDS
- `API_key` - Issued *after* CUP by individual City
REGISTERING A VEHICLE

- **register_vehicle**
- **update_vehicle_status**
- **deregister_vehicle**
## VEHICLE REGISTRATION

**register_vehicle Endpoint (post)**

<table>
<thead>
<tr>
<th>Key</th>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>unique_id</td>
<td>UUID</td>
<td>UUID v4 provided by Operator to uniquely identify a vehicle</td>
</tr>
<tr>
<td></td>
<td>provider_id</td>
<td>string</td>
<td>Unique service provider identifier that will be issued by the City as part of the permit application process.</td>
</tr>
<tr>
<td></td>
<td>vehicle_id</td>
<td>string</td>
<td>Vehicle identification number</td>
</tr>
<tr>
<td></td>
<td>registration_status</td>
<td>enum</td>
<td>Status of the record (Active / Inactive)</td>
</tr>
<tr>
<td></td>
<td>vehicle_type</td>
<td>enum</td>
<td>Vehicle Type</td>
</tr>
<tr>
<td></td>
<td>propulsion_type</td>
<td>enum</td>
<td>Type of propulsion</td>
</tr>
<tr>
<td></td>
<td>vehicle_year</td>
<td>enum</td>
<td>Year manufactured</td>
</tr>
<tr>
<td></td>
<td>vehicle_mfgr</td>
<td>enum</td>
<td>Manufacturer name</td>
</tr>
<tr>
<td></td>
<td>vehicle_model</td>
<td>enum</td>
<td>Vehicle model name</td>
</tr>
<tr>
<td></td>
<td>vehicle_status1</td>
<td>enum</td>
<td>Event_type from MDS Event Types table</td>
</tr>
<tr>
<td></td>
<td>vehicle_status2</td>
<td>enum</td>
<td>Reason from MDS Event Types table</td>
</tr>
</tbody>
</table>
IMPLEMENTATION NOTES

register_vehicle Endpoint (post)

- Each vehicle will be digitally identified based on a UUID v4 to identify vehicle as a unique_id provided by the operator.

- The vehicle_id submitted must be the numerical value represented by the scannable code on the bike.

- Both unique_id and vehicle_id will need to be the same for Agency & Provider APIs.
IMPLEMENTATION NOTES

register_vehicle Endpoint (post)

- New registrations will default to inactive and will not be counted against you.

- No re-registrations. Register_Vehicle can only be called once for a given provider_ID, unique_id, vehicle_id combination.
VEHICLE RE-REGISTRATION

update_vehicle_status Endpoint (put)

- Re-registration occurs automatically when `update_vehicle_status` is called

- API requires submission of vehicle status parameters using the same enumerations as the Provider API.

```
  event_type
t  reason_code
```
VEHICLE DE-REGISTRATION
deregister_vehicle  Endpoint (put)

This call updates the vehicle registration status from active to inactive for a given unique_id.
Agency API Workflows

RENT VEHICLE

start_trip -> Update_vehicle_telemetry -> end_trip
Starting a trip requires vehicles to be registered and deployed
**SUBMITTING TRIP DATA**

update_trip_telemetry **Endpoint** *(post)*

- Data structures are the same as provider API
- Frequency of transmission shall be between **5-6 seconds**.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Required/Optional</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>trip_id</td>
<td>UUID</td>
<td>Required</td>
<td>Issued by InitMovementPlan() API</td>
</tr>
<tr>
<td>timestamp</td>
<td>Unix Timestamp</td>
<td>Required</td>
<td>Time of day (UTC) data was sampled</td>
</tr>
<tr>
<td>route</td>
<td>Route</td>
<td>Required</td>
<td>See detail below.</td>
</tr>
<tr>
<td>accuracy</td>
<td>Integer</td>
<td>Required</td>
<td>The approximate level of accuracy, in meters, represented by start_point and end_point.</td>
</tr>
</tbody>
</table>
## ENDING A TRIP

**end_trip** **Endpoint** *(post)*

### Field | Type | Required/Optional | Other
---|---|---|---
trip_id | UUID | Required | See [start_trip](#)
timestamp | Unix Timestamp | Required | Date/time that event occurred. Based on GPS clock.
location | Point | Required | Location at the time of status change in WGS 84 (EPSG:4326) standard GPS projection
accuracy | Integer | Required | The approximate level of accuracy, in meters, represented by start_point and end_point.
battery_pct_end | Float | Require if Applicable | Percent battery charge of device, expressed between 0 and 1
TRIP PROCESSING EXCEPTIONS

Exception codes:

1. 250: Trip_ID not found
2. 251: Trip Event updates low frequency
3. 252: Trip Event updates high frequency
4. 253: Trip not properly ended
5. 270: Vehicle_ID used in trip was not registered
message

For 'message', options are:

- 200: OK
- 201: Created
- 202: Accepted
- 203: Added
- 204: Removed
- 210: Warning: vehicle used in this trip has not been properly registered
- 305: Error: vehicle is already registered
- 306: Error: vehicle registration cannot be found
- 310: Error: vehicle is not properly registered
- 311: Error: duplicate registration found, please use a different unique_id
- 315: Error: vehicle is not active
- 320: Error: vehicle trip has not been properly started
RECURSIVE BEHAVIOR

- Register Vehicle
- Deploy Vehicle
- Start Trip
- Update Telemetry
- End Trip

Errors:
- 306 Error
- 315 Error
- 320 Error
API’s and data standards for municipalities

Inspired by GTFS and GBFS. Specifically, the goals of the Mobility Data Specification (MDS) are to provide API and data standards for municipalities to help ingest, compare and analyze mobility as a service provider data.

The specification is a way to implement real-time data sharing, measurement and regulation for municipalities and mobility as a service providers. It is meant to ensure that governments have the ability to enforce, evaluate and manage providers.
Q & A
FLEET SIZE & ABANDONED DEVICES

MAXIMUM CAP & EQUITY BONUS ALLOWANCES

- Citywide Cap: 3,000 devices
  Equity Bonus: +2,500 devices in areas scored above 75th percentile in CalEnviroScreen 3.0
  Equity SF Bonus: +5,000 devices in areas scored above 75th percentile in CalEnviroScreen 3.0 within San Fernando Valley
  GeoJSON/Shapefile: Available on geohub.lacity.org

LADOT will make periodic calls to Provider API from MDS to check compliance with fleet size requirements

ABANDONED DEVICES

- LADOT will make periodic calls to Provider API from MDS to check whether device has been parked in one location for more than 5 consecutive days
PROHIBITED AREAS & FUTURE

PROHIBITED AREAS

- GeoJSON/Shapefile: Available on geohub.lacity.org
Update Frequency: Beginning of every month
Future: With Agency MDS, operators may call `check-parking` API
LADOT will make periodic calls to Provider API to check compliance with parking in prohibited areas

ADDITIONAL RULES

- The City of Los Angeles may add additional service quality criteria in the future
Next Steps:

- City Council Must Adopt Enacting Legislation - (Estimated September 18)
- CUP Application Released - September 19
- CUP Applications due: September 30, October 30, November 30
- MDS Reviews: Hunter.Owens@lacity.org / Todd.Petersen@lacity.org
- Technical questions and Pull Requests on Github
- Full Permit Application released end of October