CloudKit: Structured Storage for Mobile Applications

Karl Andreas Eggan & Alf Martin Eggan
Motivation

- Multiple devices per person
- Seamless switches
- Hard to develop
  - Backend server interaction
  - Local cache
  - Sync mutations
  - Potentially millions of users
  - Consistency and durability
  - Regain connectivity
  - Selective filtering
  - Notifications
CloudKit

- Apple’s cloud backend service
- Application development framework
- Scale, consistency, durability and security
- First of its kind
Data Model

- Mobile use-case design
- Types
  - Containers
  - Databases
  - Records
  - Record zones
  - Environments and Schema
Container
Databases
# Groceries Record

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>“My Groceries” (string)</td>
</tr>
<tr>
<td>Body</td>
<td>“milk, bread, eggs” (string)</td>
</tr>
<tr>
<td>CreationDate</td>
<td>03/01/2017 (date)</td>
</tr>
<tr>
<td>Folder</td>
<td>… (reference)</td>
</tr>
</tbody>
</table>
Record zone
Environments and Schemas

- Strong schema management
  - Still rapid development
- Environments
  - Developer
  - Production
API, Semantics, Architecture

- CloudKit APIs
- Dashboard
- Architecture overview
- Data placement
- Read and update semantics
API, Semantics, Architecture – CloudKit APIs

- Client-side libraries
  - Swift
  - Objective-C
  - JavaScript
- Provides functionality to access and update data
API, Semantics, Architecture – Dashboard

- Tool for developers
- View and manage app data, logs and statistics
- Web-client
- Both production and development environment
API, Semantics, Architecture
- Architecture overview

- REST-like web interface
- gRPC
- Custom interface over TCP
API, Semantics, Architecture
- Data Placement

- Data sharded into multiple CloudKit partitions
  - Each user to a single partition
- Each partition assigned to one Cassandra cluster
  - Containing multiple Cassandra partitions
- Each custom zone assigned one Cassandra partition
Conflict Resolution

- App’s responsibility
- Three copies of the record
  - Client record
  - Server record
  - Ancestor record
- Not resolvable conflicts
  - Error dialog box
Use Patterns

- Publish-Subscribe
- Cross-Device Sync
- Sharing and Collaboration
- Bounded Queue
- Cloud Storage
Change-Tracking (Sync)

- Synchronize between devices
- Requirements
  - Initiated by client
  - Efficient w.r.t. network traffic
  - Able to sync down a part of the data at a time
  - May be interrupted at any time
  - Client should sync to a consistent state
- Types
  - Forward Sync
  - Reverse Sync and Meta-Sync
  - Snapshot Sync
Sharing

Owner
1. Create a Share (private database)
2. Add Participant

Participant
3. send URL (e.g., email)
4. accept share
5. Set participant status to accepted, update permissions index

ACK + zone metadata
6. Add zone to shared database (if not there)

Share Record

“Shopping List”
https://www.icloud.com/shopping-app/<unique_token>

publicPermission: none
participants:

Participant #1
acceptanceStatus: accepted
permission: readWrite

Participant #2
acceptanceStatus: invited
permission: readOnly
Queries, Subscriptions and Notifications

- Queries retrieve subset of records in zone
- Single record type across record zones
- Flexible filters

- Three type of subscriptions
  - Zone Subscription
  - Database Subscription
  - Query Subscription

- Notify user
- Notify app
**Schema Management**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>type_id (primary key)</td>
<td>uuid1</td>
</tr>
<tr>
<td>type_name</td>
<td>Restaurant</td>
</tr>
<tr>
<td>type_definition_id</td>
<td>uuid2</td>
</tr>
<tr>
<td>Field types</td>
<td>(see below)</td>
</tr>
<tr>
<td>Query subscription types</td>
<td>(see Section 7)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>field id</th>
<th>name</th>
<th>type</th>
<th>flags</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&quot;Name&quot;</td>
<td>String</td>
<td>QSZ</td>
</tr>
<tr>
<td>2</td>
<td>&quot;Cuisine&quot;</td>
<td>Integer</td>
<td>Q</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>schema_id</th>
<th>uuid3</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent schema_id</td>
<td>uuid4</td>
</tr>
<tr>
<td>record types</td>
<td>Restaurant</td>
</tr>
<tr>
<td></td>
<td>Review</td>
</tr>
<tr>
<td></td>
<td>Reservation</td>
</tr>
</tbody>
</table>
CloudKit Use

- Public and private data
  - 20% apps only public
  - 49% apps only private
  - 31% apps both
- CAS contention (150 apps)

<table>
<thead>
<tr>
<th>Fail rate</th>
<th>Number of apps</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;0.005%</td>
<td>75</td>
</tr>
<tr>
<td>&lt;0.1%</td>
<td>122</td>
</tr>
<tr>
<td>&lt;0.05%</td>
<td>All internal apps</td>
</tr>
<tr>
<td>&gt;1%</td>
<td>8 third-party apps</td>
</tr>
</tbody>
</table>
Summary

- CloudKit is a leading mobile backend service
- 100s of millions active daily users
- Partitions app data into private and public databases
- Schema prepares for scale
- Explicit set of tools for consistency across devices
  - Sync
  - Subscription
  - Queries