March 20, 2018

METHOD FOR SUPPORTING LARGE-SCALE INVENTORY FOR IN-APP PURCHASES ON MOBILE DEVICES

Brian Schmidt

METHOD FOR SUPPORTING LARGE-SCALE INVENTORY FOR IN-APP PURCHASES ON MOBILE DEVICES

Users purchase virtual and physical products through a marketplace (e.g., an application (app) store provided by a marketplace server, etc.) by interacting with an app displayed via a user device (e.g., a mobile device, a smart phone). The marketplace may serve as an exchange for user devices within an ecosystem.

Typically, an app (e.g., provided by a server device) is allocated a fixed set of product identifiers from a marketplace (e.g., provided by a marketplace server). The server device may map a unique product identifier from the fixed set of product identifiers to each of the products available for purchase within the app. The mapping may be a one-to-one static mapping (e.g., each product permanently has a corresponding product identifier). The server device may map each product identifier used by the app to a purchase price to be used on the marketplace. Responsive to a user selecting a product to purchase within the app (e.g., virtual item in a game app, paid sponsorship in a media app, physical item in a merchandise app, etc.), the server device may use the assigned product identifier and purchase price to execute the purchase transaction on the marketplace.

Some apps include a large-scale inventory that exceeds the fixed set of product identifiers available on the marketplace. For example, an app may offer a paid sponsorship for each channel on a platform. Thousands to millions of channels may be available, for which sponsorship could be purchased. The marketplace may only offer tens to hundreds of product identifiers. This causes many channels on the platform to be unsupported by the marketplace. Once the app reaches the limit set by the marketplace, the app cannot offer more products for sale. This situation is illustrated in Tables 1-2.

Table 1 may be maintained by a server device that provides an app for making purchases over a marketplace. As shown in Table 1, the app may have an inventory of M products for sale and the server device may map each product to a marketplace price tier (e.g., Tier a, Tier b, etc.). Table 2 may be maintained by a marketplace server that provides the marketplace. Table 2 includes a mapping from marketplace price tiers to currency-specific prices. The server device may map the tiers on Table 1 to tiers on Table 2. For example, the server device may map "Marketplace Price Tier a" from Table 1 to "Marketplace Price Tier 1" on Table 2. The server device may adjust to which tier on Table 2 each tier on Table 1 is mapped. The marketplace server may adjust the currency specific prices for each tier on Table 2. The marketplace may offer N unique product identifiers.

As shown in Table 1, the server device may map M products for sale (e.g., for sale via the app) to product identifiers. There is a fixed number N of product identifiers available for each app via the marketplace. The number of tiers may be limited, but multiple products can share the same pricing tier (e.g., product 10 and product 500 may both utilize Tier 5). The number of pricing tiers P is unrelated to M and N, but typically P<N. The server device uses the product identifiers to purchase a product via the marketplace. When the M products to offer for sale via the app exceeds the N unique product identifiers available via the marketplace (M>N), products N+1 to M are unsupported and cannot be purchased on the marketplace. Because of this, typical marketplaces do not support a large-scale inventory for in-app purchases.

The above and other deficiencies are addressed by providing a user-level transaction table for purchases. The mapping may be inverted in the user-level transaction table so that product identifiers are dynamically mapped to products on a per-user basis (e.g., instead of each product being statically mapped to a product identifier regardless of the user). Upon receiving a user

request to purchase a first product item, a server device may map a first product item to a first pricing tier. The first pricing tier may be associated with a first set of product identifiers. The server device may select a first product identifier that is not actively being used by the first user account from the first set of product identifiers. The server device may transmit the first product identifier and the first pricing tier to a marketplace server to execute purchase of the first product item. Responsive to the purchase, the server device may enter, in the user-level transaction table, an entry including an indication of the first product item, the first product identifier, and a first lifetime for the purchase. The user-level transaction table may be associated with a specific user account to store purchases made via the user account. Upon expiration of the first lifetime for the purchase, the server device may remove the first entry from the user-level transaction table and another product may be purchased.

By using a user-level transaction table, instead of mapping the limited set of product identifiers available via the marketplace to the maximum amount of products that can be offered for sale by an app, the limited set of product identifiers are mapped to the maximum amount of products that can be purchased per each user account. A user account can purchase any of the items offered via the app in a dynamic fashion. Use of a user-level transaction table allows an app to provide M products for sale via a marketplace that has N product identifiers available, even when M>>N. Each user can purchase up to N of the M products for sale via the app.

FIG. 1 depicts a flow diagram for an illustrative example of a method for implementing a user-level transaction table for purchases. The method in FIG. 1 is an example method from the perspective of a processing device that manages the user-level transaction table. The method may be performed by a processing device that may include hardware, software, or a combination of both. The processing device may be part of a server device that provides an app. The app may be

used to make purchases via a marketplace.

Referring to FIG. 1, a processing device may receive, from a user device associated with a first user account, a user request to purchase a first product item. The user device may send the user request to the processing device. In some implementations, the processing device may cause the user device to display an application (e.g., may stream at least a portion of the application to the user device, the application may be a web application) and the user request may be entered in the application via the user device. In some implementations, the user device may execute an application (e.g., a native application) and the user device may transmit the user request to the processing device via the application. The user request may identify a product displayed by the application. The user request may provide payment (or an indication of payment) to purchase the product.

The product may be a virtual good (e.g., a sponsorship of a channel, a gem or coin in an online game, etc.) or a physical good (e.g., a t-shirt, merchandise, etc.). The user request may be to purchase the first product item via a marketplace (e.g., a marketplace provided via a marketplace server, an app store). For example, the user request may be to purchase a sponsorship of a channel via an app store.

The processing device may determine that the user account is authorized to purchase the first product item. For example, the processing device may determine that the first product item is not already on the user-level transaction table for the user account (e.g., determine the user is not currently sponsoring the same channel as the channel specified in the user request). In another example, the processing device may determine the user account has not met a maximum limit of purchases with an active lifetime in that price tier.

Returning to FIG. 1, the processing device may map the first product item (e.g.,

associated with the user request) to a first pricing tier. The processing device may map the first product item to a first product tier (e.g., map “Product 0” to “Tier a”) on a primary table (e.g., Table 3). The processing device may map each of the M product items available for purchase via the app to a corresponding price tier on the primary table. For example, an administrator may provide the price tier for each of the products to the server device. The processing device may store and maintain the primary table. The processing device may cause the user device (e.g., by instructing the user device) to cache the primary table on the user device.

Returning to FIG. 1, the processing device may determine a set of product identifiers corresponding to the first pricing tier. The processing device may maintain and store a secondary table (e.g., Table 4) that indicates product identifiers available for each tier. For example, “Tier 0” may be associated with product identifiers ID x and ID y. The processing device may map the tiers from the primary table to the tiers in the secondary table. The processing device may cause the user device (e.g., by instructing the user device) to cache the secondary table on the user device.

Returning to FIG. 1, the processing device may select, from the set of product identifiers, a first product identifier that is unused by the first user account. If all of the product identifiers for the first pricing tier are being used for purchases made via the app, the purchase associated with the user request may be unsupported until one of the purchases expires. For example, if ten product identifiers corresponding to channel sponsorships are available for a first product tier and if the user is already sponsoring ten channels, the user should wait until one of the ten sponsorships expires to be able to purchase another sponsorship at the first product tier. The number of purchased items with an active lifetime may be limited to the number of product identifiers available in the marketplace for a given pricing tier. Each product identifier

(regardless of the associated product) may be associated with the same pricing tier across users, otherwise, the purchase may be rejected by the marketplace (e.g., the marketplace may require the app to set fixed mapping from a product identifier to a pricing tier).

Returning to FIG. 1, the processing device may transmit, to a marketplace server, the first product identifier and the first pricing tier to execute purchase of the first product item. The marketplace server may verify the first product identifier and the first pricing tier. The marketplace server may provide the product corresponding to the product identifier to the user device.

Returning to FIG. 1, the processing device may enter, into a user-level transaction table (e.g., Table 5) corresponding to the first user account, a first entry including an indication of the first product item, the first product identifier, and a first lifetime for the purchase. The user-level transaction table may be limited to N purchases with an active lifetime, where N is the amount of unique product identifiers offered by the marketplace. Each of the N purchases may be mapped to one of the M products that are available via the app, where M>N. An administrator may provide the lifetime for the purchase for each product identifier or for each product item to the server device.

Each user of the app may have a user account or other data structure associated with the user or user device in which purchase transactions are recorded. The processing device may store and maintain the user-level transaction table. The processing device may cause the user device (e.g., by instructing the user device) to cache the user-level transaction table on the user device.

In one example, in exchange for the channel sponsorship purchase price, a channel creator may provide benefits (e.g., badging in chat messages, emails, newsletters, membership in a fan club) to the user for a lifetime of the channel sponsorship purchase (e.g., a limited amount

of days). In another example, in exchange for a virtual item purchase price, a game distributor may provide a virtual item (e.g., a virtual coin, an extra life in the game) to a user via a game for a lifetime of the virtual item (e.g., until the virtual item is used in the game). In another example, in exchange for a t-shirt purchase price, a merchandise distributor may provide a t-shirt to a user and the t-shirt purchase may have a lifetime (e.g., until the t-shirt is shipped, a set amount of time after the purchase, etc.). In another example, the lifetime for the product may be instantaneous (e.g., the product identifier can be used for another product once the purchase goes through).

Returning to FIG. 1, responsive to determining that the first lifetime for the purchase has expired, the processing device may remove the first entry from the user-level transaction table (see Table 5).

Tables 6-8 illustrate an example of using a user-level transaction table to purchase channel sponsorships. A media app provided by a server device may provide viewers access to 1M video channels on a platform provided by the server device. The media app may be associated with a marketplace that provides 100 fixed pricing tier levels, where tier X represents XUSD (and equivalents in other currencies). The media app supports monthly paid channel sponsorships at “Tier 10” (10USD) and “Tier 100” (100USD). As shown in Table 6, of the 1M channels on the platform, channels 1-500K are priced at “Tier 10,” while the rest are priced at “Tier 100.”

In the illustrated example, the marketplace supports 50 product identifiers. As shown on Table 7, the app chooses to support up to 10 active paid sponsorships at any level, requiring only 20 of the 50 available product identifiers (e.g., 10 for “Tier 10” and 10 for “Tier 100”). The remaining 30 product identifiers can be used for other products.

Table 8 illustrates a user-level transaction table for a specific user account. The app

supports up to 10 active paid sponsorships, so the user account has purchased a full set of channel sponsorships the app allows. The user account is not constrained by the number of channel subscriptions available for purchase. Upon expiration of the subscription for channel 155 (in 1 day per Table 8), the user account will be able to purchase another channel sponsorship. If the user account is to sponsor channel 1M, the primary table (Table 6) indicates 1M is a "Tier 100" product (e.g., has a price of 100USD on the marketplace). The app consults the secondary table (Table 7) to find that the first unused product identifier at Tier 100 is "ID 13." The app issues a purchase transaction for 100USD (or equivalent currency) for product identifier "ID 13" and updates the user-level transaction table accordingly.

ABSTRACT

A method for supporting large-scale inventory for in-app purchases on mobile devices is described. A server device receives, from a user device associated with a first user account, a user request to purchase a first product item and maps the first product item to a first pricing tier. The server device selects, from a set of product identifiers corresponding to the first pricing tier, a first product identifier that is unused by the first user account. The server device transmits, to a marketplace server, the first product identifier and the first pricing tier to purchase the first product item. The server device enters, into a user-level transaction table, a first entry including an indication of the first product item, the first product identifier, and a first lifetime for the purchase and removes the first entry from the user-level transaction table upon expiration of the first lifetime.

Keywords: in-app purchase, IAP, in-application purchase, in-game purchase, virtual goods purchase, ecom purchase, online purchase, purchase, transaction, buy, order, product, model, unit, item, identifier, ID, parameter, number, code, price, cost, rate, lifetime, life, lifespan, duration, subscription period, table, list, record, catalogue, index, register, limit, restrict, obstruct, control, ceiling, constrain, inverting, reversing, changing, transposing, secondary, external, peripheral, subsidiary

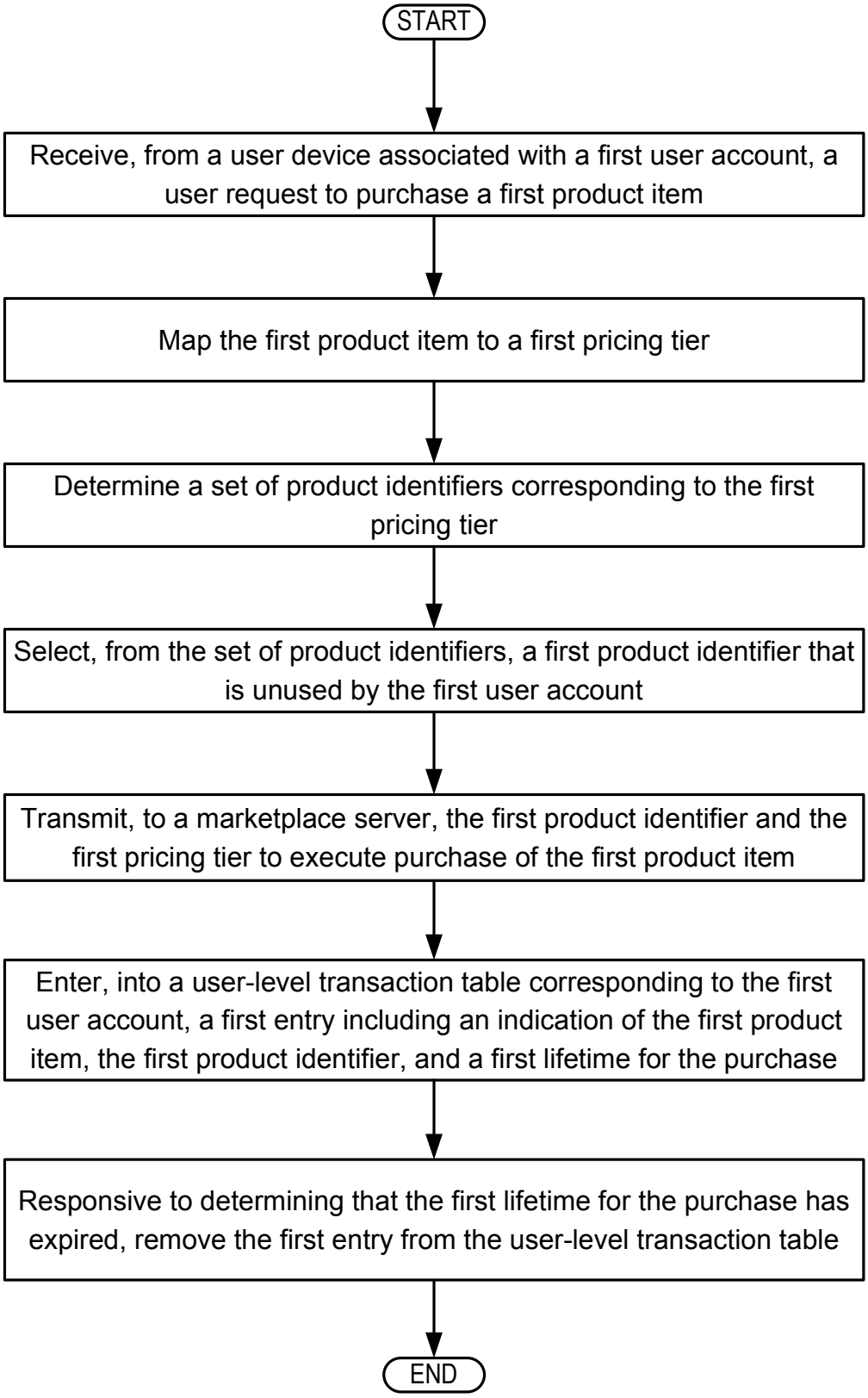
START
Receive, from a user device associated with a first user account, a user request to purchase a first product item
Map the first product item to a first pricing tier
Determine a set of product identifiers corresponding to the first pricing tier
Select, from the set of product identifiers, a first product identifier that is unused by the first user account
Transmit, to a marketplace server, the first product identifier and the first pricing tier to execute purchase of the first product item
Enter, into a user-level transaction table corresponding to the first user account, a first entry including an indication of the first product item, the first product identifier, and a first lifetime for the purchase
Responsive to determining that the first lifetime for the purchase has expired, remove the first entry from the user-level transaction table
END

FIG. 1

Product	Marketplace Product ID	Marketplace Price Tier
Product 0	ID 0	Tier a
Product 1	ID 1	Tier b
…	…	…
Product N	ID N	Tier c
Product N + 1	Unsupported	Tier d
…	Unsupported	…
Product M	Unsupported	Tier e

Table 1: Mapping from app-specific products to marketplace identifiers and pricing tiers

Marketplace Price Tier	USD	CAD	…	JPY
Tier 0	0.00	0.00		0
Tier 1	0.99	1.19		120
…				
Tier P	999.99	1199.99		118800

Table 2: Mapping from pricing tiers to currency-specific prices

Product	Marketplace Price Tier
Product 0	Tier a
Product 1	Tier b
…	…
Product M	Tier c

Table 3: App-level mapping of products to prices

Marketplace Price Tier	Marketplace Product ID
Tier 0	ID x, ID y,…
Tier 1	ID e, ID f, …
…	…
Tier P	ID q, ID r, …

Table 4: App-level mapping of prices to associated product identifiers

Purchased Product	Marketplace Product ID	Lifetime
Product a	ID x for Product a	Lifetime for Product a
Product b	ID y for Product b	Lifetime for Product b
…	…	…

Table 5: User-level table of purchased products

Product	Marketplace Price Tier
Channel 1	Tier 10
Channel 2	Tier 10
…	Tier 10
Channel 500K	Tier 10
Channel 500K+1	Tier 100
…	Tier 100
Channel 1M	Tier 100

Table 6: Media app mapping of channels to price tiers

Marketplace Price Tier	Marketplace Product ID
Tier 1	(empty)
…	Empty
Tier 10	ID 1, …, ID 10
…	Empty
Tier 100	ID 11, …, ID 20

Table 7: Media app mapping of prices to associated product identifiers

Purchased Product	Marketplace Product ID	Expiration
Channel 10	ID 1	30 days
Channel 999	ID 2	12 days
Channel 4	ID 3	14 days
Channel 5765	ID 4	30 days
Channel 990990	ID 20	28d
Channel 155	ID 5	1 day
Channel 8076	ID 11	5 days
Channel 750600	ID 18	12 days
Channel 500000	ID 12	23 days
Channel 1089	ID 6	8 days

Table 8: User-level table of purchased channel sponsorships