Uber System Design (Basic) Cheat Sheet

V2021.03.17

(Dr Yan Xu)

Overview

- Web Service
- Real-time Location-based Service
- Rider-Driver Marketplace

Core Services

- Driver Service
  - driver information CUID
- **Location Service (challenging)**
  - geo-location update and index
- Rider Service
  - rider information CUID
- **Dispatch Service (challenging)**
  - effective and fare dispatch
- **Map Service (challenging)**
  - estimate travel time from A to B
- Pricing Service
  - promotion and groupon
  - surge price: demand/supply rebalance
- Payment Service
  - transaction
  - reports

System KPIs

- accurate location any time (avg <5m)
- accurate map
- accurate trip duration estimation (avg <20%)
- short rider's waiting time (medium < 5mins)
- maximize drivers' busy time (avg > 50%)
- fare task assignment among drivers

System Non-Functional Requirement

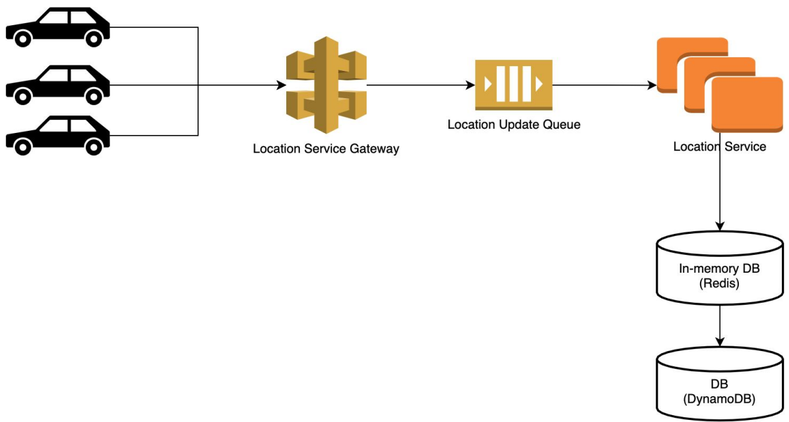
- Drivers: **1M**
- Location Update: **once per 3 sec**
- Peak Active Drivers: **200K**
  - Active: accessing service API, not online
- Location Storage:
  - 100 B * 200K * 3600 * 24 = **1.6 TB per day**
- Riders: **20M**
- Peak Active Riders: **10K**
  - Active: accessing service API, not online

System High-level Design

- Each city runs its own Uber system
  - sharing code & CI/CD
  - city -scale configuration: **10%** of system config
    - Peak Location Update: **20K QPS**
    - Location Storage: **100B*20K*3600*24 ~200GB per day**
    - Peak Active Riders: **1K QPS**
- Use Google Map as Map Service
- Drivers update vehicle location once every 3 seconds.
  - It could be optimized by smart location, which update locations more around intersections and less on straight roads.

Location Service

- Queue: Kafka (due to high performance streaming)
- Location Service:
  - lat/lng -> Google S2 location
  - Update Google S2 location to Redis
  - Flush Redis to DynamoDB periodically
- Location Storage
  - Redis: most recent location
  - DynamoDB: 24 hours for fast access
  - S3: storage (e.g. 3 years)

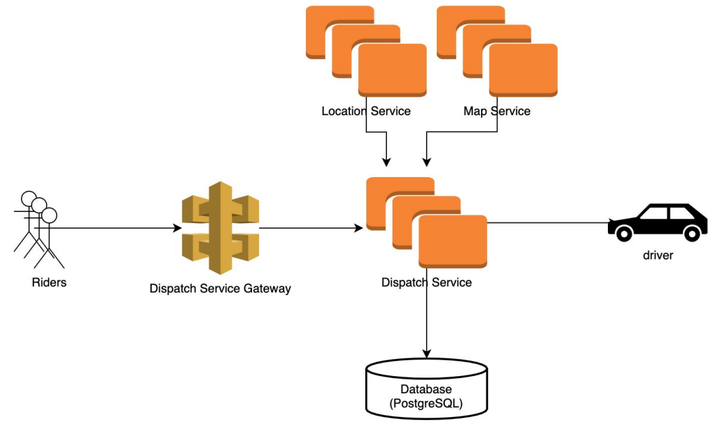


Map Service

- fast trip duration estimation during driver dispatch
- accurate trip duration estimation during fee estimation
- deal with various rider requirement (e.g. no toll)
- keep up-to-date with map updates

Dispatch Service

- Step 1: Obtain rider's location
- Step 2: Fetch nearby available drivers
- Step 3: Estimate waiting time (<- Map Service)
- Step 4: Vehicle Dispatch
- Step 5: Assignment Confirmation
- Step 6: Estimate Arrival Time and Fee
  - It could be another service
- *(it should reuse the driver connection socket)*



Role of Machine Learning

- Demand / Supply Prediction
- Driver Fatigue Detection
- Predict Prob of Like between Riders and Drivers
- Travel Time Prediction (it is difficult for ML)
- Customer Service
- Uber Eat Recommendation

Real-world Considerations

- Uber Share/Pool
- Uber Eat
- City-scale Development becomes not suitable when the # of cities becomes 100+