Ad Serving Workflow Document

I. Introduction

1. The function of Ad Server is to deliver relevant ads to intended audience at right time.
2. Briefly Ad Server receives ad requests from mobile apps integrated with SDKs and delivers a relevant ad to the respective apps.
3. Additionally, Ad Server is responsible for inserting relevant and accurate records into databases for reporting, billing and analytical purposes.

II. Terminology

1. **Advertiser:** An entity which intends to communicate information or a message about their product or services via ad network.

2. **Publisher/App Developer:** An entity which uses SDK for monetization of content/apps.

3. **User:** An individual which has at least one app integrated with SDK running on their device (Mobile/Tablet).

4. **Ad Server (or Ad Server or API)**: AdServer is cluster of servers running AdServering code on managed hosting infrastructure.

5. **SDK** : Software library which app developers integrate with their app for monetization. SDK is responsible for making ad requests from app user's device to AdServer. SDK is also responsible for rendering the ad creative and making notification calls for events like impressions, clicks and conversions.

6. **Frequency CAP (or FCAP):** Maximum number of ads served per user. FCAPs can be of multiple levels – Campaign level or user level.

7. **Campaign:** An advertiser can have multiple campaigns each having set of attributes like start time, daily budget, targeting and other specifications.

8. **Creative:** An ad banner or text message which is to be displayed on user's device.

9. **Delivery Cluster:** Cluster of DB slave nodes specifically used for highly available read only operations.

10. **Transactional DB:** Cluster of DB nodes used for logging transaction information like impressions, clicks and conversions.

11. **Budget Cluster:** DB cluster nodes used for realtime set and retrieval of real-time budgeting information.

III. Components

1. **Webnodes**
    a. Webnodes are servers in Ad Serving cluster which receive ad requests and respond back with ad response.
    b. Webnodes also record information about impressions, clicks and conversions.

2. **Cache**
    a. Redis cluster are used to cache most frequently accessed data.
    b. They are additionally used to maintain information like FCAP etc.

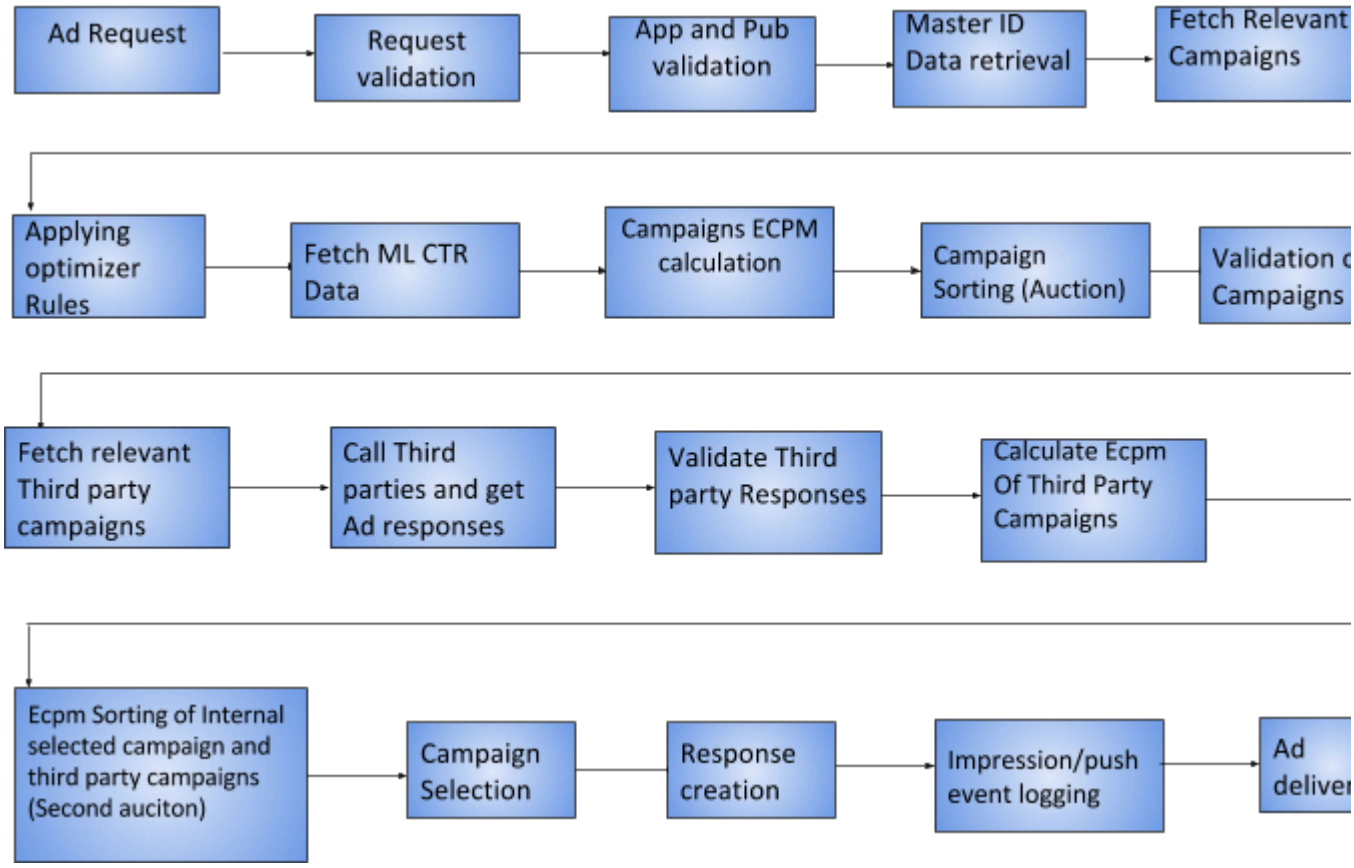
3. **UserInfoDB**
    a. UserInfoDB is a cluster which store necessary information for each user.
    b. Multiple data points like installed apps, some device information like manufacturer, model etc are stored in this cluster.

4. **DBs**
    a. Publisher/Developer level settings and details are stored in Delivery cluster.
    b. Advertiser/Campaign specific information is stored in Delivery cluster.
    c. Transactional DB stores information about impressions, clicks and conversions. This data is then imported into DB cluster for analytical processing.
    d. Realtime budgeting information is set and retrieved per transaction on group of low latency nodes called Budget Cluster.

5. **Third party libraries/services – To be decided**
    a. **GeoMaxmind** – IP to location mapping.
    b. **Ip2location** – Another such mapping service.
    c. **Device Atlas** – Used to derive device, browser related information from user agent
    d. **Net Acuity** – IP to location mapping service

IV. Workflow

A. Ad Request workflow



Brief explanation of each step:

1. **Request Validation**
   - **a.** Parameters Validation
   - **b.** Fraud checks (imei, ip fcap, other checks)

2. **App and User validation**
   - **a.** App running status
   - **b.** Ad Format support check
   - **c.** App id marked as fraud or not check

3. **Master Id Data Retrieval**
   - **a.** Fetch ids corresponding all the targeting parameter names (ex: country id for country name, device id for device name etc)

4. **Fetch relevant campaigns**
   - **a.** Fetch campaigns from DB/cache based on ad type, targeting, Geo, device, time, carrier and manufacturer. (Some targeting is done in this phase).
   - **b.** In this step internal campaigns + static feeds are fetched from DB

5. **Applying Optimizer Rules**
    a. Fetch Optimizer rules from DB/Cache for the campaigns
    b. Adjust bid based on :
        i. Country
        ii. State
        iii. Carrier
        iv. Device Model
        v. Device Manufacturer
        vi. Publisher
        vii. Application
        viii. Time
        ix. Creative
        x. Landing page
        xi. Network connection type

6. **Fetch ML CTR Data:**
    a. Fetch ML CTR values for the campaigns from cache based on ad format, container etc

7. **ECPM Calculation**
    a. For each campaign Ecpm is calculated as below
        i. FOR CPC campaigns, Ecpm = 1000*CTR*CPC bid
        ii. FOR CPM campaigns, Ecpm = CPM bid

8. **Campaign Sorting**
    a. Campaigns are sorted in descending order of the following parameters
        i. Priority
        ii. Ecpm

9. **Validation of campaigns**
    a. Creative & Landing page validation
    b. Budget validation
    c. FCAP
    d. Min Bid validation
    e. Some of the targeting are validated in this step
        i. Day Parting
        ii. Zip code
        iii. City Targeting
        iv. App/Pub Targeting/ Blocking
        v. App inclusion/exclusion
        vi. IP targeting

10. **Fetch Relevant Third Party campaigns**
    a. Based on country, ad format, container fetch relevant Third party campaigns from DB/cache/config

- **b.** Validations
    - i. App Brand Safe Check
    - ii. Package Google Play Non-Google Play check etc
- **c.** Floors calculation
    - i. Based on country fetch static, hard floor, margin from cache for the campaigns
    - ii. Calculate floor value based on above parameters

11. Call Third Party partners and Get Responses

- **a.** Request generation for each third party campaign
- **b.** Make a http call to get the response

12. Validate Third Party Campaigns Responses

- **a.** Parse the response and following validation are done on parsed response
    - i. Empty null/empty response check
    - ii. Response Format/assets check
    - iii. Ecpm zero check

13. Second Auction

- **a.** Auction between selected internal campaign and third party campaigns are based on following parameters
    - i. Priority
    - ii. ECPM

14. Campaign Selection

- **a.** Campaign that won the second price auction is selected for serving

15. Response Creation

- **a.** Ad format specific response creation

16. Impression Logging

- **a.** Creation of transaction log with relevant campaign, app , cpm information and inserting it on Transaction DB.

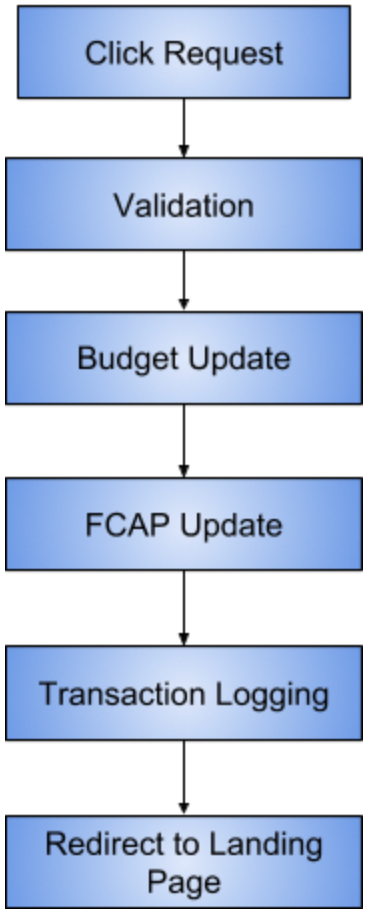
17. Ad Delivery

- **a.** Responding back to SDK ad request.

NOTE:

1. Feeds, OpenX are third party demand sources.
2. Their creative information is received on the fly via a network call.
3. An internal auction (second auction) is conducted with internal campaigns in order to select the best campaign.

B. Ad Click Workflow



Brief explanation of each step:

1. **Validation**
    a. Transaction duplication check (based on transaction id)
    b. Parameter validation for sanity

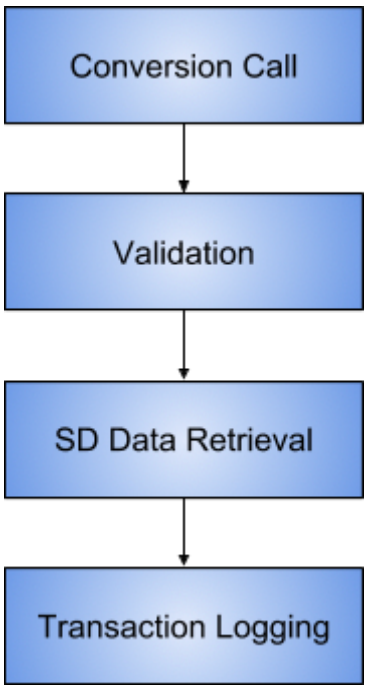
2. **Update Budget**
    a. Reduction of budgets for campaign and advertiser.

3. **Update FCAP**
    a. Update FCAP info in cache cluster.

4. **Transaction logging**
    a. Creation of transaction log with relevant campaign, app , cpm information and inserting it on Transaction DB.

5. **Redirect**
    a. Redirecting user to creative landing page.

C. Conversion Workflow



Brief explanation of each step

1. **Validation**
    a. Transaction duplication check (based on transaction id)
    b. Parameter validation for sanity

2. **SD Data Retrieval**
    a. Fetch SD data for the guid from cache cluster

3. **Transaction Logging**
    a. Creation of transaction log with relevant campaign, app , cpm information and inserting it on Transaction DB

D. Brief architecture diagram