



(43) International Publication Date
9 August 2012 (09.08.2012)

(10) International Publication Number
WO 2012/106655 A2

- (51) International Patent Classification:
G06Q 20/40 (2012.01)
- (21) International Application Number:
PCT/US2012/023856
- (22) International Filing Date:
3 February 2012 (03.02.2012)
- (25) Filing Language: English
- (26) Publication Language: English
- (30) Priority Data:
61/439,879 5 February 2011 (05.02.2011) US
61/444,100 17 February 2011 (17.02.2011) US
- (71) Applicant (for all designated States except US): **VISA INTERNATIONAL SERVICE ASSOCIATION** [US/US]; 900 Metro Center Blvd., Foster City, California 94404-2172 (US).
- (72) Inventors; and
- (75) Inventors/Applicants (for US only): **KATZIN, Edward N.** [US/US]; 195 Bella Vista Drive, Hillsborough, CA 94010 (US). **KUMNICK, Phillip Lyle** [US/US]; 1607 East Briarwood Terrace, Phoenix, AZ 85048 (US). **HARRIS, Theodore David** [US/US]; 199 Tiffany Avenue, Unit 209, San Francisco, CA 94110 (US). **FAITH, Patrick** [US/US]; 2810 Jones Gate Court, Pleasanton, CA 94566 (US). **SCHULZ, Jennifer Ann** [US/US]; 876 Iliff Street, Pacific Palisades, CA 90272 (US).
- (74) Agent: **HANCHUK, Walter**; Chadbourne & Parke LLP, 30 Rockefeller Plaza, New York, New York 10112 (US).
- (81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.
- (84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).
- Published:
— without international search report and to be republished upon receipt of that report (Rule 48.2(g))

(54) Title: MERCHANT-CONSUMER BRIDGING PLATFORM APPARATUSES, METHODS AND SYSTEMS

(57) Abstract: The merchant-consumer bridging platform apparatuses, methods and systems ("MCB-Platform") various activity requests (e.g., transaction request, merchant information update request, offer issuance request, etc.) via MCB-Platform components into transaction records and merchant database updates outputs. In one implementation, a method is disclosed, comprising: receiving an activity request including merchant information associated with a merchant; retrieving a previously stored merchant record from a database; determining merchant information update indicia based on a comparison of the merchant information and the previously stored merchant record; determining a confidence metric for the merchant information update; retrieving a confidence requirement based on the activity request; determining, within a low-latency processing time-frame, whether the determined confidence metric satisfies the retrieved confidence requirement; performing the requested activity and updating the previously stored merchant record with the merchant information update indicia when the determined confidence metric satisfies the retrieved confidence requirement; and declining the activity request when the determined confidence metric satisfies the retrieved confidence requirement.



WO 2012/106655 A2

MERCHANT-CONSUMER BRIDGING PLATFORM APPARATUSES, METHODS AND SYSTEMS

[0001] This application for letters patent disclosure document describes inventive aspects directed at various novel innovations (hereinafter “disclosure”) and contains material that is subject to copyright, mask work, and/or other intellectual property protection. The respective owners of such intellectual property have no objection to the facsimile reproduction of the disclosure by anyone as it appears in published Patent Office file/records, but otherwise reserve all rights.

RELATED APPLICATIONS

[0002] Applicant hereby claims priority under 35 USC §119 for United States provisional patent application serial no. 61/439,879, filed February 5, 2011, entitled “Apparatuses, Methods And Systems For A Merchant-Consumer Bridging Platform,” attorney docket no. P-42003PRV|20270-102PV, and United States provisional patent application serial no. 61/444,100, filed February 17, 2011, entitled “Apparatuses, Methods And Systems For A Universal Value Exchange Merchant-Consumer Bridging Platform,” attorney docket no. P-42012PRV|20270-125PV.

[0003] The instant application is related to United States application serial no. 13/366,083, filed February 3, 2012, entitled “Merchant-Consumer Bridging Platform Apparatuses, Methods And Systems,” attorney docket no. P-42003US01|20270-102US.

[0004] The entire contents of the aforementioned applications are herein expressly incorporated by reference.

FIELD

1

2 **[0005]** The present innovations are directed generally to merchant promotion
3 distribution and redemption, and more particularly, to MERCHANT-CONSUMER
4 BRIDGING PLATFORM APPARATUSES, METHODS AND SYSTEMS.

BACKGROUND

5

6 **[0006]** Consumers may shop for products they are interested at a variety of
7 places, e.g., a supermarket, a department store, etc. For example, a consumer may go to
8 a supermarket to look for a desired product. He may find and pick up the desired
9 product at the supermarket, bring the product to a check-out lane, and pay for the
10 product at a point of sale (POS) terminal at the supermarket to complete the purchase.

11 **[0007]** Service providers such as banks and merchants run loyalty or rewards
12 programs to reward their customers for being loyal to their business, encourage more
13 spending, or entice new customers. These rewards may be in the form of points, cash
14 back, gift cards, miles, etc.

BRIEF DESCRIPTION OF THE DRAWINGS

15

16 **[0008]** The accompanying appendices and/or drawings illustrate various non-
17 limiting, example, innovative aspects in accordance with the present descriptions:

18 **[0009]** FIGURES 1A-1C provides block diagrams illustrating various example
19 embodiments of the MCB-Platform;

20 **[0010]** FIGURE 2A shows a block diagram illustrating data flows 200a between

1 MBC-Platform server and affiliated entities within various embodiments of the MCB-
2 Platform;

3 **[0011]** FIGURES 2B-2E show logic flow diagrams illustrating various
4 embodiments of the MCB-Platform;

5 **[0012]** FIGURE 3A provides a block diagram illustrating consumer merchant
6 checkout stack 300a within embodiments of the MCB-Platform;

7 **[0013]** FIGURES 3B-3C provide a data flow and a logic flow diagram illustrating
8 transaction at merchant POS within implementations of the MCB-Platform;

9 **[0014]** FIGURES 4A-4B provide a data flow and a logic flow diagram illustrating
10 mobile to mobile checkout within implementations of the MCB-Platform;

11 **[0015]** FIGURES 5A-5B provide a data flow and a logic flow diagram illustrating
12 card enrollment tokenization within implementations of the MCB-Platform;

13 **[0016]** FIGURES 5C-5D provide a data flow and a logic flow diagram illustrating
14 acquirer routing within implementations of the MCB-Platform;

15 **[0017]** FIGURES 6A provides a block diagram illustrating data flows of merchant
16 database updates within various embodiments of the MCB-Platform;

17 **[0018]** FIGURES 6B-6D provide a block diagram and logic flow diagrams
18 illustrating web claws for merchant information within various embodiments of the
19 MCB-Platform;

20 **[0019]** FIGURES 6E-6F provide logic flow diagrams illustrating merchant
21 database updates within various embodiments of the MCB-Platform;

22 **[0020]** FIGURES 6G-6H provide logic flow diagrams illustrating alternative

1 implementations of scoring mechanism within various embodiments of the MCB-
2 Platform;

3 **[0021]** FIGURES 7A-7L provide exemplary UIs illustrating consumer registration
4 within various embodiments of MCB-Platform;

5 **[0022]** FIGURES 8A-8H provide exemplary UIs illustrating consumer
6 registration and transaction with MCB-Platform within embodiments of the MCB-
7 Platform;

8 **[0023]** FIGURES 9A-9F provide exemplary UIs illustrating merchant distributing
9 offers over social media with MCB-Platform within embodiments of the MCB-Platform;

10 **[0024]** FIGURES 9G-9H provides exemplary mobile screens 900g-900h of offer
11 bridging via MCB-Platform within implementations of the MCB-Platform;

12 **[0025]** FIGURES 10A-B show block diagrams illustrating various example
13 embodiments of the MCB-Platform;

14 **[0026]** FIGURES 10C-D show data flow diagrams illustrating MCB-Platform
15 program configuration embodiment of the MCB-Platform;

16 **[0027]** FIGURES 11A-C show data flow diagram illustrating closed/open loop gift
17 card value exchange embodiments of the MCB-Platform;

18 **[0028]** FIGURES 12A-D show logic flow diagrams illustrating closed/open loop
19 gift card value exchange embodiments of the MCB-Platform;

20 **[0029]** FIGURE 13 shows a data flow diagram illustrating source/destination
21 value exchange embodiment of the MCB-Platform;

22 **[0030]** FIGURES 14A-B show logic flow diagrams illustrating source/destination

1 value exchange component embodiment of the MCB-Platform;

2 **[0031]** FIGURES 15A-B show logic flow diagrams illustrating equivalent value
3 determination component embodiment of the MCB-Platform;

4 **[0032]** FIGURE 16 shows a logic flow diagram illustrating cross-ecosystem
5 exchange component embodiment of the MCB-Platform;

6 **[0033]** FIGURES 17A-D show screenshot diagrams illustrating exchange mode
7 embodiments of the MCB-Platform;

8 **[0034]** FIGURE 17E shows screenshot diagrams illustrating exchange rate mode
9 embodiment of the MCB-Platform;

10 **[0035]** FIGURES 17F-I show screenshot diagrams illustrating management mode
11 embodiment of the MCB-Platform;

12 **[0036]** FIGURES 17J-K show screenshot diagrams illustrating MCB-Platform
13 point mode embodiment of the MCB-Platform;

14 **[0037]** FIGURES 17L-N show screenshot diagrams illustrating source/destination
15 exchange mode embodiment of the MCB-Platform;

16 **[0038]** FIGURE 18 shows a block diagram illustrating example aspects of a
17 centralized personal information platform in some embodiments of the MCB-Platform;

18 **[0039]** FIGURES 19A-F show block diagrams illustrating example aspects of data
19 models within a centralized personal information platform in some embodiments of the
20 MCB-Platform;

21 **[0040]** FIGURE 20 shows a block diagram illustrating example MCB-Platform
22 component configurations in some embodiments of the MCB-Platform;

1 **[0041]** FIGURE 21 shows a data flow diagram illustrating an example search
2 result aggregation procedure in some embodiments of the MCB-Platform;

3 **[0042]** FIGURE 22 shows a logic flow diagram illustrating example aspects of
4 aggregating search results in some embodiments of the MCB-Platform, e.g., a Search
5 Results Aggregation (“SRA”) component 500;

6 **[0043]** FIGURES 23A-D show data flow diagrams illustrating an example card-
7 based transaction execution procedure in some embodiments of the MCB-Platform;

8 **[0044]** FIGURES 24A-E show logic flow diagrams illustrating example aspects of
9 card-based transaction execution, resulting in generation of card-based transaction data
10 and service usage data, in some embodiments of the MCB-Platform, e.g., a Card-Based
11 Transaction Execution (“CTE”) component 700;

12 **[0045]** FIGURE 25 shows a data flow diagram illustrating an example procedure
13 to aggregate card-based transaction data in some embodiments of the MCB-Platform;

14 **[0046]** FIGURE 26 shows a logic flow diagram illustrating example aspects of
15 aggregating card-based transaction data in some embodiments of the MCB-Platform,
16 e.g., a Transaction Data Aggregation (“TDA”) component 900;

17 **[0047]** FIGURE 27 shows a data flow diagram illustrating an example social data
18 aggregation procedure in some embodiments of the MCB-Platform;

19 **[0048]** FIGURE 28 shows a logic flow diagram illustrating example aspects of
20 aggregating social data in some embodiments of the MCB-Platform, e.g., a Social Data
21 Aggregation (“SDA”) component 1100;

1 **[0049]** FIGURE 29 shows a data flow diagram illustrating an example procedure
2 for enrollment in value-add services in some embodiments of the MCB-Platform;

3 **[0050]** FIGURE 30 shows a logic flow diagram illustrating example aspects of
4 social network payment authentication enrollment in some embodiments of the MCB-
5 Platform, e.g., a Value-Add Service Enrollment (“VASE”) component 1300;

6 **[0051]** FIGURES 31A-B show flow diagrams illustrating example aspects of
7 normalizing aggregated search, enrolled, service usage, transaction and/or other
8 aggregated data into a standardized data format in some embodiments of the MCB-
9 Platform, e.g., a Aggregated Data Record Normalization (“ADRN”) component 1400;

10 **[0052]** FIGURE 32 shows a logic flow diagram illustrating example aspects of
11 recognizing data fields in normalized aggregated data records in some embodiments of
12 the MCB-Platform, e.g., a Data Field Recognition (“DFR”) component 1500;

13 **[0053]** FIGURE 33 shows a logic flow diagram illustrating example aspects of
14 classifying entity types in some embodiments of the MCB-Platform, e.g., an Entity Type
15 Classification (“ETC”) component 1600;

16 **[0054]** FIGURE 34 shows a logic flow diagram illustrating example aspects of
17 identifying cross-entity correlation in some embodiments of the MCB-Platform, e.g., a
18 Cross-Entity Correlation (“CEC”) component 1700;

19 **[0055]** FIGURE 35 shows a logic flow diagram illustrating example aspects of
20 associating attributes to entities in some embodiments of the MCB-Platform, e.g., an
21 Entity Attribute Association (“EAA”) component 1800;

1 **[0056]** FIGURE 36 shows a logic flow diagram illustrating example aspects of
2 updating entity profile-graphs in some embodiments of the MCB-Platform, e.g., an
3 Entity Profile-Graph Updating (“EPGU”) component 1900;

4 **[0057]** FIGURE 37 shows a logic flow diagram illustrating example aspects of
5 generating search terms for profile-graph updating in some embodiments of the MCB-
6 Platform, e.g., a Search Term Generation (“STG”) component 2000;

7 **[0058]** FIGURE 38 shows a user interface diagram illustrating an overview of
8 example features of virtual wallet applications in some embodiments of the MCB-
9 Platform;

10 **[0059]** FIGURES 39A-G show user interface diagrams illustrating example
11 features of virtual wallet applications in a shopping mode, in some embodiments of the
12 MCB-Platform;

13 **[0060]** FIGURES 40A-F show user interface diagrams illustrating example
14 features of virtual wallet applications in a payment mode, in some embodiments of the
15 MCB-Platform;

16 **[0061]** FIGURE 41 shows a user interface diagram illustrating example features of
17 virtual wallet applications, in a history mode, in some embodiments of the MCB-
18 Platform;

19 **[0062]** FIGURES 42A-E show user interface diagrams illustrating example
20 features of virtual wallet applications in a snap mode, in some embodiments of the
21 MCB-Platform;

1 **[0063]** FIGURE 43 shows a user interface diagram illustrating example features
2 of virtual wallet applications, in an offers mode, in some embodiments of the MCB-
3 Platform;

4 **[0064]** FIGURES 44A-B show user interface diagrams illustrating example
5 features of virtual wallet applications, in a security and privacy mode, in some
6 embodiments of the MCB-Platform;

7 **[0065]** FIGURE 45 shows a data flow diagram illustrating an example user
8 purchase checkout procedure in some embodiments of the MCB-Platform;

9 **[0066]** FIGURE 46 shows a logic flow diagram illustrating example aspects of a
10 user purchase checkout in some embodiments of the MCB-Platform, e.g., a User
11 Purchase Checkout (“UPC”) component 4600;

12 **[0067]** FIGURES 47A-B show data flow diagrams illustrating an example
13 purchase transaction authorization procedure in some embodiments of the MCB-
14 Platform;

15 **[0068]** FIGURES 48A-B show logic flow diagrams illustrating example aspects of
16 purchase transaction authorization in some embodiments of the MCB-Platform, e.g., a
17 Purchase Transaction Authorization (“PTA”) component 4800;

18 **[0069]** FIGURES 49A-B show data flow diagrams illustrating an example
19 purchase transaction clearance procedure in some embodiments of the MCB-Platform;

20 **[0070]** FIGURES 50A-B show logic flow diagrams illustrating example aspects of
21 purchase transaction clearance in some embodiments of the MCB-Platform, e.g., a
22 Purchase Transaction Clearance (“PTC”) component 5000; and

1 **[0071]** FIGURE 51 shows a block diagram illustrating embodiments of a MCB-
2 Platform controller.

3 **[0072]** The leading number of each reference number within the drawings
4 indicates the figure in which that reference number is introduced and/or detailed. As
5 such, a detailed discussion of reference number 101 would be found and/or introduced
6 in Figure 1. Reference number 201 is introduced in Figure 2, etc.

7 DETAILED DESCRIPTION

8 **[0073]** The MERCHANT-CONSUMER BRIDGING PLATFORM APPARATUSES,
9 METHODS AND SYSTEMS (hereinafter “MCB-Platform”) provides a merchant-
10 consumer bridging platform, whereby merchants and consumer electronic wallet
11 accounts are registered with the MCB-Platform to facilitate consumer targeted offer
12 distribution, redemption and payment during a purchasing transaction. In one
13 implementation, the MCB-Platform may maintain a merchant database storing
14 merchant information to provide matching offers to a consumer. In one
15 implementation, the MCB-Platform may monitor and update merchant information
16 such as, but not limited to merchant inventory, merchant product category, merchant
17 geographical location, merchant business promotion, and/or the like.

18 **[0074]** In one embodiment, a consumer may register his smartphone (e.g., an
19 Apple iPhone) with an electronic wallet service on the MCB-Platform. The consumer
20 may receive discount information, electronic coupons, offers, rewards, etc., from an
21 enrolled merchant website via emails, text messages, and/or the like. For example, if
22 McDonalds is an enrolled merchant at MCB-Platform, a registered consumer may

1 receive a "\$2.99 happy meal" SMS from the MCB-Platform, and may walk to a
2 McDonalds store to purchase the "\$2.99 happy meal" by providing his electronic wallet
3 information, e.g., by swiping a MCB-Platform magstripe card, by engaging a NFC
4 contactless handshake via his smartphone, and/or the like.

5 **[0075]** For another example, a consumer may visit a merchant's website, and
6 select a desired product from the merchant's website to associate the product to his
7 electronic wallet account. In one implementation, the consumer may pick up the
8 product at an enrolled merchant store and the MCB-Platform may automatically process
9 the payment with the enrolled merchant store, upon submission of the consumer's
10 electronic wallet information. In further implementations, the consumer may engage in
11 MCB-Platform in multiple channels, such as, but not limited to mobile
12 networks/devices, social networks, and/or the like.

13 **[0076]** In a further embodiment, the MCB-Platform may facilitate a merchant
14 collecting information related to consumer's purchasing habits, preferences of products,
15 and/or the like, based on which the merchants may design targeted campaign programs
16 and marketing promotions. For another example, the MCB-Platform may facilitate
17 loyalty/affinity programs for merchants. In one implementation, the merchant registry
18 at a MCB-Platform database may match a consumer loyalty/affinity program
19 membership, which facilitates delivery of offers, rewards and/or the like, to consumers.
20 For example, a merchant may launch a "15% off invitation" offer to attract first-time
21 consumers, and a "25% off loyalty" offer for returning consumers.

1 **[0077]** FIGURES 1A-1C provides block diagrams illustrating various example
2 embodiments of the MCB-Platform. FIGURE 1A shows exemplary aspects of matching
3 merchant offers to consumers based on consumer opt-in activities 100a within
4 implementations of the MCB-Platform. In one implementation, a consumer may engage
5 in a plurality of opt-in activities 105 that reflect his preferences and interests in
6 consumer goods, purchasing patterns, and/or the like. For example, the consumer
7 "John Smith" may tweet his purchase of a Starbucks coffee, e.g., at 118a; such social
8 media activities 105a may be captured by the MCB-Platform to track consumer
9 behaviors. For another example, the MCB-Platform may obtain information as to "John
10 Smith's" membership with Starbucks, as he added a Starbucks card to his electronic
11 wallet, showing his loyalty and constant consumption habits 105b in Starbucks coffee.
12 For another example, the MCB-Platform 120 may obtain GPS coordinates of "John
13 Smith" 105c (e.g., via consumer's smartphone, etc.), based on which the MCB-Platform
14 may recommend offers from nearby merchant stores to the consumer. The consumer
15 opt-in activities 105 may comprise a variety of other activities, such as, but not limited to
16 consumer browsing history, in-store transaction history, shopping cart history, wish list,
17 various social media activities (e.g., "like" or comment on a merchant link or product on
18 Facebook, etc.), blog experiences, and/or the like.

19 **[0078]** In one implementation, the MCB-Platform 120, upon obtaining the
20 consumer opt-in activity data 105, may find merchant offers that match with the
21 consumer's interest 106. For example, the MCB-Platform 120 may determine "John
22 Smith" is interested in coffee products 103a based on his constant purchasing at
23 Starbucks. In one implementation, the MCB-Platform 120 may prompt matched offers
24 107 to the user, e.g., via his mobile wallet at 108. Such offers may comprise a coupon of

1 a coffee shop located close to "John Smith's" location based on his GPS coordinates,
2 which matches the consumer's interests in coffee 103b.

3 **[0079]** FIGURE 1B provides a diagram illustrating aspects of merchant-consumer
4 cloud 100b within embodiments of the MCB-Platform. In one implementation, the
5 MCB-Platform cloud 120 may obtain various information into the cloud. For example,
6 MCB-Platform may obtain consumer generated content 112a, such as consumer user
7 profile, social media posts, comments, "like" activities, purchasing history, wish list,
8 mobile wallet information, loyalty program information, web browsing history, and/or
9 the like. For another example, the MCB-Platform may obtain merchant generated
10 content 112b, such as but not limited to merchant profile information, merchant store
11 location, merchant inventory information, merchant website server ID, merchant web
12 server address, merchant terminal ID, merchant terminal transaction history, and/or
13 the like. Such obtained consumer and/or merchant contents 112a/b may provide
14 information as to consumer's interests in merchant products, and the most update
15 merchant offers and promotions.

16 **[0080]** In further implementations, the MCB-Platform cloud 120 may obtain
17 various information from the Internet and/or the cloud, such as music 113a, video 113b,
18 photos 113c, applications and documents 113d, shopping experiences 113e, and/or the
19 like. Within implementations, such contents may provide insights to the MCB-Platform
20 cloud 120 on merchant updates, e.g., a news article reporting a merchant store's grand
21 opening, a Facebook advertisement of a merchant, etc. Further implementations of
22 MCB-Platform web claws are described in FIGURES 6A-6D.

23 **[0081]** FIGURE 1C provides a diagram illustrating aspects of merchant-consumer

1 cloud update 100c within embodiments of the MCB-Platform. In one implementation,
2 as shown in FIGURE 1C.(a), the MCB-Platform cloud 120 may receive information as to
3 the merchant store location, e.g., an address line from the merchant "Starbucks" website
4 110b indicates the closest store at zipcode "00000" is at "1332 Dream Street" 115. The
5 MCB-Platform may compare this address with the previously stored merchant profile
6 and find inconsistency, as the stored address is at "1400 Dream Street" 116. The MCB-
7 Platform cloud 120 may not update or change the stored merchant address immediately
8 as it is uncertain which information is accurate 118, but instead may determine how
9 much confidence it has on the received information 115 from the merchant site 110b.
10 For example, a confidence score may be generated associated with the received
11 information 115, as further discussed in FIGURES 6 and 19A-19F.

12 **[0082]** Continuing on in FIGURE 1C. (b), the MCB-Platform cloud 120 may
13 receive further indications of address change of the Starbucks store, e.g., receiving a
14 consumer transaction record 121a indicating a purchase of Starbucks coffee at "1332
15 Dream Street" 119a, and another consumer transaction record 121b showing redemption
16 of Starbucks coupon at the same address 119b. Such information may enhance the
17 confidence level that the Starbucks store location should be at "1332 Dream Street"
18 instead of the previously stored "1400 Dream Street," e.g., at 122. The MCB-Platform
19 may then update the stored merchant profile of Starbucks to correct the store location
20 information, e.g., at 123.

21 **[0083]** FIGURE 2A shows a block diagram illustrating data flows 200a between
22 MBC-Platform server and affiliated entities within various embodiments of the MCB-
23 Platform. Within various embodiments, one or more consumers user(s) 202, MCB-

1 PLATFORM server 220, MCB-PLATFORM database(s) 219, merchant store(s) 210,
2 mobile carrier 225, financial network(s)/system(s) 230, merchant website(s) and/or
3 social media 250 are shown to interact via various communication network 213.

4 **[0084]** In one embodiment, a consumer 202, may be associated with an electronic
5 wallet 203, which may comprise one or more of a bank account, a MCB-Platform service
6 account, a merchant membership account, and/or the like, possessed with the consumer
7 202. For example, a consumer may possess an electronic wallet linked a Bank of
8 America checking account, a Chase credit card account, a Sam's Club membership
9 account, and/or the like. For another example, the consumer's electronic wallet may be
10 registered for the MCB-Platform service.

11 **[0085]** In one embodiment, upon registering with a MBC-Platform, the consumer
12 202 may visit a participating merchant store's website and obtain product information
13 218. For example, the consumer 202 may browse a merchant's online catalogue, a third
14 party shopping website featuring the merchant's product (e.g., Amazon.com,
15 Zappos.com, etc.) and/or visit product advertisement via social media 250 (e.g.,
16 Facebook, Twitter, etc.). The consumer 203 may submit an online purchase request
17 and/or a subscription to a desired product 213. For example, a consumer device (e.g., a
18 browser running on a consumer smartphone, computers, etc.) may generate a Hypertext
19 Transfer Protocol Secure (HTTPS) POST message to make a subscription/purchase
20 request including the desired item information in the form of data formatted according
21 to the XML. Below is an example HTTP(S) POST message including an XML-formatted
22 subscription message 211 for the MCB-Platform server:

23 POST /SubscriptionRequest.php HTTP/1.1
24 Host: 206.205.82.130

```

1   Content-Type: Application/XML
2   Content-Length: 418
3   <?XML version = "1.0" encoding = "UTF-8"?>
4   <SubscriptionRequest>
5     <Time> 12:30:30 </Time>
6     <Date> 10-10-2015 </Date>
7     <MerchantID> Amazon </MerchantID>
8     <MerchantName> Amazon.com </MerchantName>
9     <MerchantURL> www.amazon.com </MerchantURL>
10    <PickupLocation> 1332 Dream Street </PickupLocation>
11    <StoreName> Starbucks
12    ...
13    <TransactionType> Wish List </TransactionType>
14    ...
15    <ItemID> BF00001 </ItemID>
16    <ItemName> Starbucks Coffee Mug </ItemName>
17    <ItemPrice> $15.99 </ItemPrice>
18    ...
19    <Consumer>
20      <ID> JS001 </ID>
21      <Name> John Smith </Name>
22      <CardNo> 0000 0000 0000 0000 </CardNo>
23      <WalletID> JS-W-001 </WalletID>
24      ...
25    </Consumer>
26  </SubscriptionRequest>
27

```

28 **[0086]** The merchant website 250 may synchronize the purchase request 253
 29 (e.g., which may take a similar form to the subscription request 211) to a merchant store
 30 210 for consumer's pickup, and may also provide the consumer a geographical location
 31 of a merchant store that features the desired product.

32 **[0087]** In one embodiment, the consumer 202 may provide his MCB-Platform
 33 wallet information 207 to a merchant store 210 prior to his check-out. For example, in
 34 one implementation, the consumer may swipe a MBC-Platform membership magstripe
 35 card at a POS terminal of the merchant store. For another example, the consumer may
 36 operate a smart phone for registration with the POS via short messages. For another
 37 example, the consumer may register with the merchant via bar code scan of the
 38 consumer's MBC-Platform membership card and/or the product.

39 **[0088]** In one embodiment, the merchant store 210 may obtain the "wallet"

1 information 207 at its POS terminal, which may comprise the consumer's wallet account
2 information (e.g., a wallet ID, the associated bank information, etc.), the product
3 reservation information, and/or the like. The merchant store 210 may generate a
4 payment request 250 to a MCB-Platform server, wherein the payment request may
5 comprise merchant store/terminal identification information, consumer wallet
6 identification information, a payment amount, and/or the like. For example, the
7 merchant terminal 210 may generate a HTTPS POST message to make a payment
8 request including the desired item information in the form of data formatted according
9 to the XML. Below is an example HTTP(S) POST message including an XML-formatted
10 payment request message 250 for the MCB-Platform server:

```
11 POST /SubscriptionRequest.php HTTP/1.1
12 Host: 206.205.82.135
13 Content-Type: Application/XML
14 Content-Length: 418
15 <?XML version = "1.0" encoding = "UTF-8"?>
16 <PaymentRequest>
17   <Time> 18:30:30 </Time>
18   <Date> 10-10-2015 </Date>
19   <MerchantID> STUX </MerchantID>
20   <MerchantName> Starbucks </MerchantName>
21   <ReferredURL> www.amazon.com </ReferredURL>
22   <TerminalID> ST0001 </TerminalID>
23   ...
24   <TransactionType> Purchase </TransactionType>
25   ...
26   <Item>
27     <ItemID> BF00001 </ItemID>
28     <ItemName> Starbucks Coffee Mug </ItemName>
29     <ItemPrice> $25.99 </ItemPrice>
30     ...
31   </Item>
32   ...
33   <Consumer>
34     <ID> JS001 </ID>
35     <Name> John Smith </Name>
36     <CardNo> 0000 0000 0000 0000 </CardNo>
37     <CardType> Visa </CardType>
38     <WalletID> JS-W-001 </WalletID>
39     ...
40   </Consumer>
41   <BIN> 0000000r </BIN>
42   ...
43 </PaymentRequest>
44
```

1 **[0089]** In one embodiment, the MCB-Platform server 220 may process the
2 payment request, and communicate with a financial network 230 to exchange financial
3 data 233b to perform the financial transaction (as further illustrated in FIGURES 3B-
4 3C). In another implementation, the MCB-Platform server 220 may be integrated with
5 a financial payment platform.

6 **[0090]** In one embodiment, the MCB-Platform server 220 may send payment
7 approval 255 (e.g., see also 316b in FIGURE 3B) to the merchant store POS terminal 210
8 and the consumer 202 to notify the completion of the payment. In one implementation,
9 the MCB-Platform may send the merchant information 260, and the transaction
10 approval details to a mobile carrier 225 wherein the consumer is a subscriber. The
11 mobile carrier 225 may send the transaction information and merchant information 60
12 to the consumer 202 via text messages, emails, customer service robot calls, and/or the
13 like.

14 **[0091]** In one embodiment, the MCB-Platform server 220 may establish data
15 records of registered consumers, merchants, past transactions 223 for storage in a
16 database 219. A merchant registry at the MCB-Platform may comprise data entries
17 such as, but not limited to merchant ID, merchant URL, URI, US DMA, MSA, NAICS
18 codes, position coordinates, latitude, longitude, consumer preferences, opt-in activities,
19 history, offer notifications, messaging campaign settings, campaign management, offer
20 delivery, messaging, redemption, analytics, and/or the like.

21 **[0092]** For example, an exemplar XML code of a merchant record may take a
22 form similar to the following:

23 POST /Merchant.php HTTP/1.1
24 Host: 206.205.82.135

```

1  Content-Type: Application/XML
2  Content-Length: 418
3  <?XML version = "1.0" encoding = "UTF-8"?>
4  <Merchant>
5      <MerchantID> 123456789 </MerchantID>
6      <MerchantName> All Grocery </MerchantName>
7      </MerchantAddress> 111 White road </MerchantAddress>
8      <MerchantGPSInfo> N 66666 W 7777777 </MerchantGPSInfo>
9      ...
10     </MerchantTerminalID>11111111</MerchantTerminalID>
11     <MerchantLicenseInfo>
12         ....
13     </MerchantLicenseInfo>
14     <MerchantWebsite> www.allGrocery.com </MerchantWebsite>
15     <MerchantParticipatingSite>
16         <Site1>
17             www.amazon.com
18         </Site1>
19         ...
20     </MerchantParticipatingSite>
21     <MerchantProduct>
22         <Product1>
23             <ProductID> 00023213 </ProductID>
24             <ProductBrand> Green Farm </ProductBrand>
25             <ProductBarCode> ... </ProductBarCode>
26             ...
27         </Product1>
28         ...
29     </MerchantProduct>
30     ...
31 </Merchant>
32

```

33 **[0093]** Further implementations of the database 219 are discussed in FIGURE 3.

34 **[0094]** In a further implementation, the MCB-Platform may support wholesale
 35 API delivery of embodiments of the MCB-Platform.

36 **[0095]** FIGURE 2B provides a logic flow 200b diagram illustrating embodiments
 37 of the MCB-Platform. In one embodiment, the consumer 202 and a merchant 210 may
 38 register to the MCB-Platform 220 prior to an in-store purchase. For example, in one
 39 implementation, the consumer 202 may submit identifying information (e.g. consumer
 40 name, address, phone number, email address, social media account, and/or the like)
 41 and financial information to associate his bank account information, merchant
 42 membership information and/or the like 205 to the MCB-Platform 220 to create an
 43 electronic wallet. In another implementation, the merchant 210 may enroll in the MCB-

1 Platform 220 by providing its identification information, geographical location
2 information, website URL information, and/or the like. In one implementation, the
3 merchant may be registered as a brand 208, e.g., the brand "GAP" may be registered
4 associated with its retail stores, etc, . In another implementation, a POS terminal at a
5 merchant may be registered separately at the MCB-Platform, e.g., one terminal at Whole
6 Food, Inc., located at 10110 White Street, etc..

7 **[0096]** In one implementation, upon registration 209, a merchant may bridge
8 with consumers in a variety of electronic wallet vehicles. For example, the merchant
9 may cooperate with carriers to provide smartphone applications for NFC handshakes.
10 For another example, a merchant may equip MCB-Platform products barcode/NFC
11 plate reading machines at its POS terminals. For another example, a merchant may
12 accept magstripe cards to provide electronic wallet information.

13 **[0097]** In one embodiment, a consumer may browse a merchant website to
14 reserve a desired product online 212. In another implementation, a consumer may
15 retrieve coupon information from the merchant website, e.g., a "\$2.99 happy meal"
16 from McDonalds, etc. In another implementation, a consumer may click a product
17 advertisement on a social media platform (e.g., Facebook, Twitter, etc.) and select to
18 reserve a product.

19 **[0098]** The merchant web-server may generate a pre-order 221 and transmit the
20 tentative pre-order information 222 to the MCB-Platform. For example, the pre-order
21 information may comprise product information, consumer wallet information, and/or
22 the like. In one implementation, the MCB-Platform 220 may form a query based on the
23 product information for a list of locations of merchant stores that feature the product,

1 e.g., at 224a. In a further implementation, the consumer information may comprise
2 GPS information of the consumer, e.g., the consumer may operate a GPS-enabled smart
3 phone (e.g., Apple iPhone, etc.) to submit the pre-order; the consumer wallet
4 information may comprise one or more consumer's home addresses, etc.. The MCB-
5 Platform may then provide a list of merchant stores ranked by the distance to the
6 consumer's location, e.g., 224b.

7 **[0099]** In one implementation, at a merchant store, the consumer may submit
8 consumer's wallet information 232. For example, while waiting in the line for check-
9 out, the consumer may swipe a magstripe card at a POS terminal. For another example,
10 the consumer wallet information may be submitted via a bar code reading and/or NFC
11 handshake machine.

12 **[00100]** In one implementation, the MCB-Platform may receive consumer wallet
13 information from a merchant store, and then retrieve the consumer's pre-order
14 information, and verify the consumer/merchant status 235. For example, in one
15 implementation, upon receiving consumer wallet information, the MCB-Platform may
16 confirm the wallet holder (e.g., the consumer) is physically present at a merchant store,
17 and also confirm the merchant store is an MCB-Platform service enrolled acceptor. In a
18 further implementation, the MCB-Platform may verify whether restriction may be
19 applied for the MCB-Platform wallet service. For example, a consumer may specify a
20 maximum payment amount allowable in MCB-Platform purchase to improve security.

21 **[00101]** In one embodiment, the merchant store may generate a transaction
22 payment order at a POS terminal 238 based on the pre-order information associated
23 with the consumer wallet. The consumer may then verify whether the payment amount

1 is correct, e.g., whether the payment amount matches the amount that was offered
2 online, etc.. If the consumer agrees with the payment amount, he may enter an amount
3 in his electronic wallet 241 to purchase. For example, the consumer may choose a
4 variety of payment methods, such as, but not limited to mobile pay with electronic cash
5 register (ECR), and/or the like. If the amount is not correct, e.g., a consumer receive a
6 payment of "\$4.56" for an order of "\$2.99 happy meal" at McDonalds, etc., the
7 consumer may request customer service 242.

8 **[00102]** In one embodiment, when the MCB-Platform receives consumer's
9 payment notification, the MCB-Platform may confirm payment ability of the consumer
10 and initiate payment 245. For example, the MCB-Platform may communicate with a
11 financial account of the consumer to check payment ability, and if a checking amount
12 with insufficient funds is associated with the wallet, the MCB-Platform may reject the
13 transaction.

14 **[00103]** In one embodiment, the MCB-Platform may send payment approval to the
15 merchant store 251 to approve the transaction, and the consumer may receive a
16 purchase receipt 255 from the merchant. In another example, the consumer may
17 receive a report of transaction from the MCB-Platform via emails, text messages, and/or
18 the like.

19 **[00104]** FIGURE 2C provides a logic flow diagram 200c illustrating alternative
20 embodiments of the MCB-Platform. In one embodiment, a consumer 102 may engage
21 in opt-in activities 261, such as, but not limited to POS sale transactions with a
22 credit/debit card, online purchase on a merchant website using electronic wallet
23 account, and/or the like. The merchant store 110, and/or the merchant website 150 (or

1 a third party shopping site carrying the merchant's products) may collect information
2 related to the consumer's opt-in activities and send such information 262 to the MCB-
3 Platform 120.

4 **[00105]** In one embodiment, the MCB-Platform 120 may form a query for
5 merchant offers based on received information 263. For example, in one
6 implementation, if the received information indicates the consumer engaged in online
7 grocery purchase and delivery, the MCB-Platform may query for online grocery delivery
8 offers from the merchants. The MCB-Platform may then determine whether there is a
9 consumer device available, e.g., whether the consumer has a registered phone number,
10 email, membership card, and/or the like. If yes, the MCB-Platform may send a list of
11 matched offers to the consumer 264. Otherwise if not, the MCB-Platform may send the
12 matched offers to the merchant store 265.

13 **[00106]** In one embodiment, upon receiving a list of matched offers, the consumer
14 102, and/or the merchant store 110 may determine whether there is any indication of
15 redeeming the offer. If yes, the offer may be redeemed by the consumer 266 and/or the
16 merchant store 267. For example, the consumer may receive an offer code for "10% OFF
17 grocery delivery" via SMS, and he may redeem the offer by entering the offer code at the
18 check-out page of ordering grocery delivery online. For another example, the merchant
19 store may receive an indication of redeem the offer "10% OFF grocery delivery" when
20 the consumer proceeds at check-out by providing his wallet information, and the
21 merchant may redeem the offer by applying the discount for the consumer.

22 **[00107]** In one implementation, the MCB-Platform 120 may update the transaction
23 record 268, recording information such as, but not limited to consumer information,

1 product information, transaction time, transaction amount, offer redeemed, and/or the
2 like.

3 **[00108]** FIGURE 2D provides a logic flow 200d illustrating matching offers for
4 consumers within one embodiment of the MCB-Platform. In one embodiment, the
5 MCB-Platform may retrieve a stored record of consumers 268. For each consumer entry
6 270, the MCB-Platform may confirm the consumer status as opt-in 271, e.g., the
7 consumer is actively engaged in opt-in activities such as, but not limited to online
8 purchase, POS sales with a credit card, and/or the like. The MCB-Platform may then
9 generate a query based on the consumer opt-in information to match merchant offers
10 272. In one implementation, the MCB-Platform may extract information indicative of
11 consumer preferences from the consumer opt-in information. For example, if a
12 consumer's opt-in activities reflect a pattern of shopping for organic grocery products,
13 the MCB-Platform may store this consumer preference information and match
14 merchant offers related to organic products for the consumer.

15 **[00109]** In one implementation, the MCB-Platform may retrieve matched
16 merchant offers based on specified heuristics 273. For example, the list of matched
17 merchant offers may be sorted by the top sponsored merchants. For another example,
18 the list of matched offers may be sorted by relevance.

19 **[00110]** In one implementation, the MCB-Platform may send the list of matched
20 offers to the selected consumer 274, such as, but not limited to emails, SMS, customer
21 service calls, and/or the like. The MCB-Platform may then de-queue the consumer
22 entry, and proceed with the next consumer entry in the consumer record.

23 **[00111]** In one implementation, the MCB-Platform may update the matched offers

1 for each consumers periodically, e.g., on a daily basis, and/or the like.

2 **[00112]** FIGURE 2E provides a logic flow 200e illustrating alternative
3 embodiments of the MCB-Platform. In one embodiment, the MCB-Platform may visit
4 every merchant site entry 280 stored in the database 280, and retrieve an associated
5 merchant URL site 281. The MCB-Platform may then determine whether there is a
6 MCB-Platform service enabled POS terminal associated with the merchant 282. For
7 example, a merchant entry may be a solely online shopping site without a physical store,
8 which does not have a POS terminal. In that case, the MCB-Platform may proceed with
9 the next merchant entry.

10 **[00113]** If there is a POS terminal associated with the merchant, the MCB-Platform
11 may parse the merchant URL page and extract information such as, but not limited to IP
12 address, country origin, title, and other HTML tags 283, and store the parsed
13 information into a merchant database 284.

14 **[00114]** In one implementation, the MCB-Platform may retrieve stored
15 information 285 for merchant offer matching. The MCB-Platform may parse the stored
16 values to query the merchant ID database 286. For example, for a shopping website
17 selling brand shoes, the MCB-Platform may parse information from the webpage such
18 as the brand names, the shoe type names, and/or the like, and form a query to find
19 merchant stores that carry the shoes.

20 **[00115]** In one implementation, upon obtaining a list of merchant stores from the
21 query, the MCB-Platform may determine whether there is a MCB-Platform service
22 enabled POS associated with each merchant ID. If not, the MCB-Platform may proceed
23 with the next merchant ID. If yes, the merchant may verify additional heuristics 287,

1 such as, but not limited to whether the merchant is a top sponsor of MCB-Platform,
2 whether the merchant is the most relevant, and/or the like.

3 **[00116]** In one implementation, the MCB-Platform may bind the matched
4 merchant ID with parsed merchant webpage record 288, and retrieve geographic
5 information associated with the merchant ID 289.

6 **[00117]** In a further implementation, the MCB-Platform may create and/or update
7 a merchant campaign profile with the geo-location of a merchant store and the stored
8 parsed merchant page 290. For example, for a merchant brand "GAP," the MCB-
9 Platform may associate its campaign program profile with the queried geo-locations of
10 merchant stores, such as "GAP" stores, department stores that carries "GAP" products,
11 and/or the like, and the parsed information from "www.gap.com".

12 **[00118]** In a further implementation, the MCB-Platform may devise merchant
13 campaign programs based on consumer opt-in activities heuristics. For example, the
14 MCB-Platform may send individual in-store coupons to a consumer for the merchant
15 "GAP" at a fixed "GAP" store, if the consumer opt-in activities show a regular purchasing
16 pattern at the fixed "GAP" store location.

17 **[00119]** FIGURE 3A provides a block diagram illustrating consumer merchant
18 checkout stack 300a within embodiments of the MCB-Platform. Within
19 implementations, a merchant 310 POS terminal and a consumer 302 may operate a
20 variety of devices (e.g., 310.1 ~ 310.4) for payment transactions via different protocols
21 303.

22 **[00120]** For example, in one implementation, an employee checkout terminal 310.1
23 (e.g., with an electronic cash register, etc.) may be installed with the merchant 310, e.g.,

1 connected via analog dial or IP. If the merchant 310 is with no NFC reader, it may
2 receive payment information from the consumer 302 by reading a magnetic strip card
3 302.1. In one implementation, the employee checkout terminal 310.1 may be connected
4 to a payment network via TC40 303a.

5 **[00121]** In another implementation, the merchant 310 may employ a self-checkout
6 terminal 310.2 which may comprise a NFC component to receive payment information
7 from NFC handshake 303b, e.g., from a RFID card 302.2, etc. The POS terminal 310.2
8 may be equipped with radio component, such as NFC-296/896 Antenna Tuning Unit
9 (ATU), and/or the like).

10 **[00122]** In another implementation, the merchant 310 may employ feature phone
11 checkout terminal 310.4. For example, a store employee may operate a feature phone to
12 checkout consumer's purchases. In one implementation, the consumer may operate a
13 feature phone 302.4 to checkout, which may send and receive SMS via a cellular
14 network from the merchant feature phone 310.4. In further implementations, the
15 feature phone may be equipped with a card reader plug-in 307a, and/or a NFC 307b
16 component, e.g., the feature phone PANDA N1, etc., so that the consumer's magnetic
17 card 302.1 and RFID card 302.2 may be read by the feature phone terminal 310.4, or the
18 consumer feature phone 302.4. Within implementations, the feature phone checkout
19 may be operated via cellular communication networks, TON, audio phone
20 communication 303c, and/or the like.

21 **[00123]** In another implementation, the merchant 310 may employ a smartphone
22 checkout terminal 310.3. For example, a store employee may operate a smart phone
23 310.3 to checkout consumer's purchases. For another example, the consumer 302 may

1 operate a smartphone 302.3 which has an electronic wallet application for checkout.
2 For example, to checkout, SMS messages including transaction request and
3 authorization code may be exchanged between merchant and consumer phones via a
4 cellular network. In further implementations, the feature phone may be equipped with
5 a card reader plug-in 307a (e.g., Square card reader accessory, etc.), and/or a NFC 307b
6 component, etc., so that the consumer's magnetic card 302.1 and RFID card 302.2 may
7 be read by the smartphone terminal 310.e, or the consumer's smartphone 302.3. Within
8 implementations, the smartphone checkout may be operated cellular communication
9 networks, TCP/IP, Bluetooth, GPS, audio phone communication, video/image
10 capturing, accelerometer 303d, and/or the like. For another example, the consumer
11 302 may snap a photo of the barcode of the purchased item (and/or a generated QR
12 code generated at the terminal) with the smartphone 302.3, which may transmit an
13 image of the barcode or the QR code to a processing network. In another
14 implementation, instead of operating smartphones 310.3 as barcode readers, the POS
15 terminal may be equipped with barcode readers, such as, but not limited to Unitech All
16 Terminals Ms146i-3ps2g Ms146 Barcode Slot Reader Ps2 Conn Infrared Ip54 Std, Intel
17 IMAGETEAM 3800LR Bar Code Reader, and/or the like.

18 **[00124]** In one implementation, a merchant may be equipped with a mobile phone
19 as an acceptance device, but may not require two phones to tap or connect via blue
20 tooth, wifi, or NFC in order to connect to each other to allow flexible usage of mobile
21 checkout with different types of mobile phones and smartphones. In one
22 implementation, the consumer device and merchant mobile terminal may be connected
23 via WAP or data connections that would facilitate a connection to the MCB-Platform
24 server via USSD or IP.

1 **[00125]** Within implementations, a variety of consumer merchant checkout
2 formats discussed at 300a may further facilitate data collection such as, but not limited
3 to web claws, merchant statistics, dollar ranges of merchant products, card history,
4 and/or the like. Such data collection is further discussed in FIGURE 6A.

5 **[00126]** FIGURE 3B provides a data flow diagram illustrating transaction at
6 merchant POS within implementations of the MCB-Platform. Within implementations,
7 the consumer 302 may initiate a transaction by submitting his wallet information 304 to
8 a merchant terminal. For example, in one implementation, the consumer wallet 303
9 may comprise any of the consumer payment devices 302.1-302.4 discussed in FIGURE
10 3A, wherein the merchant 110 may comprise a barcode NFC plate beacon 310a at the
11 electronic cash register (ECR) 310b. For example, the NFC enabled POS 310a may
12 receive information from a payment card with a RFID and obtain wallet information
13 304. In one implementation, the wallet information message 304 may take a form
14 similar to 207 in FIGURE 2A.

15 **[00127]** In one implementation, the NFC enabled POS 310a may send a pre-check
16 request 306 to the MCB-Platform server 302, e.g., to check whether the merchant has
17 participated with MCB-Platform services. For example, POS 310a may generate a
18 HTTPS POST message to make a merchant eligibility pre-check including the merchant
19 information in the form of data formatted according to the XML. Below is an example
20 HTTP(S) POST message including an XML-formatted pre-check request message 306
21 for the MCB-Platform server:

```
22 POST /PreCheckRequest.php HTTP/1.1
23 Host: 206.205.82.130
24 Content-Type: Application/XML
25 Content-Length: 418
26 <?XML version = "1.0" encoding = "UTF-8"?>
```

```
1    <PreCheckRequest>
2    <Time> 12:30:30 </Time>
3    <Date> 10-10-2015 </Date>
4    <MerchantID> STBUX </MerchantID>
5    <MerchantName> StarBucks </MerchantName>
6    <TerminalID> ST0001 </TerminalID>
7    <Location> 1332 Dream Street </Location>
8    <Zipcode> 00000 </Zipcode>
9    <TransactionType> MCB-Platform transaction </TransactionType>
10   ...
11   </PreCheckRequest>
12
13
14
```

15 **[00128]** In the above example, the pre-check message 306 include information as
16 to check whether the merchant has registered to participate in a MCB-Platform
17 transaction, e.g., a transaction using MCB-Platform issued wallet, a merchant offer
18 bridging transaction, and/or the like.

19 **[00129]** In one implementation, the MCB-Platform may query on a merchant
20 database 319 based on a merchant ID to determine whether the requesting merchant
21 has enrolled. If yes, the MCB-Platform may retrieve previously stored merchant
22 information 308 (e.g., see also 260 in FIGURE 2A) from the merchant database 319.

23 **[00130]** Once the pre-check eligibility has been established, a cashier 305 may
24 enter a sale request 309 at the ECR 310b, including a payment amount 311. For
25 example, the payment amount may be told by the cashier to the consumer 302. For
26 another example, the payment amount may be sent to the consumer's mobile wallet via
27 SMS, as further discussed in FIGURE 4B. For another example, a QR code may be
28 generated at the ECR 310b, wherein the consumer may operate his mobile wallet to snap
29 a picture of the QR code to obtain the payment amount 311.

30 **[00131]** In one implementation, the consumer may confirm the payment amount,
31 and the POS 310a may then generate a payment request 312 to the MCB-Platform server
32 320. For example, POS 310a may generate a HTTPS POST message to make a

1 transaction payment request including the purchasing information and consumer
 2 account information in the form of data formatted according to the XML. Below is an
 3 example HTTP(S) POST message including an XML-formatted payment request
 4 message 312 for the MCB-Platform server:

```

5  POST /PaymentRequest.php HTTP/1.1
6  Host: 206.205.82.130
7  Content-Type: Application/XML
8  Content-Length: 418
9  <?XML version = "1.0" encoding = "UTF-8"?>
10 <Paymentrequest>
11   <Time> 12:30:30 </Time>
12   <Date> 10-10-2015 </Date>
13   <MerchantID> STBUX </MerchantID>
14   <MerchantName> StarBucks </MerchantName>
15   <MerchantAddress> 1400 Dream St </MerchantAddress>
16   <Zipcode> 00000 </Zipcode>
17   <TerminalID> ST0001 </TerminalID>
18   <Location> 1332 Dream Street </Location>
19   <Zipcode> 00000 </Zipcode>
20   <PurchaseItem>
21     <ItemID> 000909090 </ItemID>
22     <ItemName> Coffee-001 </ItemName>
23     <Price> $4.00 </Price>
24     <Tax> $0.20 </Tax>
25     <Total> $4.20 </Total>
26     ...
27   </PurchaseItem>
28   <Payment>
29     <CardNumber> 0000 0000 0000 0000 </CardNumber>
30     <CVV> 000 </CVV>
31     <ExpirationDate> 12-2020 </ExpirationDate>
32     <CardType> Visa </CardType>
33     <BIN> 000000 </BIN>
34     <BankRouting> 0000000000 </BankRouting>
35     ...
36   </Payment>
37   ...
38 </PaymentRequest>
39

```

40 **[00132]** In one implementation, the MCB-Platform server 320 may verify the
 41 received payment request information 312 with the retrieved merchant information
 42 308, e.g., to avoid fraudulent transaction requests, etc. If there is any inconsistency,
 43 e.g., the merchant address in 312 differs from that in 308 (see also FIGURE 1C), the
 44 MCB-Platform may generate a confidence inquiry 324 to a scoring component 315 to
 45 determine how confident it is that the received transaction request 312 is accurate. For

1 example, the MCB-Platform server 320 may generate a HTTPS POST message to make a
 2 confidence inquiry 324 including the received transaction request information and the
 3 inconsistent information portion in the form of data formatted according to the XML.
 4 Below is an example HTTP(S) POST message including an XML-formatted confidence
 5 inquiry message 324 for the MCB-Platform server:

```

6  POST /confidenceinquiry.php HTTP/1.1
7  Host: www.MCB-platform.com
8  Content-Type: Application/XML
9  Content-Length: 418
10 <?XML version = "1.0" encoding = "UTF-8"?>
11 <ConfidenceInquiry>
12   <Time> 12:31:02 </Time>
13   <Date> 10-10-2015 </Date>
14   <Merchant>
15     <MerchantID> STBUX </MerchantID>
16     <MerchantName> StarBucks </MerchantName>
17     <MerchantAddress> 1400 Dream St </MerchantAddress>
18     <Zipcode> 00000 </Zipcode>
19     <TerminalID> ST0001 </TerminalID>
20     <Location> 1332 Dream Street </Location>
21     <Zipcode> 00000 </Zipcode>
22     ...
23   </Merchant>
24   <ActivityType> Transaction Payment </ActivityType>
25   <InconsisntInfo>
26     <DataField> Merchant Address </DataField>
27     <Received> 1332 Dream Street </Received>
28     <Stored> 1400 Dream Street </Stored>
29     ...
30   </InconsistentInfo>
31   <Consumer>
32     <ID> JS 000 </Consumer>
33     <Name> John Smith </Name>
34     <WalletID> JSW001 </WalletID>
35     ...
36   </Consumer>
37   ...
38 </ConfidenceInquiry>
39

```

40 **[00133]** In the above example, the confidence inquiry message 324 provide
 41 merchant information associated with the request activity, consumer information, and
 42 an activity type as "transaction payment." The MCB-Platform scoring component 315
 43 may determine how confident they are with the inconsistent new merchant information
 44 based on a variety of information, such as, but not limited to web claws 325 from

1 Internet web 325 (e.g., news articles, merchant websites, etc.), consumer inputs 322
2 (e.g., consumer social media activities showing interactions of the merchant, purchasing
3 history in the wallet, etc.), merchant updates, and/or the like. For example, one
4 indicator for the confidence determination would be whether similar inconsistent
5 information included in the payment request 312 (e.g., the address change as reflected
6 in the above example) has been shown in additional information inputs such as 322-
7 323.

8 **[00134]** Further implementations of the scoring component are discussed in
9 FIGURES 6A-6E and 19A-19F. For example, the MCB-platform may provide inputs to
10 the scoring component such as but not limited to account history, account purchasing,
11 TCP/IP address, yak-tech paring, Internet claw, social media activities, accelerometer
12 information (e.g., provided in the protocols in FIG. 3A in the stack). In one
13 implementation, the scoring component may be used to assign the confidence value in
14 updating merchant information in MCB-Platform. For example, the scoring component
15 (or the centralized personal information platform in FIGURES 18-37) may perform
16 analytics of the obtained data, and generate and constantly update a confidence level of
17 a piece of merchant information updates based on the most updated data inputs.
18 Further implementations of the scoring component 315 are provided in FIUGRES 6A-
19 6E.

20 **[00135]** In one implementation, the MCB-Platform scoring component 315 may
21 generate a confidence decision 326 to the MCB-Platform server 320 indicating whether
22 the transaction may be processed in light of the inconsistent merchant information. For
23 example, the MCB-Platform scoring component 315 may generate a HTTPS POST

1 message to inform the confidence decision 326 in the form of data formatted according
 2 to the XML. Below is an example HTTP(S) POST message including an XML-formatted
 3 confidence decision message 326 for the MCB-Platform server:

```

4  POST /confidencedecision.php HTTP/1.1
5  Host: www.scoring.com
6  Content-Type: Application/XML
7  Content-Length: 418
8  <?XML version = "1.0" encoding = "UTF-8"?>
9  <ConfidenceDecision>
10     <Time> 12:31:33 </Time>
11     <Date> 10-10-2015 </Date>
12     <Merchant>
13         <MerchantID> STBUX </MerchantID>
14         <MerchantName> StarBucks </MerchantName>
15         <MerchantAddress> 1400 Dream St </MerchantAddress>
16         <Zipcode> 00000 </Zipcode>
17         <TerminalID> ST0001 </TerminalID>
18         <Location> 1332 Dream Street </Location>
19         <Zipcode> 00000 </Zipcode>
20     ...
21 </Merchant>
22 <InconsisntInfo>
23     <DataField> Merchant Address </DataField>
24     <Received> 1332 Dream Street </Received>
25     <Stored> 1400 Dream Street </Stored>
26     ...
27 </InconsistentInfo>
28 <ActivityType> Transaction Payment </ActivityType>
29 <ActivityThreshold> 0.50 </ActivityThreshold>
30 <InfoScore> 0.55 </InfoScore>
31 <ActivityStatus> Allowed </ActivityStatus>
32 <RequestedAction> DB update </RequestedAction>
33 ...
34 </ConfidenceDecision>
35

```

36 **[00136]** In the above example, the confidence decision indicates the confidence
 37 score of the requested payment transaction in 312 has met the threshold requirement.
 38 Therefore, the MCB-Platform may approve the transaction and update the merchant
 39 database. In one implementation, a processing request 313 may be sent to the financial
 40 network 330 (e.g., the payment account's associated bank, etc.) for processing. For
 41 example, the MCB-Platform 320 may generate a HTTPS POST message to request
 42 financial processing in the form of data formatted according to the XML. Below is an
 43 example HTTP(S) POST message including an XML-formatted processing request

1 message 326 for the financial network 330:

```

2  POST /processingrequest.php HTTP/1.1
3  Host: www.MCB-Platform.com
4  Content-Type: Application/XML
5  Content-Length: 418
6  <?XML version = "1.0" encoding = "UTF-8"?>
7  <processingrequest>
8  <Time> 12:31:57 </Time>
9  <Date> 10-10-2015 </Date>
10 <Merchant>
11     <MerchantID> STBUX </MerchantID>
12     <MerchantName> StarBucks </MerchantName>
13     <MerchantAddress> 1400 Dream St </MerchantAddress>
14     <Zipcode> 00000 </Zipcode>
15     <TerminalID> ST0001 </TerminalID>
16     <Location> 1332 Dream Street </Location>
17     <Zipcode> 00000 </Zipcode>
18     ...
19 </Merchant>
20 <PurchaseItem>
21     <ItemID> 000909090 </ItemID>
22     <ItemName> Coffee-001 </ItemName>
23     <Price> $4.00 </Price>
24     <Tax> $0.20 </Tax>
25     <Total> $4.20 </Total>
26     ...
27 </PurchaseItem>
28 <Payment>
29     <CardNumber> 0000 0000 0000 0000 </CardNumber>
30     <CVV> 000 </CVV>
31     <ExpirationDate> 12-2020 </ExpirationDate>
32     <CardType> Visa </CardType>
33     <BIN> 000000 </BIN>
34     <BankRouting> 0000000000 </BankRouting>
35     <BankName> Bank of America </BankName>
36     ...
37 </Payment>
38     ...
39 </ProcessingRequest>
40

```

41 **[00137]** For another example, the processing request message 326 may take a form
42 similar to the Visa Single Message System (SMS) format, Visa Original Credit
43 Transaction (OCT) format, and/or the like.

44 **[00138]** In one implementation, upon the transaction has been processed at the
45 financial network, an approval message 316a may be sent to the merchant 310 through
46 an acquirer 340. For example, the acquirer may facilitate routing the approval message
47 316b to the merchant terminal 310b. In another implementation, the approval notice
48 315 may be sent to the consumer via his electronic wallet 303, e.g., email, instant

1 messages, SMS, and/or the like. For example, the MCB-Platform 320 may generate a
 2 HTTPS POST message to notify approval of the transaction in the form of data
 3 formatted according to the XML. Below is an example HTTP(S) POST message
 4 including an XML-formatted transaction approval message 315 for the consumer
 5 (and/or the merchant):

```

6  POST /approval.php HTTP/1.1
7  Host: www.MCB-Platform.com
8  Content-Type: Application/XML
9  Content-Length: 418
10 <?XML version = "1.0" encoding = "UTF-8"?>
11 <approval>
12   <Time> 12:32:02 </Time>
13   <Date> 10-10-2015 </Date>
14   <Consumer>
15     <WalletID> JSW001 </WalletID>
16     <Name> John Smith </Name>
17     <PhoneNumber> 000-000-0000 </PhoneNumber>
18     ...
19   </Consumer>
20   <Merchant>
21     <MerchantID> STBUX </MerchantID>
22     <MerchantName> StarBucks </MerchantName>
23     <MerchantAddress> 1400 Dream St </MerchantAddress>
24     <Zipcode> 00000 </Zipcode>
25     <TerminalID> ST0001 </TerminalID>
26     <Location> 1332 Dream Street </Location>
27     <Zipcode> 00000 </Zipcode>
28     ...
29   </Merchant>
30   <PurchaseItem>
31     <ItemID> 000909090 </ItemID>
32     <ItemName> Coffee-001 </ItemName>
33     <Price> $4.00 </Price>
34     <Tax> $0.20 </Tax>
35     <Total> $4.20 </Total>
36     ...
37   </PurchaseItem>
38   <Payment>
39     <CardNumber> ***** 0000 </CardNumber>
40     ...
41   </Payment>
42   <TransactionStatus> Complete </TransactionStatus>
43   ...
44 </Approval>
45

```

46 **[00139]** FIGURE 3C provides a logic flow diagram illustrating transaction at
 47 merchant POS within implementations of the MCB-Platform. Within embodiments, the
 48 consumer 302 may submit consumer wallet information 335 (e.g., see 304 in FIGURE

1 3B) at a merchant POS terminal 310, which may establish the wallet is at the merchant
2 and submit a pre-check request 338 (e.g., see 306 in FIGURE 3B). For example, in one
3 implementation, the consumer may confirm his presence at the merchant store with a
4 cashier. In another implementation, the consumer's electronic wallet (e.g., on a
5 smartphone) may transmit his GPS coordinates to the MCB-Platform to confirm the
6 consumer's location.

7 **[00140]** In one implementation, the merchant may send the pre-check request to
8 confirm the merchant is an enrolled acceptor via NFC handshake, bar code read or
9 merchant beacon, and/or the like.

10 **[00141]** Upon receiving the pre-check request, the MCB-Platform may form a
11 query in a merchant database 340. If the merchant 310 is not an eligible participating
12 member 343, the merchant 310 may receive a denial 345 at the terminal, e.g., a notice at
13 the terminal that "wallet not acceptable," etc. Alternatively, if the merchant is eligible,
14 the merchant may initiate the wallet payment at its ECR 346. For example, a cashier
15 may select a wallet payment button (e.g., a "Visa V" button, a "V.me" button, etc, on a
16 touch screen panel) on ECR / terminal so it is prepared to record the sale.

17 **[00142]** In one implementation, the cashier may inform the consumer of the
18 amount. In another implementation, the MCB-Platform may provide offers that may be
19 redeemable for the purchase to the consumer via the consumer's wallet, or via the
20 merchant terminal, and/or the like. For example, when the MCB-Platform receives
21 purchase information of an Toshiba product at BestBuy, the MCB-Platform may query
22 its offer database and provide a "5% off Any Toshiba Laptop" offer to the consumer or
23 the POS terminal at BestBuy. The purchase price may then be calculated to reflect

1 redemption of the offer. Further implementations of offer matching are discussed in
2 FIGURES 2C-2D.

3 **[00143]** In one implementation, the consumer may enter amount in wallet, select
4 payment type and authenticate themselves 348 by entering wallet password (e.g., by
5 login onto his mobile wallet, by entering a password at the POS terminal, etc.).

6 **[00144]** In one implementation, the MCB-Platform may verify merchant accepts
7 payment type, apply merchant offers, discounts, loyalty calculations, confirm ability to
8 pay (issuer approval) and initiate merchant payment to send an approval code (e.g., see
9 315 in FIGURE 3B) to wallet 350. In another implementation, the MCB-Platform may
10 perform consumer/merchant confidence level verification 352 to determine whether the
11 transaction may be authorized, as discussed at 324 in FIGURE 3B.

12 **[00145]** In one implementation, the MCB-Platform may send the transaction to a
13 financial network 330 for processing 353, e.g., to deduct funds from the consumer's
14 account and credit the funds to the merchant's account. Upon authorizing the
15 transaction, the MCB-Platform may send an approval (e.g., see 316a/b in FIGURE 3B)
16 to the merchant via acquirer 355, and then the merchant may "close ticket" (e.g., the
17 transaction payment session, etc.) with a final paid amount, with identified discounts on
18 receipt / in system. Such approval may be used to pay the merchant when the wallet
19 account manager created an authorization response for the transaction, e.g., a receipt
20 365 to the consumer's wallet in the form of SMS, print by merchant, etc.

21 **[00146]** In alternative embodiments, e.g., at 338, the merchant may establish that
22 the wallet is at the merchant via NFC handshake (e.g., Paveway, etc.), and the consumer
23 wallet may receive a requested purchase amount from ECR. In such scenarios, the

1 merchant ECR/POS may associate the consumer's wallet ID with an "open ticket" (e.g.,
2 the wallet information has been reserved for subsequent payment).

3 **[00147]** FIGURE 4A provides a data flow diagram illustrating a mobile to mobile
4 POS checkout within implementations of the MCB-Platform. As show in FIGURE 3A,
5 the merchant POS terminal may comprise a smartphone terminal 310.3, which may be
6 enabled with SMS for communication with consumer mobile wallets 320.3. For
7 example, the MCB-Platform may connect a consumer wallet phone to the merchant
8 checkout terminal, e.g., with a merchant operating a SQUARE® accessory and/or a
9 mobile acceptance app, and/or the like. In one implementation, the merchant with
10 mobile checkout may type where payment amount is final, which is equivalent of
11 "pairing" two devices, e.g., the merchant smartphone and consumer smartphone. In
12 one implementation, the MCB-Platform may determine the final payment amount if
13 redemptions and rewards are factored in.

14 **[00148]** In one implementation, the consumer 402 may submit a mobile phone
15 number 404 to the merchant terminal 410 which operates a mobile phone 410a. For
16 example, the consumer may tell the mobile number to a cashier. For another example,
17 the consumer may instantiate his wallet application 403 which may capture the
18 consumer's mobile number and send it to the merchant mobile phone 410a, e.g., via
19 SMS.

20 **[00149]** The merchant may then generate a payment request summary message to
21 the consumer via SMS including a payment amount, a reference number and/or the like
22 411. For example, in one implementation, the SMS 411 may take a form similar to the
23 following:

1 From: 140
2 Time: 12:30:23
3 Date: 09-09-2014
4 Content: \$4.25 \$\$REF: d09dsffsfsade
5

6 **[00150]** The consumer may then inserts the amount and the reference number in
7 the SMS to his mobile wallet 412. For example, the mobile wallet may comprise an
8 entry for extracting payment amount and reference number obtained via SMS. The
9 MCB-Platform may then generate a payment request 407 to the merchant database, and
10 perform confidence authorization to proceed the transaction via the scoring component
11 415, in a similar manner as discussed at 324 and 326 in FIGURE 3B.

12 **[00151]** In one implementation, the MCB-Platform may generate a processing
13 request 413 to the financial network 430, which may take a form similar to 313 in
14 FIGURE 3B. Accordingly, a transaction approval 416a/b may be sent to the merchant
15 via the acquirer 440.

16 **[00152]** In another implementation, the approval 414 may be sent to the consumer
17 via SMS. For example, in one implementation, the SMS 414 may take a form similar to
18 the following:

19 From: Wallet
20 Time: 12:30:45
21 Date: 09-09-2014
22 Content: Your purchase has been approved! APPROVAL CODE #FDF&FSFDSF094
23

24 **[00153]** FIGURE 4B provides a logic flow diagram illustrating a mobile to mobile
25 POS checkout within implementations of the MCB-Platform. In one implementation,
26 the consumer may submit consumer wallet information 435, e.g., by providing a mobile
27 phone number. The merchant may instantiate the merchant mobile app, which has an
28 open ticket, e.g., a V.me mobile app, etc., and enter the consumers mobile number 438.

1 **[00154]** In one implementation, the merchant may send a SMS (e.g., 411 in
2 FIGURE 4A) to the consumer's mobile wallet via the merchant mobile application on a
3 mobile phone 440. SMS contains merchant app generated reference number and a
4 payment amount. In such manner, there is no NFC needed or proxy "card" to swipe.

5 **[00155]** Within implementations, consumer may receive the SMS with amount
6 from merchant number (e.g., a proxy for merchant ID), inserts securely to his mobile
7 wallet, and submit consumer authentication credentials 445 to the MCB-Platform.

8 **[00156]** Within implementations, the MCB-Platform may verify merchant accepts
9 payment type selected 446, and then apply merchant offers, discounts, loyalty
10 calculations 452. The MCB-Platform may then confirm ability to pay (issuer approval
11 on the available funds in the account, e.g., credit limit, funds in a debit account, etc.) 455
12 and process the financial transaction 453 with the financial network.

13 **[00157]** In one implementation, the consumer may receive an approval message at
14 his wallet, e.g., via SMS 455 (see 414 in FIGURE 4A).

15 **[00158]** Within implementations, the MCB-Platform may send approval to
16 merchant via acquirer 457, which may require insertion of acquirer merchant ID based
17 on the mobile number proxy. Within implementations, the approval message may be
18 used to pay the merchant when the wallet account manager created the "auth response,"
19 e.g., receipt to consumer wallet, SMS, print receipt at the terminal, etc.

20 **[00159]** Within implementations, merchant mobile checkout app may close ticket
21 (e.g., app authenticates incoming confirmation with reference number sent in SMS and
22 merchant ID), with final amount, identifies discounts 460 on virtual receipt / in system.
23 The consumer may receive a purchase receipt via SMS 465.

1 **[00160]** In further implementations, interacting with the wallet in real time at the
2 POS may provide real time rewards, redemptions and offers to the consumer 402. The
3 offer matching to the consumer may be performed in a similar manner as discussed in
4 FIGURES 2C-2E.

5 **[00161]** FIGURE 5A provides a data flow diagram illustrating wallet enrollment
6 tokenization within implementations of the MCB-Platform. Within implementations, a
7 consumer may desire to add an account to his electronic wallet, e.g., by linking his bank
8 account to his wallet, by adding a payment entry to his mobile wallet, etc. In one
9 implementation, the consumer may submit enrollment information 512 to the MCB-
10 Platform server (e.g., a wallet management unit, etc.). For example, the consumer 502
11 may call a MCB-Platform representative to add a card to his wallet. In another
12 implementation, the consumer 502 may add an account via a web based UI (e.g., see
13 FIGURES 8A-8B). For example, the consumer wallet 503 application may generate a
14 HTTPS POST message to enroll an account in the form of data formatted according to
15 the XML. Below is an example HTTP(S) POST message including an XML-formatted
16 card enrollment message 512 for the MCB-Platform:

```
17 POST /CardEnrollment.php HTTP/1.1
18 Host: 206.205.82.130
19 Content-Type: Application/XML
20 Content-Length: 418
21 <?XML version = "1.0" encoding = "UTF-8"?>
22 <CardEnrollment>
23   <Time> 12:32:02 </Time>
24   <Date> 10-10-2015 </Date>
25   <Consumer>
26     <WalletID> JSW001 </WalletID>
27     <Name> John Smith </Name>
28     <PhoneNumber> 000-000-0000 </PhoneNumber>
29     ...
30   </Consumer>
31   <NewCard>
32     <CardNumber> ***** 0000 </CardNumber>
33     <CardHolderName> John Smith </CardholderName>
34     <CardType> Visa </CardType>
35     <ExpirationDate> 11/2018 </ExpirationDate>
```

```

1         <CVV> 000 </CVV>
2         <BankID> BOA </BankID>
3         <BankName> Bank of America </BankName>
4         <BankRouting> 0000000 </Bankrouting>
5         <BIN> 00000001 </BIN>
6         ...
7     </NewCard>
8     ...
9 </CardEnrollment>
10

```

11 **[00162]** In one implementation, the MCB-Platform server 520 may require the
 12 card to be compliant with regulatory acts, e.g., the Durbin Amendment, etc. In one
 13 implementation, the financial network 530 (e.g., the card's issuing bank) may provide a
 14 token 514a, which may be converted a tokenized PAN number 514b for the consumer
 15 wallet 503. In one implementation, the card information is replaced with the tokenized
 16 PAN 514b that preserves the BIN so as not to impede merchant routing choice at the
 17 POS terminals. For example, the tokenized PAN 514b may comprise an integer value.

18 **[00163]** In one implementation, the consumer 502 may initiate a transaction by
 19 submitting wallet information 504 (e.g., see 304 in FIGURE 3B, 404 in FIGURE 4A) to
 20 a NFC enabled POS terminal 510b, which may be forwarded to the merchant 510. In
 21 one implementation, the merchant may determine a issuer network 516 and route the
 22 transaction request to the financial network 530 based o the issuer network indication
 23 516.

24 **[00164]** Alternatively, the merchant may determine a BIN number from the wallet
 25 info based on the token of a selected account and send the BIN number 508 to an
 26 acquirer which may determine the issuer network 516b to route the payment transaction
 27 message on behalf of the merchant 510. For example, the acquirer may generate a
 28 HTTPS POST message to send issuer network brand in the form of data formatted
 29 according to the XML. Below is an example HTTP(S) POST message including an XML-

1 formatted issuer network brand message 516b for the MCB-Platform:

```

2  POST /IssuerBrand.php HTTP/1.1
3  Host: 206.205.82.130
4  Content-Type: Application/XML
5  Content-Length: 418
6  <?XML version = "1.0" encoding = "UTF-8"?>
7  <IssuerBrand>
8  <Time> 12:32:32 </Time>
9  <Date> 10-10-2015 </Date>
10 <Consumer>
11     <WalletID> JSW001 </WalletID>
12     <Name> John Smith </Name>
13     <PhoneNumber> 000-000-0000 </PhoneNumber>
14     ...
15 </Consumer>
16 <Token> 000000000sad </Token>
17 <BIN> 00000001 </BIN>
18 <IssuerID> BOA </IssuerID>
19 <RoutingNo> 000001 </RoutingNo>
20
21     ...
22 </CardEnrollment>

```

23 **[00165]** The MCB-Platform may then route a processing request 518 (e.g., see 313
 24 in FIGURE 3B) to a determined issuer 550 based on the received issuer network
 25 message 516a/b. The issuer 550 may in turn send back an authorization message 519a
 26 (e.g., see the approval message 316a in FIGURE 3B), which may be provided to the
 27 merchant terminal, e.g., at 519b.

28 **[00166]** FIGURE 5B provides a logic flow diagram illustrating wallet enrollment
 29 tokenization within implementations of the MCB-Platform. During an initial
 30 enrollment 500a, e.g., when a Durbin Compliant card is enrolled initially in a wallet, the
 31 card information is replaced with a token that preserves the BIN so as not to impede
 32 merchant routing choice at the POS. Within implementations, the consumer 502 may
 33 submit enrollment information 535 (e.g., 512 in FIGURE 5A) to the MCB-Platform 520,
 34 which may provide a token to replace the card information 542. Such token may be
 35 issued by an issuer network which may associate the token with the card number 543.

36 **[00167]** Upon enrollment, the merchant routing 500b may be performed based on

1 the token number. Within implementations, upon consumer submitting wallet
2 information including the token number of an enrolled card 552, the merchant 510, who
3 may be a wallet POS acceptor, may retrieve a BIN number 553 from the token number
4 in the wallet. The merchant may then determine whether an issuer network is
5 determined 555. If yes, the merchant may insert the brand code for the network (e.g.,
6 an issuer network) they select (e.g., a new field in the auth) based on the BIN number
7 558. If the merchant POS can not designate the network brand 555, then the merchant
8 may send the BIN number to the acquirer and the acquirer may do so on behalf of the
9 merchant 560.

10 **[00168]** Within implementations, the transaction may come to MCB-Platform for
11 “token conversion” 563 (acquirers may know this from the fact the POS service is
12 identified as a wallet payment, e.g., “V,me,” etc.), which may convert the token to
13 retrieve card information, and routed to the appropriate issuer / network. The financial
14 network 530 may develop pricing 564 for the transactions that do not cause the
15 acquirers to assert that choosing a processing platform other than MCB-Platform may
16 result in a “penalty.”

17 **[00169]** Within implementations, an approval message may be sent to the
18 merchant via the acquirer 565. In further implementations, the MCB-Platform may
19 develop a fee structure that charges any other network 566, and not the merchant for
20 delivering secure wallet transactions from the POS, e.g., a licensing fee or a delivery /
21 transport fee, etc.

22 **[00170]** FIGURE 5C provides a data flow diagram illustrating acquirer routing 560
23 within implementations of the MCB-Platform; and FIGURE 5D provides a logic flow

1 diagram illustrating acquirer routing 560 within implementations of the MCB-Platform.
2 Within implementations, the consumer 502 may submit s check out request 571 (e.g.,
3 504 in FIGURE 5A) to a merchant 510, which may provide a sign on prompt 572 for the
4 merchant to sign in with MCB-Platform wallet checkout service if the merchant has
5 been verified to be a MCB-Platform wallet checkout participant.

6 **[00171]** In one implementation, the MCB-Platform may provide a user login
7 request 573 to the consumer, e.g., by displaying a login request to the consumer at the
8 consumer's mobile device (e.g., see FIGURE 40F), wherein the consumer may in turn
9 enter login credentials, e.g., a username and a password, etc. The consumer may further
10 provide card selection 574 along with credential submissions to the MCB-Platform 520.
11 For example, the consumer wallet may generate a HTTPS POST message to send login
12 credentials and card selection in the form of data formatted according to the XML.
13 Below is an example HTTP(S) POST message including an XML-formatted card
14 selection message 574 for the MCB-Platform:

```
15 POST /CardSelection.php HTTP/1.1
16 Host: 206.205.82.130
17 Content-Type: Application/XML
18 Content-Length: 418
19 <?XML version = "1.0" encoding = "UTF-8"?>
20 <CardSelection>
21   <Time> 12:32:32 </Time>
22   <Date> 10-10-2015 </Date>
23   <Consumer>
24     <WalletID> JSW001 </WalletID>
25     <Name> John Smith </Name>
26     <PhoneNumber> 000-000-0000 </PhoneNumber>
27     <Password> ***** </Password>
28     <PhoneID> JSAppleiOS0001 </PhoneID>
29     ...
30   </Consumer>
31   <SelectCard>
32     <CardAlias> MyVisaCard </CardAlias>
33     <Token> 000000000sad </Token>
34     <BIN> 00000001 </BIN>
35     ...
36   </SelectCard>
37   ...
38 </CardSelection>
```

1
2 **[00172]** Within implementations, the MCB-Platform may route an issuer
3 authorization request message 575 to a processing gateway 540, e.g., based on the BIN
4 number of the selected payment card 574. For example, the MCB-Platform may
5 generate a HTTPS POST message to request issuer authorization in the form of data
6 formatted according to the XML. Below is an example HTTP(S) POST message
7 including an XML-formatted issuer authorization request message 574 for the
8 processing gateway 540:

```
9  POST /IssAuthRequest.php HTTP/1.1
10  Host: 206.205.82.130
11  Content-Type: Application/XML
12  Content-Length: 418
13  <?XML version = "1.0" encoding = "UTF-8"?>
14  <IssAuthRequest>
15    <Time> 12:32:32 </Time>
16    <Date> 10-10-2015 </Date>
17    <Consumer>
18      <WalletID> JSW001 </WalletID>
19      <Name> John Smith </Name>
20      <PhoneNumber> 000-000-0000 </PhoneNumber>
21      <Password> ***** </Password>
22      <PhoneID> JSAppleiOS0001 </PhoneID>
23    ...
24  </Consumer>
25  <BIN> 00000001 </BIN>
26  <Issuer>
27    <IssuerID> BOA </IssuerID>
28    <IssuerName> Bank of America </IssuerName>
29    <RoutingNo> 00000001 </RoutingNo>
30    ...
31  </Issuer>
32  <Payment>
33    <Amount> $4.20 </Amount>
34    <CardNo> 0000 0000 0000 0000 </CardNo>
35    <CardHolderName> John Smith </CardholderName>
36    ...
37  </Payment>
38  <Merchant>
39    <MerchantID> STBUX </MerchantID>
40    <MerchantName> StarBucks </MerchantName>
41    <MerchantAddress> 1400 Dream St </MerchantAddress>
42    <Zipcode> 00000 </Zipcode>
43    <TerminalID> ST0001 </TerminalID>
44    <Location> 1332 Dream Street </Location>
45    <Zipcode> 00000 </Zipcode>
46    <MerchantAccount> 0000 0000 0000 1111 </MerchantAccount>
47    ...
48  </Merchant>
49  ...
50  ...
51  </IssAuthRequest>
```

1

2 **[00173]** The processing gateway 540 may transmit the issuer authorization
3 message 576 to the issuer 550, which may in turn generate an issuer authorization
4 response 577 (e.g., see also 519a in FIGURE 5A), and routed to the MCB-Platform.

5 **[00174]** Within implementations, the MCB-Platform may generate session results
6 580a to the merchant 510, wherein the session results may comprise the status of the
7 transaction, e.g., an approval message. Upon merchant receiving the session results
8 580b, the processing gateway may capture a transaction file 581a and send it to the
9 acquirer 581b. In one implementation, the processing gateway may provide acquirer
10 advice 579 to the acquirer.

11 **[00175]** FIGURE 6A provides a data flow diagram MCB-Platform merchant
12 database updates 600a within implementations of the MCB-Platform.

13 **[00176]** In one implementation, the MCB-Platform scoring component may obtain
14 various data inputs related to merchant information. For example, the scoring
15 component 605 may obtain web claws 623 from the Internet 620, merchant updates
16 628 and transaction record 627 from merchant site 615 and merchant stores 610, mobile
17 information 626 from mobile carriers 640, social media activities 624 from social media
18 630, and/or the like.

19 **[00177]** For example, in one implementation, the web claws 623 may comprise
20 new articles from a news page that mentions the merchant name, product information,
21 and/or the like. For another example, the merchant updates 628 and/or the transaction
22 record 627 may comprise merchant profile information, inventory information, pricing
23 information, and/or the like. For another example, social media updates 624 may

1 comprise a merchant's Facebook page updates, merchant Twitter updates, consumer
2 comments, "Likes" on a merchant product on Facebook, consumer tweets about the
3 merchant and/or the products, and/or the like. For another example, mobile
4 information 626 may comprise checkout request (e.g., see FIGURES 4A-4B), mobile
5 advertisement, mobile offers, and/or the like.

6 **[00178]** Within implementations, the MCB-Platform scoring component may
7 extract data fields from the various raw inputs, such as, but not limited to risk indexes,
8 number of items for a defined merchant record, average item price on site, item
9 diversity on site, defined merchant information verses estimated merchant information,
10 average price on site verse average print by merchant information, hosting country,
11 hosting service, average number of hosting services a year, site age, and/or the like.

12 **[00179]** In another implementation, the MCB-Platform scoring component may
13 create a look-up table to determine whether a received merchant data field change
14 indication has been verified. For example, such look up table may comprise data fields
15 such as, but not limited to valid email, valid address, valid phone, verified email, verified
16 address, verified phone, known spammer, deny list, allow list, and/or the like. For
17 example, FIGURE 6B provides an exemplary data structure of the extracted information
18 and how the information is interrelated within embodiments of the MCB-Platform.

19 **[00180]** Within implementations, the MCB-Platform may obtain information via a
20 consumer 602 initiating a transaction payment at a merchant terminal via various
21 protocols 604, such as, but not limited RFID 604a, TCP/IP 604b, mobile 604c, alias
22 telephone 604d, and/or the like. The scoring component 605 may receive information
23 such as a RFID 616, GPS 617, snapshot 618 (e.g., a picture of the storefront), audio

1 feedbacks 614 (e.g., audio recording of the store atmosphere), accelerometer data 612
 2 (e.g., movement data of a consumer smartphone), and/or the like. Further examples of
 3 merchant consumer checkout protocol stacks are discussed in FIGURE 3A.

4 **[00181]** In one implementation, upon verification of a confidence score (e.g., see
 5 FIGURE 6B), the MCB-Platform may update merchant information 629 at a merchant
 6 database. For example, the MCB-Platform may generate a HTTPS PUT message to
 7 request database update in the form of data formatted according to the XML. Below is
 8 an example HTTP(S) PUT message including an XML-formatted merchant record
 9 update message 629 for the merchant database 619:

```

10  PUT /DBupdate.php HTTP/1.1
11  Host: www.MCB-Platform.com
12  Content-Type: Application/XML
13  Content-Length: 418
14  <?XML version = "1.0" encoding = "UTF-8"?>
15  <IssAuthRequest>
16  <Time> 12:32:56 </Time>
17  <Date> 10-10-2015 </Date>
18  <NewMerchantRecord>
19      <MerchantID> STBUX </MerchantID>
20      <MerchantName> StarBucks </MerchantName>
21      <MerchantAddress> 1400 Dream St </MerchantAddress>
22      ...
23  </NewMerchantRecord>
24  <Confidence> Good </Confidence>
25  <ActivityType> Transaction </ActivityType>
26  <Status> Allow </Status>
27  ...
28  </DBupdate>
29

```

30 **[00182]** FIGURES 6C-6D provide logic flow diagrams illustrating merchant site
 31 monitoring within implementations of the MCB-Platform. For example, the MCB-
 32 Platform scoring component may obtain merchant URL either at merchant enrollment
 33 or from a wallet request to purchase from the merchant URL. To "spider" the merchant
 34 website (e.g., monitoring information updates from URL, etc.), in one implementation,

1 the MCB-Platform may collect all links on the site URL and then loop through the list of
2 URLs to look for information updates until the list is exhausted.

3 **[00183]** As shown in FIGURE 6C, the MCB-Platform may obtain a new URL from
4 merchant enrollment 6010 and add the merchant URL to a hash table. While there is
5 any URL exists in hash table 6012, the MCB-Platform may get a new URL from the hash
6 table 6015, and scrape contents from the URL 6018 and extract linked URLs from
7 contents 6020. In one implementation, the MCB-Platform may add the extracted URLs
8 to the hash table 6025 if not in list 6023. The MCB-Platform may then pop the
9 examined URL from the hash table and add it to a list of seen URLs 6030.

10 **[00184]** Alternatively, as shown in FIGURE 6D, the MCB-Platform may adopt a
11 stealth procedure, which may launch several threads from multiple domains with a
12 shared memory that simulated a scenario of many users navigating the merchant site
13 URL.

14 **[00185]** Within implementations, the MCB-Platform may add a merchant URL
15 from merchant enrollment to a hash table 6010, and then launch several threads 6040
16 which would get new URLs from hash table 6015, scrape content of the URL 6018,
17 extract URLs from contents 6020, pop URLs from hash table 6030, add reviewed URL
18 to a list of seen URLs 6025. The MCB-Platform may then sleep for random amount of
19 time 6035, and pick one URL 6040 from extracted URLs to resume at 6015. If no URL
20 exists 6040, the MCB-Platform may continue monitoring.

21 **[00186]** In one implementation, the MCB-Platform may extract table fields
22 information from "spider" URLs, such as, but not limited to url, date time, metadata,

1 content, images, javascript, css, referring sites, country of origin , hosting service, tcp/ip
2 packets, and/or the like.

3 **[00187]** FIGURES 6E-6F provide logic flow diagrams illustrating merchant
4 database update within embodiments of the MCB-Platform. Within implementations,
5 the MCB-Platform may receive an activity request with the merchant information 641,
6 wherein the activity may comprise a transaction request, an offer issuance request, a
7 merchant record update request, and/or the like.

8 **[00188]** For example, the consumer may submit in-store merchant information
9 (e.g., GPS information, snapshot pictures, audio recording, etc.) 635, e.g., the consumer
10 may submit such information to "check-in" via social media in order to obtain related
11 offers, submit a purchase transaction request, and/or the like.

12 **[00189]** In another implementation, the MCB-Platform may obtain transaction
13 information submission 636 from the merchant store 110, such as but not limited to a
14 server IP, physical store location, and/or the like. For example, a merchant store POS
15 terminal may submit a transaction request to the MCB-Platform, which includes
16 merchant store information. For another example, the merchant may submit a request
17 to the MCB-Platform to update merchant profile information.

18 **[00190]** In another implementation, the MCB-Platform may obtain miscellaneous
19 non-instant transaction information, such as web claws, social media updates, 637,
20 and/or the like.

21 **[00191]** Within implementations, the MCB-Platform may extract merchant key
22 terms from the merchant information 642 embedded in the activity request, e.g., a
23 merchant ID. The MCB-Platform may then query the merchant database based on the

1 merchant ID 643, and retrieve the previously stored merchant record. The MCB-
2 Platform may compare the stored merchant information with the received information
3 645 from the received activity request 645, to determine whether the received
4 information is new or inconsistent with the previously stored information 648. For
5 example, the MCB-Platform may receive a purchase transaction request which indicates
6 a different physical merchant store address from the previously stored merchant
7 address (e.g., see FIGURE 1C.(b)). For another example, the purchase transaction
8 request may comprise a product that is not included in the inventory information from
9 the previously stored merchant information.

10 **[00192]** When there is no new or inconsistent merchant information at 648, the
11 MCB-Platform may periodic monitor the process, and proceed with 641.

12 **[00193]** When there is such new or inconsistent merchant information 648,
13 continuing on with FIGURE 6F, the MCB-Platform may obtain a confidence score at a
14 scoring component 651 for the received merchant information, wherein such confidence
15 score may be calculated at a data aggregating platform including specific XML modules
16 to the MCB-Platform, as described in FIGURES 19A-19F. For example, all the inputs
17 635-637 in FIGURE 6E may be fed to the centralized personal information platform
18 1804 in FIGURE 18.

19 **[00194]** In one embodiment, the MCB-Platform scoring component may use the
20 adopt a structure to provide confidence levels for the types of inputs discussed. An
21 exemplary XML-formatted scoring structure may take a form similar to the following:

```
22 <init>
23   summary_Level="0"
24   environment_type="RT"
25   tumblr_location="../tumblars/merchantRisk.tumblr"
26 <output>
```

```
1      keyname=modelRes
2      file="stdout"
3  </output>
4  <input>
5      keyname=inputmerchant
6      file=../data/merchantRisk.test.csv
7      format=csv
8      meta_data=../metaData/merchantRisk.meta"
9  </input>
10 <constant>
11     indexname=_constant_10000000
12     value=10000000
13     type=float
14 </constant>
15 </init>
16 <macro="buildfeatrato">
17     tumblar-default="100000"
18     function-l="dividebyconst"
19     fnc-contant="10000000"
20     type="float"
21 </macro>
22 <vault>
23 <door=0>
24 <lock inkey="inputmerchant" inkeyname="url" outkey="normalised"
25 outkeyname="numItems" macro="buildFeatRato" tumblar="rskIdx_numItems"
26 />
27 <lock inkey="inputmerchant" inkeyname="url" outkey="normalised"
28 outkeyname="avgItemPrice" macro="buildFeatRato" tumblar="rskIdx_avgItemPrice"
29 />
30 <lock inkey="inputmerchant" inkeyname="url" outkey="normalised"
31 outkeyname="itemDiversity" macro="buildFeatRato" tumblar="rskIdx_itemDiversity"
32 />
33 <lock inkey="inputmerchant" inkeyname="url" outkey="normalised"
34 outkeyname="MCCdifference" macro="buildFeatRato" tumblar="rskIdx_MCCdifference"
35 />
36 <lock inkey="inputmerchant" inkeyname="url" outkey="normalised"
37 outkeyname="avgPrcOnsite" macro="buildFeatRato" tumblar="rskIdx_avgPrcOnsite"/>
38 <lock inkey="inputmerchant" inkeyname="url" outkey="normalised"
39 outkeyname="hostingCountry" macro="buildFeatRato"
40 tumblar="rskIdx_hostingCountry" />
41 <lock inkey="inputmerchant" inkeyname="url" outkey="normalised"
42 outkeyname="hostingService" macro="buildFeatRato"
43 tumblar="rskIdx_hostingService" />
44 <lock inkey="inputmerchant" inkeyname="url" outkey="normalised"
45 outkeyname="avgNumHostingServices" macro="buildFeatRato"
46 tumblar="rskIdx_avgNumHostingServices" />
47 <lock inkey="inputmerchant" inkeyname="url" outkey="normalised"
48 outkeyname="SiteAge" macro="buildFeatRato" tumblar="rskIdx_SiteAge" />
49 <lock inkey="inputmerchant" inkeyname="url" outkey="normalised"
50 outkeyname="LinksInByOrgin" macro="buildFeatRato"
51 tumblar="rskIdx_LinksInByOrgin" />
52 </door>
53
54 <door=1>
55 <lock inkey="validEmail" inkeyname="email"
56 outkey="mdlflags" outkeyname="valid_email_list"
57 function="instant"
58 tumblar="valid_email_list"
59 tumblar-default="0"
60 type="int"
61 />
62 <lock
63 inkey="validAddress" inkeyname="addressid"
```

```
1      outkey="mdlflags"      outkeyname="valid_address_list"
2      function="instant"
3      tumblar="valid_address_list"
4      tumblar-default="0"
5      type="int"
6  />
7  <lock
8      inkey="verifiedPhone"  inkeyname="phone"
9      outkey="mdlflags"      outkeyname="verified_phone_list"
10     function="instant"
11     tumblar="verified_phone_list"
12     tumblar-default="0"
13     type="int"
14 />
15 <lock
16     inkey="verifiedEmail"  inkeyname="email"
17     outkey="mdlflags"      outkeyname="verified_email_list"
18     function="instant"
19     tumblar="verified_email_list"
20     tumblar-default="0"
21     type="int"
22 />
23 <lock
24     inkey="verifiedAddress" inkeyname="addressid"
25     outkey="mdlflags"      outkeyname="verified_address_list"
26     function="instant"
27     tumblar="verified_address_list"
28     tumblar-default="0"
29     type="int"
30 />
31 <lock
32     inkey="verifiedPhone"  inkeyname="phone"
33     outkey="mdlflags"      outkeyname="verified_phone_list"
34     function="instant"
35     tumblar="verified_phone_list"
36     tumblar-default="0"
37     type="int"
38 />
39
40 <lock groupkeyname="mdlflags" groupkeyjoin=':' outkey="normalised"
41 outkeyname="phoneaddressemail_risk" macro="buildFeatRato"
42 tumblar="phoneaddressemail_risk" />
43
44 </door>
45
46 <door=2>
47 <lock
48     inkey="inputmerchant" inkeyname="url"
49     outkey="negflags"      outkeyname="url_knownspammer"
50     function="instant"
51     tumblar="card_negative_list"
52     tumblar-default="0"
53     type="int"
54 />
55 <lock
56     inkey="inputmerchant" inkeyname="url"
57     outkey="negflags"      outkeyname="url_denylist"
58     function="instant"
59     tumblar="card_negative_list"
60     tumblar-default="0"
61     type="int"
62 />
63 </door>
```

```

1
2 <door=3>
3 <lock
4   inkey="inputmerchant" inkeyname="url"
5   outkey="posflags"      outkeyname="url_allow"
6   function="instant"
7   tumblr="url_allow_list"
8   tumblr-default="1"
9   type="int"
10 />
11 </door>
12
13 <door=4>
14 <lock name="sumprob"
15   groupkeyname="normalised"
16   outkey="score" outkeyname="sumprob"
17   function="sumprob"
18   fnc-weights=".7|.8|.6|.3|.9|.1|.6|.6|.7|.2|.9"
19   type="float"
20 />
21 <lock
22   groupkeyname="score"
23   groupkey2name="negflags"
24   outkey="finalMdl" outkeyname="finalMdl"
25   function="takemax"
26   type="float"
27 />
28 </door>
29
30 <door=5>
31 <lock inkey="inputmerchant" inkeyname="url"      outkey="modelres"
32   outkeyname="url"      />
33 <lock
34   groupkeyname="finalMdl"
35   groupkey2name="posflags"
36   outkey="Score" outkeyindex="2"
37   function="takemin"
38   type="float"
39 />
40 </door>
41 </vault>
42

```

43 **[00195]** Further examples of scoring structures may be illustrated in the
 44 centralized personal information platform as discussed in FIGURES 18-37. In a further
 45 implementation, the confidence level determination may be performed by an integration
 46 of the XML-formatted scoring structure discussed above and the centralized personal
 47 information platform.

48 **[00196]** Continuing on with 653 in FIGURE 6F, in some embodiments, the MCB-
 49 Platform may retrieve a threshold for the activity type. For example, various activity

1 types may have greater or lower weights for confidence level evaluation. Based on the
2 nature of the requested activity (e.g., a transaction payment request, a merchant
3 information update request, a consumer bridging offer issuance request, etc.), the MCB-
4 Platform may require lower or higher values to accept the transaction and or update the
5 merchant database, allow the bridging event (e.g, prompt an offer). For example, a
6 lower confidence value that the merchant and consumer are related of 0.2 (normalized
7 confidence value) may be acceptable for issuing a merchant offer to a consumer).
8 However, a higher confidence level may be required, e.g., 0.5, to allow a transaction for
9 users that have a good account history. Higher confidence levels, e.g., 0.9, may be
10 required for updating merchant information. In another implementation, the MCB-
11 Platform scoring component may operate in parallel by constantly updating with regard
12 to the received transaction information.

13 **[00197]** The MCB-Platform may determine whether the obtained confidence score
14 is greater than the retrieved threshold 655. If yes, the MCB-Platform may update the
15 merchant record 657 with the new or updated information (e.g., see 123 in FIGURE 1C,
16 etc.), and advance the activity 658, e.g., to bridge an offer for the consumer, or to
17 advance transaction authorization processing, etc. Activity authorization notice may be
18 provided to the merchant 661a, and the consumer may obtain a confirmation notice
19 662a, e.g., a purchase receipt, a matched offer, etc.

20 **[00198]** In another implementation, if the confidence score fails to meet the
21 threshold value at 655, the MCB-Platform may timestamp the received information and
22 activity request 656 for the scoring component (e.g., such received information itself will
23 be integrated into the centralized personal information platform evaluation, as

1 discussed in FIGURES 18-37), and decline the activity request 659. A denial message
2 may be sent to the merchant 661b and the consumer 662b.

3 **[00199]** FIGURES 6G-6H provide alternative implementations of scoring
4 mechanism within embodiments of the MCB-Platform. Within implementations,
5 continuing on from 648 in FIGURE 6E, the MCB-Platform may decode the received
6 data table to extract data source information 665, and determine data source types 666.
7 For example, the data source may comprise a consumer wallet (e.g., snapshot of a
8 storefront, GPS coordinates, etc.), a merchant store or merchant site (e.g., merchant
9 updates, transaction request from POS terminals, etc.), Internet web, social media,
10 and/or the like.

11 **[00200]** In one implementation, if the data source comprises web/press news from
12 the Internet web, the MCB-Platform may determine whether it is a new web URL 667.
13 If not, the MCB-Platform may assign a confidence level associated with the known
14 web/press source 669. If yes, the MCB-Platform may assign a new confidence level, e.g.,
15 0.1, to a new web 668.

16 **[00201]** In another implementation, if the data source comprises consumer
17 submitted data, the MCB-Platform may further determine a data type 670. For
18 example, if it comprises purchase transaction information and the consumer's GPS
19 coordinates, the MCB-Platform may extract merchant information from the transaction
20 information 671a, and assign a higher confidence level, e.g., 0.4 671b, as the transaction
21 record may indicate better accuracy. If the received data includes a storefront snapshot
22 and GPS coordinates, the MCB-Platform may extract tiff information of the snapshot
23 photo 672a, and perform optical character recognition (OCR) to extract merchant

1 information from the photo 672b. Such data may be assigned a confidence score of 0.4,
2 672b. In another implementation, if the received data comprises audio recording and
3 the consumer's GPS coordinates, the MCB-Platform may perform audio analysis to
4 extract merchant information from the background recording 673a. Such data may be
5 assigned to a lower confidence level, e.g., 0.1 673b, as audio background sounds may be
6 less indicative or accurate.

7 **[00202]** In further implementations, if the data source comprises a merchant, e.g.,
8 merchant requested data updates, etc., the MCB-Platform may extract merchant
9 information 674a, and assign a higher confidence level of 0.4 674b.

10 **[00203]** In other implementations, if the data source comprises social media, the
11 MCB-Platform may extract merchant information 678a from the social media feeds,
12 news, activities, etc., and determine a confidence level based on the social media
13 message source 678b. For example, a consumer's post indicating an address change of a
14 merchant on Facebook may have a lower score of 0.1, but news posted on an official
15 merchant social media channel (e.g., a Starbucks page on Facebook, a Starbucks Twitter
16 account, etc.) may have a higher score of 0.3, and/or the like.

17 **[00204]** FIGURE 6H provides a logic flow illustrating aspects of progressive
18 scoring within implementations of the MCB-Platform. Continuing on with 641 in
19 FIGURE 6E, in an alternative implementation, the MCB-Platform may receive new
20 merchant update 681, and determine data fields of the inconsistent merchant
21 information 682. The MCB-Platform may then determine a confidence level of the new
22 merchant update 683 (e.g., as discussed in FIGURE 6G), and determine whether it
23 meets a threshold associated with the requested activity type 684. If yes, the MCB-

1 Platform may update the merchant record in the merchant database, and proceed with
2 658.

3 **[00205]** If not, the MCB-Platform may form a search on available merchant
4 information update based on the inconsistent merchant information 685. For example,
5 if the received merchant updates include a different store address from the record, the
6 MCB-Platform may query on the new store address. If such additional data exist 687,
7 the MCB-Platform may retrieve the data record 690 and its associated confidence level.
8 The MCB-Platform may then update the aggregated confidence score 692 to determine
9 whether it meets the threshold value 695.

10 **[00206]** For example, if a transaction request requires a threshold of 0.5, a first
11 received transaction request indicates a different terminal address than previously
12 stored in the merchant database, but a singular transaction request is assigned a
13 confidence level of 0.3, and fails to meet the threshold. If a second transaction request
14 indicating the different terminal address, the new confidence score of the different
15 terminal address would be $0.3 \times 2 = 0.6$, which satisfy the threshold requirement.

16 **[00207]** In one implementation, the MCB-Platform may score the received
17 information progressively. If all information has been exhausted but the accumulated
18 confidence score fails to meet the threshold, the MCB-Platform may decline the
19 merchant update 699.

20 **[00208]** FIGURES 7A-7L provide exemplary UIs illustrating consumer registration
21 within various embodiments of MCB-Platform. FIGURES 7A provides an exemplary
22 consumer registration screen 700a within implementations of MCB-Platform. As shown
23 in FIGURE 7A, a consumer may create a MCB-Platform card, and select interested

1 rewards type 703, such as but not limited to airline miles 705, cash back 706, charitable
2 contributions 707, retail rewards 708, and/or the like.

3 **[00209]** As shown in FIGURE 7B, a consumer may set up cardholders 710 at an
4 exemplary UI screen 700b. For example, the consumer may select a card device type,
5 such as, but not limited to plastic card 710a, mini card 710b, Visa payWave Micro tag
6 card 710c, mobile payment 710d, and/or the like.

7 **[00210]** As shown in FIGURE 7C, a consumer may choose a card image 715 at an
8 exemplary UI screen 700c. For example, the consumer may scroll to view a list of card
9 images and select a desired one. In further implementations, the consumer may elect to
10 upload a picture from his computer, and the MCB-Platform may generate a customized
11 card image based on the consumer uploaded picture.

12 **[00211]** As shown in FIGURE 7D, a consumer may complete card setup 716 at an
13 exemplary UI screen 700d. For example, the consumer may be allowed to add
14 additional cardholders to the added account 718, e.g., a spouse shared card, a parent
15 credit card, a group account, and/or the like.

16 **[00212]** As shown in FIGURE 7E, a consumer may select card benefits 720 at an
17 exemplary UI screen 700e. For example, the consume may choose from a list of
18 benefits, such as but not limited to emergency card replacement and cash disbursement
19 720a, lost/stolen card reporting 720b, zero liability 720c, and/or the like.

20 **[00213]** As shown in FIGURE 7F, a consumer may select special packages at an
21 exemplary UI screen 700f. For example, the consume may choose from a list of special
22 packages, such as but not limited to travel package 722a, retail package 722b, and/or the
23 like. Further details of the travel package are provided at the exemplary screen 700g in

1 FIGURE 7G; and a list of add-on options are provided at the exemplary screen 700h in
2 FIGURE 7H.

3 **[00214]** As shown in FIGURES 7I-7K, upon completion of registration, the
4 consumer may view a summary of terms and conditions of the new card 700i, summary
5 of the core benefits 700j, and a greeting screen including confirmation of the card
6 enrollment 700K.

7 **[00215]** As shown in FIGURE 7L, a consumer may configure notification
8 parameters 730 at an exemplary UI screen 700l. For example, the consumer may select
9 interested notification contents, notification channels (e.g., email, text messages,
10 automated voice messages, etc.), and/or the like.

11 **[00216]** FIGURES 8A-8H provide exemplary UIs illustrating consumer
12 registration and transaction with MCB-Platform within embodiments of the MCB-
13 Platform. As shown in FIGURE 8A, a consumer may view a MCB-Platform registration
14 banner, e.g., a “Visa V” logo 810, at a credit card account screen 800a. The consumer
15 may select to sign up 805 to add the credit card to the MCB-Platform.

16 **[00217]** As shown in FIGURE 8B, the consumer may proceed to registration 800b
17 with a pop-up window, requesting the consumer to create an account 816, by entering
18 an email address 818, password, etc.

19 **[00218]** As shown in FIGURE 8C, the consumer may view a pop-up window 825 in
20 the exemplary UI 800c, which provide a list of feature stores 820 with the MCB-
21 Platform. The featured stores may provide offers and coupons to the consumer via the
22 registered account.

1 **[00219]** As shown in FIGURE 8D, the consumer may visit a featured merchant site,
2 e.g., Esty, etc to receive offers 800d. For example, the consumer may view a MCB-
3 Platform logo requesting "connect now" 822 so that the shopping experience at the
4 merchant site may be linked to the consumer's MCB-Platform wallet.

5 **[00220]** As shown in FIGURE 8E, the consumer may visa a MCB-Platform pop-up
6 828 requesting permission of the user to connect to wallet at an offer bridging screen
7 800e. The consumer may click "allow" to connect to his wallet 830.

8 **[00221]** As shown in FIGURE 8F, the consumer may view offer bridging
9 information 835 at the offer bridging screen 800f. For example, the MCB-Platform may
10 provide offer suggestions based on the consumer's recent purchase history.

11 **[00222]** As shown in FIGURE 8G, the consumer may proceed to payment at an
12 offer redemption and payment screen 800g. For example, the consumer may view an
13 order summary 837, and select to pay with MCB-Platform 840.

14 **[00223]** As shown at 800h in FIGURE 8H, the consumer may view a pop-up
15 window of the order summary 845, which allows the consumer to select a payment card
16 associated with the wallet, and confirm payment 848.

17 **[00224]** FIGURES 9A-9H provide exemplary UIs illustrating merchant distributing
18 offers over social media with MCB-Platform within embodiments of the MCB-Platform.
19 In one implementations, the MCB-Platform may recognize consumers at the point-of-
20 transaction as someone the merchant values and someone who 'Likes' the merchant on
21 social media, e.g., the consumer's wallet becomes a physical "Checkin." In one
22 implementation, the MCB-Platform may issue offers to a consumer via a social media
23 platform, as a "bonus" to the consumer for sharing information with regard to a

1 merchant, e.g., comments, "likes," news feeds, "check-in" at merchant stores, and/or the
2 like. In one implementation, the MCB-Platform may populate offers and
3 communication messages to the social media via API calls, e.g., Facebook API, etc. For
4 example, wallet APIs may be facilitated with Facebook API. An exemplary Javascript
5 API to initiate payment flow UI may take a form similar to:

```
6 FB.ui({method:"payment, id: A547B243, onComplete: myCallback});
```

7

8 **[00225]** As shown at the offer bridging screen 900a in FIGURE 9A, the merchant
9 may manage campaign 905a on Facebook by designing a coupon. In one
10 implementation, the merchant may specify campaign parameters, such as campaign
11 budget, campaign schedule, campaign pricing details, and/or the like.

12 **[00226]** As shown at the UI 900b in FIGURE 9B, the merchant may further
13 configure campaign targeting parameters 905c, including locations, demographics,
14 likes/interest, connections on Facebook, and/or the like.

15 **[00227]** As shown at the UI 900c in FIGURE 9C, the merchant may view a graphic
16 presentation 910 of the campaign performance. As shown at the UI 900d in FIGURE
17 9D, a consumer may view a prompt offer via his social media page 915. As shown at the
18 UI 900e in FIGURE 9E, the consumer may select to redeem the offer via MCB-Platform
19 920, wherein the offer has already been associated with the consumer's wallet (e.g., a
20 Visa card). As shown at the UI 900f in FIGURE 9F, a social media message of the offer
21 may be automatically posted to the consumer's social media page 925, which may
22 indicate sharing with friends triggers new offers.

23 **[00228]** FIGURES 9G-9H provides exemplary mobile screens 900g-900h of offer
24 bridging via MCB-Platform within implementations of the MCB-Platform. In one

1 implementation, a consumer may operate a smart phone (e.g., an Apple iPhone) to
2 register with, and receive offers from the MCB-Platform. A consumer may maintain a
3 profile at the MCB-Platform including his name, account number, security code, PIN,
4 address, social security, GPS location, merchant account ID, and/or the like. In another
5 implementation, as shown at 930 in FIGURE 9G, and/or 940 in FIGURE 9H, the
6 consumer may retrieve reward offer from his mobile electronic wallet account connected
7 to social media. The consumer may "check in" to a place or "tag friends" who is at the
8 transaction merchant store with him 933 via social media, and may select from a
9 friends list 935 to connect with a friend who's at the same transaction merchant store.
10 As shown in FIGURE 9H, the MCB-Platform may populate an automatic message on the
11 consumer's social media profile showing their locations at the merchant store 945.

12 **MCB-Platform Universal Value Equivalents (UVE)**

13 **[00229]** FIGURES 10A and 10B show block diagrams illustrating universal value
14 equivalents within various example embodiments of the MCB-Platform. FIGURE 10A
15 shows a block diagram illustrating exemplary aspects of transforming value equivalent
16 exchange instructions in some embodiments of the MCB-Platform. In some
17 implementations, a user may desire to utilize purchasing power available to the user to
18 obtain a desired product. For example, the user may be shopping online, playing a
19 virtual online game (e.g., poker), trading on the stock market electronically, engaging in
20 foreign exchange transactions, and/or other like transactions. In some
21 implementations, the user may retain such purchasing power in various types of
22 currency. In some implementations, the user may have retained purchasing power in
23 various currency types across various ecosystems. For example, the user may have

1 access to currencies such as, but not limited to: a financial account (checking, savings,
2 debit card, credit card, open and closed loop gift cards, prepaid cards, current account,
3 money market, etc.) storing currency of a real-world monetary system (e.g., U.S. dollar,
4 Yen, Euro, etc.), an electronically tradable financial instrument (e.g., derivative financial
5 products, securities, bonds, etc.), virtual currency (e.g., online poker chips, Farmville
6 seeds, etc.), rewards program currency (e.g., rewards points, airline miles, hotel credits,
7 rental car rewards, cruise line rewards, credit card rewards points, cashback rewards,
8 etc.), and/or the like. In some implementations, the user may desire to convert
9 purchasing power available in one currency ecosystem to another currency utilized in a
10 completely different ecosystem. As one example, the user may desire to convert points
11 from traditional rewards programs into cash withdrawn from an ATM-linked account.
12 As another example, the user may desire to convert rewards points from an airline miles
13 program into virtual currency that can be utilized in an online social networking game,
14 e.g., Farmville. As another example, the user may desire to convert virtual currency into
15 currency usable to purchase stock on an electronic trading system. As another example,
16 the user may desire to convert a combination of currencies from financial accounts
17 storing currency of a real-world monetary system, electronically tradable financial
18 instruments, virtual currencies, rewards program points/miles/rewards, and/or the like
19 into a different combination of such currencies.

20 **[00230]** In some implementations, a user may desire to aggregate purchasing
21 power from a variety of source, and apply the purchasing power towards executing a
22 single transaction. For example, with reference to FIGURE 10A, a user 1001a may desire
23 to execute a transaction with a user 1001b. In some implementations, the user 101a may
24 communicate with user 1001b to execute the transaction via a universal value exchange

1 controller 1003. In some implementations, the user may optionally communicate with
2 intermediary merchants, exchanges, banks, and/or the like (e.g., 1002, 1004). For
3 example, the user may communicate with an electronic trading system (e.g., 1002a,
4 1004a) to execute a transaction. As another example, the user may communicate with a
5 bank (e.g., 1002b, 1004b) to conduct a transaction. As yet another example, the user
6 may communicate with a merchant system (e.g., 1002, 1004) for purchasing goods
7 and/or services. In some implementations, a user 1001a may provide cross-ecosystem
8 currency exchange instructions to the universal value exchange controller 1003. For
9 example, the user 1001a, in such instructions, may specify source details (of user 1001a)
10 such as, but not limited to: currency source types, currency account numbers, currency
11 access usernames, currency access passwords, and/or the like, as well as destination
12 details (of user 1001b) such as, but not limited to: currency destination types, currency
13 account numbers, currency access usernames, currency access passwords, and/or the
14 like. In some implementations, the universal value exchange controller 1003 may
15 obtain the cross-ecosystem currency exchange instructions from user 1001a. The
16 universal value exchange controller may determine the sources of currency, and
17 determine the types of currency available at the sources of the currencies. The universal
18 value exchange controller may determine exchange rates for each of the source
19 currencies, for example, relative to a standard currency (e.g., U.S. dollar, Euro, Yen,
20 privately created currency standard, and/or the like). The universal value exchange
21 controller may also determine whether there are any restrictions or conditions at each of
22 the sources of the currencies, as well as the destinations of the currencies. For example,
23 a rewards points program may have a restriction against converting its rewards points
24 into rewards points of another rewards points program; a condition that conversion can

1 only take place if fewer than a threshold amount of rewards points are utilized, and/or
2 the like. Each of the source currencies may have various restrictions and/or conditions
3 on use of the currency type of the source.

4 **[00231]** In some implementations, the universal value exchange controller may
5 obtain the restrictions and/or conditions of the sources and destinations of the
6 currencies, and may determine a currency exchange flow path based on the restrictions
7 and/or conditions at the currency sources and/or destinations. Upon determining a
8 currency exchange flow path, the universal value exchange controller 1003 may provide
9 request messages to the components in the currency exchange flow path, e.g., exchanges
10 (e.g., 1002a, 1004a), banks (e.g., 1002b, 1004b), merchants (e.g., 1002, 1004) and/or
11 the like, requesting the components to provide and/or accept currency value, based on
12 the determined currency exchange flow path. Upon completing the currency withdrawal
13 and/or deposits into each of the currency accounts involved in the cross-ecosystem
14 currency exchange, the universal value exchange controller may provide notifications to
15 the users 1001a, 1001b notifying them of completion of the requested cross-ecosystem
16 currency transaction. Various currency exchange flow paths of the MCB-Platform
17 embodiments are discussed throughout the specification.

18 **[00232]** With reference to FIGURE 10B, the MCB-Platform controller 1016 may act
19 as a gateway or a single point of access between program providers 1010, point
20 aggregators 1014, merchants 1020 and users 1018. In some implementations, program
21 providers 1010 may enter into an agreement with the MCB-Platform to participate in the
22 points/currency exchange 1012a via the MCB-Platform gateway. The program providers
23 may, via program configuration user interface (UI), identify one or more partner

1 program providers with whom the program provider may enter into exchange
2 transactions. For example, the program provider may select non-competing program
3 providers and/or affiliated program providers as partners. For program providers, the
4 facilities of the MCB-Platform platform may be an opportunity to unload the value of
5 their promotions which are carried on their balance sheets as liability. For example,
6 program providers may have customers who are holding on to their points because they
7 do not have enough points to redeem an item, for example, a ticket or a room. However,
8 at the aggregate level, there may be a significant liability for the program providers
9 because of the unredeemed points. In such a situation, allowing the customers to
10 participate in points/currency exchange may be an advantage to the program providers.

11 **[00233]** In some implementations, the program provider may also set an exchange
12 rate with respect to each of the selected program provider partners. The exchange rate,
13 in some implementations, may be established via bilateral agreement between the
14 program provider and each partner. In such a situation, there may be no need for a base
15 or intermediate currency. For example, United Airlines may enter into a bilateral
16 agreement with Hilton and establish an exchange rate where 5 United Airline miles can
17 be exchanged for 1 Hilton Honors point. In some other implementations, the exchange
18 rate may be established using a base/intermediate currency. The intermediary may be,
19 for example, a MCB-Platform currency (e.g., MCB-Platform point) or a non-
20 denominational currency (e.g., a unit). In such a case, the program provider may need to
21 negotiate with the MCB-Platform to set the exchange rate between the provider
22 currency and the MCB-Platform currency. These bilateral agreements may be carried
23 out electronically. As a part of the program provider enrollment, the program provider
24 may need to expose API(s) to their rewards/loyalty program such that the MCB-

1 Platform may obtain currency balance information and may credit/debit currency after
2 an exchange transaction. Referring to FIGURE 10B, the program providers 1010 may
3 include various types of entities or business users 1010a-c such as issuers/banks,
4 merchants, acquirers, virtual/social games, and/or the like. In some implementations,
5 the MCB-Platform may also facilitate points/currency exchange between one or more
6 program providers that are not enrolled as a program provider in the MCB-Platform
7 platform.

8 **[00234]** In some implementations, the MCB-Platform may also act as a gateway to
9 point aggregators 1014. For example, MCB-Platform may transact with point
10 aggregators to sell off or buy points when necessary. In some other implementations,
11 various merchants 1020 such as Amazon, may also utilize the facilities of the MCB-
12 Platform gateway to access the points/currencies from various program providers, and
13 allow customers to use the value of their points/currencies towards payment of
14 purchases made via the merchant. In some implementations, at the back end standard
15 settlement processes may be employed. In some implementations, such redemption
16 may be for online purchases or brick and mortar purchases using an electronic or
17 mobile wallet, a physical payment device or other methods. Further, redemption may
18 occur prior to a transaction or dynamically at the time of transaction.

19 **[00235]** From the point of view of a user 1018, the MCB-Platform provides a single
20 place where points/currencies from various program providers 1010 can be managed,
21 redeemed, exchanged 1012b, or linked to a wallet. Further, via the MCB-Platform, the
22 user may have the flexibility to make a redemption dynamically at the time of purchase
23 or prior to the purchase. The user may also have the option to combine

1 points/currencies during the redemption. In some implementations, the user may also
2 swap and liquidate points/currencies and open and closed loop gift cards.

3 **[00236]** FIGURES 10C and 10D show data flow diagrams illustrating MCB-
4 Platform program configuration embodiment of the MCB-Platform. In one
5 embodiment, the MCB-Platform may behave as a loyalty broker creating a marketplace
6 or an exchange for converting points, rewards and virtual currencies to real world
7 currencies. The loyalty broker embodiment may allow any point provider partner to
8 establish their own price for points/currencies. The loyalty broker may, in some
9 embodiments, allow a consumer to enroll and exchange points/currencies to a
10 proprietary currency (e.g., Visa Points+) or even cash. The proprietary currency may
11 then be used in inline or other purchases.

12 **[00237]** In one implementation, a partner 1024 may configure an exchange
13 program 1040 with a loyalty broker 1028. At 1050, the partner may provide bank
14 identification number (BIN), logos, accept any terms and conditions of the program,
15 and/or the like to create and/or update the exchange program. If the partner does not
16 have a BIN, one may be created. The BIN creation may be handled by an admin server
17 1026 or the loyalty broker server. At 1052, the information provided by the provider
18 and/or confirmation of the exchange program creation may be provided to the loyalty
19 broker 1028.

20 **[00238]** Once the program has been configured, the partner or the partner's
21 rewards program administrator 1030 may set exchange rates and other conditions
22 applicable to the exchange program 1042. In some implementations, the configuration
23 may be performed by the provider accessing a configuration UI in a merchant/provider

1 self-service portal 1032. In some implementations, at 1054a, the provider may set the
2 exchange rate for its points/currencies. The exchange rate may specify point/currency to
3 MCB-Platform point ratio. For example, the program provider may set the exchange
4 rate where the 105,000 miles (the provider's currency) is equivalent to 1 MCB-Platform
5 point. In one implementation, the value of the MCB-Platform point may be with respect
6 to a monetary currency such as US dollar, Canadian dollar, Yen, etc. For example, 1
7 MCB-Platform point may be equivalent to one US dollar. In one implementation the
8 price for points may be changed as frequently as the partner wishes to change it. For
9 example, it could be changed daily, weekly, monthly, yearly, etc. The exchange rate may
10 be associated with a time period for which it is effective in some implementations.

11 **[00239]** In some implementations, the partners may set exchange rules/rates for
12 various customer segments or even one customer segment. In some other
13 implementations, partners may set up exchange rules at the product (e.g., Stock-
14 Keeping Unit SKU) level. For example, some partners may wish to run a promotional
15 type of exchange rules that may not apply across the partner's business overall, but may
16 be applicable for a short period of time or a small or select group where it may not be
17 applicable or convenient to set up a separate program. In one implementation, for
18 example, a partner may set an exchange rule where customers who fall into Chase
19 segment 82C would get a different exchange rate from customers who fall into other
20 segments. In yet another implementation, for example, a partner may set an exchange
21 rule where customers who enrolled in the partner program in the last 30 days would
22 receive a special exchange rate on purchases of select items (e.g., SKU level data) at
23 another merchant (e.g., Best Buy).

1 **[00240]** At 1054b, the partner may specify rules and restrictions for any exchange
2 of the program provider's points/currencies. In some implementations, the rules and
3 restrictions may be negotiated between the provider and the loyalty broker. In other
4 implementations, the rules and restrictions may be specified via the configuration UI.
5 For example, the provider may set a minimum redemption group of 500 (e.g., redeem in
6 groups of 500 miles). In some implementations, the partner may also provide or upload
7 a pre-enrollment file at the self-service portal at 1056. Such a pre-enrollment file may
8 include information relating to customers of the program provider (e.g., customer
9 reward ID or membership ID, name, address, etc.). The pre-enrollment file may be
10 stored in one or more databases of the loyalty broker and may be used to validate users
11 when they enroll in the loyalty broker. In one implementation, at 1058 the partner may
12 also access the self-service portal to fetch reports. Example of reports available to the
13 partner provider may include report of exchange activities by customer and/or time
14 period, report on customer enrollment, and/or the like.

15 **[00241]** Once the exchange program is configured and the exchange rate and
16 conditions set up, the loyalty broker may accept customer enrollment 1044. The
17 customer may enroll in the exchange program with the loyalty broker by accessing a
18 customer facing portal, a web or mobile application, a wallet having loyalty broker
19 facilities. At 1060, the customer 1034 provide program details such as membership ID,
20 password, and any other information necessary to verify the customer as the owner of
21 the membership account. At 1060, the customer may also provide usage and other
22 preferences (e.g., use my MCB-Platform points for travel, gas, any purchase, when I
23 send a text, exchange my miles as soon as they exceed 25,000, exchange my miles when
24 the exchange rate is better than or equal to 100:1, etc.). At 1062, the loyalty broker may

1 receive the customer provided program details and may verify the details to confirm the
2 customer ownership of the membership account with the reward provider. Alternatively,
3 the loyalty broker may also utilize information in the pre-enrollment file to confirm
4 some or all of the customer/program details. At 1064, the program provider may
5 confirm the membership of the customer to the loyalty broker. At 1064, the program
6 provider may also provide the customer in question's current points/currency balance
7 information to the loyalty provider.

8 **[00242]** Referring to FIGURE 10D, the customer may access and view loyalty
9 exchange rates 1046. At 1066, the customer 1034 may fetch a landing age or launch an
10 application to view the program balance information and exchange rates. The loyalty
11 broker, in response to the customer's request, may obtain from the loyalty provider the
12 current exchange rates as well as points/currency balances and display the information
13 to the customer at 1068. In one implementation, the customer may initiate a
14 points/currency exchange transaction 1048. For example, at 1070, the customer 1034
15 may instruct the loyalty broker to exchange an amount of program points/currency
16 (e.g., 25,000 miles) for an equivalent value (e.g., 225 MCB-Platform points) 1070. At
17 1072, the loyalty broker may process the instruction by requesting the program provider
18 1036 to reduce the customer's program points/currency by the specified amount (e.g.,
19 reduce by 25,000 miles). The program provider may reduce the customer's
20 points/currency and make payment of the agreed upon amount (e.g., \$250) at 1074. In
21 one implementation, as a part of the agreement between the program provider and the
22 loyalty broker, the loyalty broker may assess a transaction processing fee. In some
23 implementations, the fee may be a percentage of the total amount the program provider
24 has approved for billing. For example, when the program provider agrees to exchange

1 25,000 miles for \$250, the loyalty broker may assess a 20% processing fee which is
2 equivalent to \$50. In some implementations, the loyalty broker may advertise the
3 exchange rate using the adjusted amount that is actually payable to the customer. For
4 example, the loyalty broker advertises to exchange 25,000 miles for \$225. In some
5 implementation, instead of assessing a processing fee on a per transaction basis,
6 subscription type fees may be assessed to partners and/or users of the MCB-Platform.
7 For example, the subscription fee amount may be tiered based on volume of MCB-
8 Platform transactions. In some other implementations, there may be revenue share
9 between the MCB-Platform and partners. In yet other implementations, MCB-Platform
10 may add and/or retain a certain number of basis points to the exchange rate, assess
11 subscription or per-use fees to the consumer or levy a percentage of the exchange value
12 as fees to the consumer/partner in exchange for the services provided.

13 **[00243]** When the bill is paid, the customer portion is credited to the MCB-
14 Platform points BIN or a Debit Processing Service (DPS) type BIN for each card. In
15 some implementations, the customer may be issued a prepaid card having the value of
16 the total MCB-Platform points obtained from the exchange. At 1076, the exchange is
17 complete. The customer's MCB-Platform points balance is incremented by the total
18 MCB-Platform points gained (e.g., +225), his/her miles balance is decremented by the
19 number of miles used in the exchange (e.g., 25,000 miles). The examples discussed
20 herein assume that a unit MCB-Platform point is equivalent to \$1. Other equivalency
21 between the MCB-Platform point and currency are contemplated in some
22 implementations of the loyalty broker.

23 **[00244]** Some embodiments of the MCB-Platform facilitate gift card exchanges and

1 conversions. The facilities of the MCB-Platform may support open loop, closed loop and
2 hybrid gift cards. Open loop gift cards can be redeemed in a variety of businesses, while
3 closed loop gift cards can be redeemed at a specific business (e.g., Apple Store card, Best
4 Buy card) or select businesses (e.g., Westfield mall gift card). For example, a user A may
5 have a gift card for the Apple Store, but the user never shops in the Apple Store, and
6 would instead prefer to exchange the Apple gift card for a Best Buy gift card. Similarly,
7 another user B may have a Best Buy gift card, but would like to exchange for an Apple
8 Store gift card. In such a situation, the MCB-Platform may facilitate the exchange of the
9 Apple and Best Buy gift cards such that both users A and B can have their preferred gift
10 cards. As another example, a user may have various gift cards in his or her hands or in
11 the wallet. The user may prefer to combine the value of all the gift cards in one gift card
12 or prepaid card, a bank account or obtain cash. In such as situation, the MCB-Platform
13 may provide facilities to consolidate the gift card values and automatically apply them in
14 a purchase transaction.

15 **[00245]** FIGURE 11A shows a data flow diagram illustrating closed loop gift card
16 value exchange embodiment of the MCB-Platform. The data flow diagram shows flow of
17 data between a user 1102a, a client 1104a, a MCB-Platform server(s) 1106a and gift card
18 issuer servers 1108a/210a, and target gift card issuer server(s) 1107a over a
19 communication network 1113a. As shown in the figure, the user 1102a may access the
20 MCB-Platform web page or application using the client 1104a to communicate with the
21 MCB-Platform server. In some implementations, the user may wish to transfer the value
22 from one gift card to another. The user may then input or select a source gift card and a
23 target gift card and request value transfer from the source gift card to the target gift card
24 at 1112. In some implementations, the client may include, but is not limited to: a

1 personal computer, mobile device, television, point-of-sale terminal, kiosk, ATM,
2 and/or the like (e.g., 1104a). In various implementations, the user input may include,
3 but not be limited to: a single tap (e.g., a one-tap mobile app purchasing embodiment)
4 of a touchscreen interface, keyboard entry, card swipe, activating a RFID/NFC enabled
5 hardware device (e.g., electronic card having multiple accounts, smartphone, tablet,
6 etc.) within the user device, mouse clicks, depressing buttons on a joystick/game
7 console, voice commands, single/multi-touch gestures on a touch-sensitive interface,
8 touching user interface elements on a touch-sensitive display, and/or the like.

9 **[00246]** In some implementations, using the user's input, the client may generate a
10 transfer request, e.g., 1114 and provide the transfer request to the MCB-Platform server.
11 For example, the client may provide a (Secure) Hypertext Transfer Protocol
12 ("HTTP(S)") POST message including data formatted according to the eXtensible
13 Markup Language ("XML"). An example transfer request 1114, substantially in the form
14 of a HTTP(S) POST message including XML-formatted data, is provided below:

```
15 POST /transferrequest.php HTTP/1.1
16 Host: www.visa.com/uve
17 Content-Type: Application/XML
18 Content-Length: 484
19 <?XML version = "1.0" encoding = "UTF-8"?>
20 <transfer_request>
21     <request_ID>45DSKFSWFG5</request_ID>
22     <timestamp>yyyy-mm-dd hh:mm:ss</timestamp>
23     <user_ID>JDoe@gmail.com</user_ID>
24     <source_details>
25         <giftcard_ID>4444566897978766</giftcard_ID>
26         <issuer_ID>apple</issuer_ID>
27         <card_value>100</card_value>
28         <currency>usd</currency>
29     </source_details>
30     <destination_details>
31         <giftcard_ID>5555566823457899</giftcard_ID>
32         <issuer_ID>bestbuy</issuer_ID>
33     </destination_details>
34     <client_details>
35         <client_IP>192.168.23.122</client_IP>
36         <client_type>smartphone</client_type>
37         <client_model>HTC Hero</client_model>
38         <OS>Android 2.2</OS>
```

```
1         <app_installed_flag>true</app_installed_flag>
2     </client_details>
3 </transfer_request>
```

4 **[00247]**

5 **[00248]** The MCB-Platform server may receive the transfer request 1114 and may
6 extract the details of the transfer request (e.g., XML data). In one implementation, the
7 MCB-Platform server may identify the issuer of the source gift card 1110a and may send
8 a balance request 1116 to the issuer of the source gift card 1110a. In one implementation,
9 the request 1116 may be in the form of a web service/API call. The gift card issuer server
10 may return the balance information message 1120 to the MCB-Platform server. At 1122,
11 the MCB-Platform server may determine equivalent value that the user may obtain after
12 the exchange. Determination of the equivalent value may be based on risk exposure, the
13 details of which are discussed with respect to FIGURES 12A-B.

14 **[00249]** In one implementation, the MCB-Platform server may send to the client a
15 request 1124 that the user confirm acceptance of the equivalent value. For example, the
16 MCB-Platform server may provide an HTML page to the client. The client may display,
17 for example, a summary of the transfer request identifying the source and destination
18 gift cards, the equivalent value of the destination gift card, terms and conditions,
19 buttons to accept or cancel the exchange, and/or the like. At 1126 the user may confirm
20 acceptance of the equivalent value, which may then be passed on as the confirmation
21 message 1128 by the client to the MCB-Platform server.

22 **[00250]** In one implementation, the MCB-Platform may have a number of gift card
23 accounts associated with a number of merchants. For example, the MCB-Platform may
24 have a gift card account for Apple, Best Buy, Macys, Barneys, and/or the like. These gift
25 card accounts may be referred to as pool gift card accounts. In one implementation, the

1 MCB-Platform server may send a balance transfer request 1130 to the source gift card
2 issuer server 1110a. The balance transfer request 1130 may include information such as
3 source gift card ID, pool source gift card ID, transfer amount, and/or the like. In one
4 implementation, the pool source gift card ID may correspond to a gift card issued by the
5 source gift card issuer and owned and maintained by the MCB-Platform (e.g., MCB-
6 Platform's apple gift card). In one implementation, the source gift card issuer server
7 may transfer the balance from the source gift card (e.g., the user's Apple gift card) to the
8 pool source gift card (e.g., MCB-Platform's Apple gift card) and may send a confirmation
9 message 1132 including the updated pool source gift card balance to the MCB-Platform
10 server. In one implementation, the source gift card issuer server may send the client the
11 updated source gift card balance 1136 confirming the transfer of the source gift card
12 value. In one implementation, the MCB-Platform server may send a target gift card
13 order 1138 to the target gift card issuer. The target gift card order may include a request
14 to transfer the determined equivalent value from the pool target gift card to a target gift
15 card. An example target gift card order 1138, substantially in the form of a HTTP(S)
16 POST message including XML-formatted data, is provided below:

```
17 POST /targetorder.php HTTP/1.1
18 Host: www.merchant.com/order
19 Content-Type: Application/XML
20 Content-Length: 484
21 <?XML version = "1.0" encoding = "UTF-8"?>
22 <giftcard_order>
23     <source_card_ID>2345678745674589</source_card_ID>
24     <target_card_ID>3486549865445678</target_card_ID>
25     <timestamp>yyyy-mm-dd hh:mm:ss</timestamp>
26     <user_ID>uve_order@visa.uve.com</user_ID>
27     <target_card_value>100</target_card_value>
28     <password>uvegiftcardpasssword</password>
29     <delivery_email>jdoe@gmail.com</delivery_email>
30     <delivery_message>standard</delivery_message>
31 </giftcard_order>
```

32 **[00251]**

33 **[00252]** The target gift card issuer server may then issue a target gift card having

1 the equivalent value to the user. The target gift card issuer server may send the client
2 the target gift card issue message 1140. In one implementation, the target gift card issue
3 message 1140 may include the target gift card ID which the user may obtain
4 electronically and utilize for purchase with the merchant associated with the target gift
5 card. An example target gift card issue message 1140 formatted in XML is provided
6 below:

```
7   <target_gift_card>
8       <target_card_ID>3486549865</target_card_ID>
9       <timestamp>yyyy-mm-dd hh:mm:ss</timestamp>
10      <target_card_value>100</target_card_value>
11      <activation_status>activated</activation_status>
12      <delivery_email>jdoe@gmail.com</delivery_email>
13      <delivery_message>Thank you for ... </delivery_message>
14  </target_gift_card>
```

15
16 **[00253]**

17 **[00254]** At 1142, the MCB-Platform server may store updated pool source gift card
18 balance (e.g., previous balance incremented by the value of the source gift card) and the
19 updated pool target gift card balance (e.g., previous value decremented by the
20 equivalent amount). In some embodiments of the MCB-Platform, when the balance in
21 any one of the pool gift cards exceeds a threshold, the MCB-Platform may initiate a sell
22 off. In one implementation, the sell off may involve issuing gift cards and selling them at
23 a discount. For example, the MCB-Platform may accumulate over time an excess
24 balance of \$10000 in one or more merchant gift card accounts. The MCB-Platform may
25 then issue (e.g., via the gift card issuer) 100 gift cards each worth \$100. The MCB-
26 Platform may then sell each gift card at a discount to users to collect some revenue. The
27 MCB-Platform may aggregate such excess balances over time by apportioning value
28 from records in the MCB-Platform database, e.g., value card 11219u. For example, when
29 source and destination field values in the value card table record reach \$0 and yet there

1 is residual value left on the card, that residual value may be used to generate such excess
2 balances for the MCB-Platform. In one example, the MCB-Platform may observe
3 consumers making purchases with merchants accepting such value; e.g., the MCB-
4 Platform may be made part of a payment network which may parse PAN/account
5 identifiers and compare such account identifiers embedded in transaction
6 request/authentication with records in the MCB-Platform database, e.g., users 11219a,
7 accounts 11219g, etc., tables. In those instances, the MCB-Platform may take a credit
8 and use its points/value equivalence to pay for the consumer's purchase and take direct
9 charge from the consumer's payment source for that value. In one embodiment the user
10 would not be aware that the purchase was made using the pool points equivalence. In an
11 alternative embodiment, the MCB-Platform would show up on the consumer's bills as
12 the merchant taking the charge for the value of the item. In yet another embodiment,
13 the user may be offered a discount on the item (e.g., the consumer would be charged
14 10% less from their payment source while the merchant would receive full value in point
15 equivalence supplied by the MCB-Platform), thereby providing a liquidation method for
16 the MCB-Platform to obtain currency exchange for its pool points/currency.

17 **[00255]** FIGURES 12A-B show logic flow diagrams illustrating closed loop gift card
18 value exchange embodiments of the MCB-Platform. The closed loop gift card value
19 exchange may begin at 1206. At 1208, client 1201 may send instructions to transfer
20 value from source gift card to a target gift card. The instructions may identify the source
21 gift card and the target gift card. For example, the source/target gift card number may
22 be included in the instructions. The instructions may be received by MCB-Platform
23 server 1202. The MCB-Platform server may parse the instructions to obtain identifiers
24 of the gift cards at 1210. The MCB-Platform server may further identify the issuers of the

1 gift cards at 1212. At 1214, the MCB-Platform server may request the source gift card
 2 issuer server 1203 to provide the balance in the source gift card. At 1216, the source gift
 3 card server may receive the request and may query one or more tables and/or databases
 4 to obtain the source gift card balance. The source gift card issuer server may provide the
 5 requested balance summary to the MCB-Platform server at 1218. The MCB-Platform
 6 server may receive the balance information at 1220 and may obtain historical data
 7 relating to the source/target gift card value transfer at 1222. In one implementation, the
 8 historical data may be obtained by querying one or more tables and/or databases using
 9 the source gift card ID and/or target gift card ID. At 1226, the MCB-Platform server may
 10 use the historical data to determine risk exposure for the exchange transaction in
 11 question. In one implementation, for example, the risk exposure determination may be
 12 based on rate of source/target gift card transactions and predefined risk thresholds.
 13 Table 1 below shows example risk thresholds, risk exposure and risk exposure weights
 14 for gift card exchange transactions.

Thresholds [Transactions/Day]	Risk Exposure	Risk Exposure Weight
100	Minimal risk	0.9
50	Low risk	0.8
25	Medium risk	0.6
10	High risk	0.3
5	Unacceptable risk	0

15 **[00256]** Table 1

16 **[00257]** In some implementations, at 1228, the MCB-Platform server may
 17 determine liquidity of the source/target gift cards. For example, the MCB-Platform may
 18 query one or more databases and/or tables to determine the balance in the pool target
 19 gift card, and the approximate number of target gift cards the balance may support. In
 20 one implementation, the MCB-Platform may use the source/target transaction rate and

1 the number of target gift cards in the MCB-Platform pool to calculate a liquidity ratio. In
2 a further implementation, a liquidity ratio greater than 1 may be indicative of high
3 liquidity, while a ratio less than 1 may indicate low liquidity. At 1230, based on the risk
4 exposure and/or the liquidity, the MCB-Platform may determine an exchange rate for
5 the source/target gift card exchange. For example, when the liquidity ratio is greater
6 than or equal to 1, the risk exposure weight may be equivalent to the exchange rate.
7 When the liquidity ratio is less than 1, a product of the risk exposure weight and
8 liquidity ratio may determine the exchange rate. In some implementations, the
9 calculation of the liquidity ratio may be optional such that the risk exposure weight
10 alone may determine the exchange rate.

11 **[00258]** $Exchange-Rate = Weight_{RISK-EXPOSURE}$ when $liquidity \geq 1$

12 (1)

13 **[00259]** $Exchange-Rate = Weight_{RISK-EXPOSURE} \times liquidity$ when $liquidity < 1$

14 (2)

15 **[00260]** Upon determining the exchange rate, the MCB-Platform may determine
16 the equivalent value that client would receive in the form of a target gift card at 1232.
17 For example, with a source gift card valued at \$100, and an exchange rate at 0.8, the
18 target gift card may have an equivalent value of \$80. At 1234, the MCB-Platform server
19 may send a request to the client to confirm the transfer of the source gift card value to
20 the equivalent value of a target gift card. At 1236, the client may receive and display the
21 confirmation request. At 1238, the client may receive an input from the user, and may
22 send the input message to the MCB-Platform server. Referring to FIGURE 12B, the
23 MCB-Platform server may receive the input message from the client at 1240, and parse

1 the message to obtain the details. In one implementation, at 1242, the MCB-Platform
2 server may determine if the transfer is confirmed by the user. If the transfer is not
3 confirmed by the user, the transfer is canceled at 1244, concluding the process at 1246.
4 However, if the transfer is confirmed at 1242, the MCB-Platform server may request the
5 source gift card issuer to transfer balance of the source gift card to the pool source gift
6 card at 1248. The source gift card issuer server may receive the transfer request and may
7 transfer the balance as requested at 1250. In one embodiment, a web services request
8 that initiates the transfer from one specified card account number to a destination
9 account number may be issued. A web request that may otherwise have been initiated
10 when a user wishes to move value from one account to another may be captured, but
11 instead of using the same user card account as a parameter in the web services call,
12 instead, a MCB-Platform value card (e.g., value equivalence held in a MCB-Platform
13 pool) may be used as either a destination or a source account parameter in the web
14 services call, e.g., to effect a transfer balance request 1250 or a transfer request 1260,
15 respectively. Such web services may vary depending on the service/program.

16 **[00261]** In one implementation, the source gift card issuer server may also send a
17 confirmation once the balance transfer has occurred. At 1256, the MCB-Platform server
18 may receive the confirmation of the balance transfer. At 1258, the MCB-Platform server
19 may request the target gift card issuer to transfer the equivalent value determined from
20 the pool target gift card to a target gift card. The target gift card issuer may receive the
21 transfer request at 1260, and may execute the requested transfer. In one
22 implementation, at 1262, upon executing the transfer, the target gift card issuer server
23 may send the issued target gift card having the equivalent value to the client. The client
24 may receive and display the target gift card at 1254. In one implementation, the target

1 gift card issuer server may send an email or text message to notify and/or provide the
2 user an electronic target gift card. In another implementation, the issued target gift card
3 may be mailed to the user's physical address. In yet another implementation, the target
4 gift card may pop up in the user's electronic wallet. In one implementation, the source
5 gift card issuer server may also send a source gift card balance confirmation (e.g., \$0
6 balance) to the client at 1252.

7 **[00262]** In one implementation, in the instance where funds cannot be reassigned
8 from a source gift card to a pool gift card, a deallocation of the source gift card in the
9 user's wallet may be effected such that the user may no longer see it or use it or
10 exchange it. The source gift card may be reallocated later to another user wanting a
11 similar exchange as further described with respect to FIGURES 11B, 12A-B. In some
12 embodiments, there may be instances of fraud where although the card is deallocated in
13 the user's wallet, the user may still effect a purchase with the physical card. In one
14 embodiment such fraud may be adjudicated upon discovery of that card no longer being
15 available for a subsequent exchange by another user. In one implementation, a charge
16 can be taken from the user's wallet (e.g., any of the funding accounts) and/or the user
17 may be given warnings or prevented from participating in such exchange programs in
18 future.

19 **[00263]** FIGURE 11B shows a data flow diagram illustrating a second closed loop
20 gift card value exchange embodiment of the MCB-Platform. Some of the gift cards that
21 users may want to exchange may be of an open loop type. In one implementation, at
22 1150, a user 1102b may request value transfer from a source gift card to a destination gift
23 card. The client 1104b may receive the user input and may generate a transfer request

1 1152. The transfer request 1152 may have similar data structure to that of the transfer
2 request 1114 of FIGURE 11B. The transfer request 1152 may be sent to the MCB-Platform
3 server 1106b. The MCB-Platform server may receive and parse the request to obtain
4 source gift card issuer ID and source gift card ID. The MCB-Platform server may send a
5 source gift card balance request 1154 to the source gift card issuer server 1110b. The
6 source gift card issuer server may look up the balance and may provide the information
7 in a gift card balance message 1156 to the MCB-Platform server. In one implementation,
8 the MCB-Platform server may send a target gift card query 1158 to the MCB-Platform
9 pool database 1117b. In one implementation, the query may return a target gift card
10 response 1160. At 1162, the MCB-Platform may determine equivalent transferable value
11 1162. In one implementation, the equivalent value may be the value that is ultimately
12 made available to the user in a target gift card. The MCB-Platform server may send a
13 request to accept transfer 1164 to the client. The client may obtain the request and may
14 render the contents of the request on the client display. The user may provide a
15 response 1166 confirming the acceptance. The client may take the user input and
16 generate a confirmation message 1168 for transfer to the MCB-Platform server. Upon
17 receiving the confirmation message 1168, the user may execute the transfer request at
18 1170. In one implementation, at 1172 the MCB-Platform server may update database
19 1119b with updated balances of the source gift card, the target gift card and destination
20 gift card. In one implementation, the MCB-Platform server may provide updated gift
21 card balances 1174 to the client such that the user may view the changes in the source
22 and destination gift card balances after the transfer.

23 **[00264]** In one implementation, when the user 1102b makes a purchase using the
24 destination gift card, the MCB-Platform server may route the charge request 1176 to the

1 target gift card issuer server 1107b. In addition to other example charge requests and
 2 authorizations provided throughout, the following is an example. An example charge
 3 request 1176, substantially in the form of a HTTP(S) POST message including XML-
 4 formatted data, is provided below:

```

5  POST /chargerequest.php HTTP/1.1
6  Host: www.targetissuer.com/charge
7  Content-Type: Application/XML
8  Content-Length: 484
9  <?XML version = "1.0" encoding = "UTF-8"?>
10 <charge_request>
11     <gift_card_ID>2345678745674589</gift_card_ID>
12     <user_ID>theoriginalowner@gmail.com</user_ID>
13     <checkout_request>
14         <checkout_ID>4NFU4RG94</checkout_ID>
15         <timestamp>2011-02-22 15:22:43</timestamp>
16         <purchase_detail>
17             <purchase_amount>100</purchase_amount>
18             <num_products>5</num_products>
19             <product_ID>AE95049324</product_ID>
20             <product_ID>MD09808755</product_ID>
21             <product_ID>OC12345764</product_ID>
22             <product_ID>KE76549043</product_ID>
23             <product_ID>SP27674509</product_ID>
24         </purchase_detail>
25         <PoS_client_detail>
26             <client_IP>192.168.23.126</client_IP>
27             <client_type>smartphone</client_type>
28             <client_model>HTC Hero</client_model>
29             <OS>Android 11.2</OS>
30             <app_installed_flag>true</app_installed_flag>
31         </PoS_client_detail>
32     </checkout_request>
33 </charge_request>
34 [00265]

```

35 **[00266]** The target gift card issuer 1110b may receive the charge request and send a
 36 charge authorization message 1178 to the MCB-Platform server. In addition to other
 37 example charge requests and authorizations provided throughout, the following is an
 38 example. An example authorization message 1178, substantially in the form of a
 39 HTTP(S) POST message including XML-formatted data, is provided below:

```

40  POST /chargeauthorize.php HTTP/1.1
41  Host: www.visa.com/uve/auth
42  Content-Type: Application/XML
43  Content-Length: 484
44  <?XML version = "1.0" encoding = "UTF-8"?>
45  <auth>

```

```

1      <gift_card_ID>2345678745674589</gift_card_ID>
2      <user_ID>theoriginalowner@gmail.com</user_ID>
3      <status>approved</status>
4      <balance>0</balance>
5      <checkout_ID>4NFU4RG94</checkout_ID>
6      <timestamp>2011-02-22 15:22:43</timestamp>
7  </auth>
8

```

9 **[00267]** The MCB-Platform server may then update the destination gift card
 10 balance at 1180.

11 **[00268]** FIGURE 11C shows a data flow diagram illustrating an open loop gift card
 12 value exchange embodiment of the MCB-Platform. In one implementation, a user 1102c
 13 may request open loop gift card value transfer at 1181. The client 1104c may receive the
 14 input and may generate a transfer request 1182 to the server 1106c. An example transfer
 15 request 1182, substantially in the form of a HTTP(S) POST message including XML-
 16 formatted data, is provided below:

```

17  POST /transferrequest.php HTTP/1.1
18  Host: www.visa.com/uve
19  Content-Type: Application/XML
20  Content-Length: 484
21  <?XML version = "1.0" encoding = "UTF-8"?>
22  <transfer_request>
23      <account_access_details>
24          <user_details1>
25              <user_ID>jdoe@gmail.com</user_ID>
26              <password>password</password>
27              <user_current_stake>100</user_current_stake>
28          </user_details1>
29          <user_details2>
30              <user_ID>jdoe@gmail.com</user_ID>
31              <password>password</password>
32              <user_current_stake>50</user_current_stake>
33          </uve_details>
34          <user_ID>uve@uve.com</user_ID>
35          <user_current_stake>50</user_current_stake>
36      </account_access_details>
37      <gift_card_ID>2345678745674589</gift_card_ID>
38      <card_issuer>xyz</card_issuer>
39      <balance>200</balance>
40      <transfer_amount>200</transfer_amount>
41      <destination_currency>uve points</destination_currency>
42      <timestamp>2011-02-22 15:22:43</timestamp>
43  <client_details>
44      <client_IP>192.168.23.122</client_IP>
45      <client_type>smartphone</client_type>
46      <client_model>HTC Hero</client_model>
47      <OS>Android 2.2</OS>
48

```

```
1         <app_installed_flag>true</app_installed_flag>
2     </client_details>
3 </transfer_request>
```

4 **[00269]**

5 **[00270]** The MCB-Platform server may then send a gift card balance request 1183
6 to the gift card issuer server 1108c to obtain the current gift card balance. The gift card
7 issuer server may look up the gift card balance information using gift card ID in the
8 request 1183. The gift card issuer server may then provide the gift card balance message
9 1184 to the MCB-Platform server. At 1185, the MCB-Platform server may determine the
10 equivalent transferable value (e.g., using process outlined in FIGURE 3D). The MCB-
11 Platform server may send a request 1186 to the client to request acceptance of the
12 equivalent value determined at 1185. The client may receive and display the request to
13 the user. At 1187, the user may confirm acceptance of the equivalent value. The client
14 may then generate a confirmation message 1188 and send the message to the MCB-
15 Platform server. At 1189, the MCB-Platform server may liquidate the gift card to an
16 equivalent value (e.g., cash, MCB-Platform points, etc.). In one implementation, the
17 user may designate the currency into which the gift card may be converted. In another
18 implementation, the MCB-Platform may allow conversion into only certain currencies
19 (e.g. MCB-Platform points). In one implementation, the equivalent amount may be
20 deposited in an account designated by the user, and may be used by the user when
21 making purchases. In one implementation, the MCB-Platform server may update a
22 value card table record (e.g., 5119u) to deallocate the user 1102c from the gift card once
23 it has been liquidated and an equivalent value has been provided. In one
24 implementation, upon liquidation at 1189, the user may be sent the updated gift card
25 balance message 1192 notifying that the gift card has been liquidated with no balance

1 remaining in the gift card. In a further implementation, the user may also be notified of
 2 the deposit of the equivalent amount in a user designated account, statement credit,
 3 cash, and/or the like.

4 **[00271]** In one implementation, the liquidated gift card may be allocated to
 5 another user. In such a situation, the MCB-Platform server may send a charge request
 6 1190, corresponding to the user 1102c's (liquidated) gift card on behalf of the new user
 7 (and not user 1102c) to the gift card issuer 1108c.

```

8   POST /chargerequest.php HTTP/1.1
9   Host: www.giftcardissuer.com/charge
10  Content-Type: Application/XML
11  Content-Length: 484
12  <?XML version = "1.0" encoding = "UTF-8"?>
13  <charge_request>
14      <gift_card_ID>675678987654</gift_card_ID>
15      <user_ID>theoriginalowner@gmail.com</user_ID>
16      <checkout_request>
17          <checkout_ID>4NFU4RG94</checkout_ID>
18          <timestamp>2011-02-22 15:22:43</timestamp>
19          <purchase_detail>
20              <purchase_amount>100</purchase_amount>
21              <num_products>5</num_products>
22              <product_ID>AE95049324</product_ID>
23              <product_ID>MD09808755</product_ID>
24              <product_ID>OC12345764</product_ID>
25              <product_ID>KE76549043</product_ID>
26              <product_ID>SP27674509</product_ID>
27          </purchase_detail>
28          <PoS_client_detail>
29              <client_IP>192.168.23.126</client_IP>
30              <client_type>smartphone</client_type>
31              <client_model>HTC Hero</client_model>
32              <OS>Android 2.2</OS>
33              <app_installed_flag>true</app_installed_flag>
34          </PoS_client_detail>
35      </checkout_request>
36  </charge_request>
37

```

38 **[00272]**

39 **[00273]** The gift card issuer may receive the charge request. In one
 40 implementation, the gift card issuer may look up the balance in the gift card to ensure
 41 that the balance in the gift card covers the purchase amount. In a further
 42 implementation, the issuer may confirm that the user ID associated with the gift card

1 number matches the user ID to whom the gift card was initially authorized. Upon
2 making payment request validation, the gift card issuer may authorize the charge
3 request and send an authorization message 1191 to the MCB-Platform server. An
4 example authorization message 1191, substantially in the form of a HTTP(S) POST
5 message including XML-formatted data, is provided below:

```
6 POST /chargeauthorize.php HTTP/1.1
7 Host: www.visa.com/uve/auth
8 Content-Type: Application/XML
9 Content-Length: 484
10 <?XML version = "1.0" encoding = "UTF-8"?>
11 <auth>
12     <gift_card_ID>675678987654</gift_card_ID>
13     <user_ID>theoriginalowner@gmail.com</user_ID>
14     <status>approved</status>
15     <balance>0</balance>
16     <checkout_ID>4NFU4RG94</checkout_ID>
17     <timestamp>2011-02-22 15:22:43</timestamp>
18 </auth>
19 [00274]
```

20 **[00275]** Once the purchase is authorized, the gift card balance may be exhausted or
21 decremented. In one implementation, the MCB-Platform server may update the gift
22 card balance at 1193 (e.g., update value card table record 5119u) to indicate the new
23 balance.

24 **[00276]** FIGURES 12C-D show logic flow diagrams illustrating closed loop gift card
25 value exchange second embodiment of the MCB-Platform. The open/closed loop gift
26 card value exchange may begin at 1263. At 1264, client 1 1201b may send instructions to
27 transfer value from source gift card to a destination gift card. The instructions may
28 identify the source gift card and the destination gift card. The instructions may be
29 received by MCB-Platform server 1265. The MCB-Platform server may parse the
30 instructions to obtain identifiers for the gift cards at 1265. The MCB-Platform server
31 may further identify the issuers of the gift cards at 1266, and obtain balance in the

1 source gift card account at 1267. At 1268, the MCB-Platform server may determine
2 whether the source gift card is an open or a closed loop gift card. If the source gift card is
3 a closed loop gift card, the MCB-Platform server may, at 1276, query one or more
4 databases and/or tables to look up a target gift card exchange request (e.g., from client 2
5 1203b) or a target gift card that available in the MCB-Platform pool 1203b for exchange.
6 If a target gift card is determined to be available at 1279 based on query results obtained
7 at 1278, the MCB-Platform server may, in one implementation, request confirmation
8 from client 2/pool that the target gift card may be used for exchange. In another
9 implementation, the exchange may be preapproved. In one implementation, at 1280,
10 the client2/pool may select and/or provide a target gift card (e.g., gift card number) to
11 the MCB-Platform server. The MCB-Platform server may obtain the target gift card
12 information at 1281 and may determine the exchange rate and equivalent value (e.g.,
13 1282-386) in a manner similar to that described with respect to FIGURES 12A-B. At
14 1287, the MCB-Platform server may send a request to client 1 asking to confirm that the
15 equivalent value and/or exchange rate is acceptable. At 1288, client 1 may confirm the
16 exchange. At 1289, upon receiving the confirmation, the MCB-Platform may deallocate
17 (or debit) the value of the source gift card such that the balance of the source gift card is
18 not available to the user. In one embodiment, the original value of the gift card will be
19 set to an allocated value card that is associated with the original card. In essence this
20 will be the value used by MCB-Platform participants. If a card is to have a value
21 deallocated, this value card will have the appropriate amount deducted from it based on
22 value exchange calculations, while the amount on the original card is as of yet
23 unaffected. For example, the transfer request data structure 282 shows the underlying
24 card value of 200 points is unaffected while participating user 1 will see only 100 points

1 of that value and participating user 2 will see 50 points and the MCB-Platform in this
2 example has allocated 50 points to itself for various transaction fees. As such, the MCB-
3 Platform may generate a new value card record in value card table 5119u having the
4 original identifier of the card, the original owner ID of the card, the target owner ID, the
5 tracking equivalent amount that deducts the appropriate value equivalent off of the
6 original amount and the transferred amount. This tracking equivalent amount is what
7 will be visible to the original owner. The target user will see the transferred value field
8 associated with the gift card. Similarly, credit/allocate may affect the field values of the
9 value card record appropriately.

10 **[00277]** At 1290, the MCB-Platform server may deallocate the value of the target
11 gift card such that the value of the target gift card is not available for the target gift card
12 for anyone else. At 1291, the destination gift card is allocated the equivalent value. In
13 one implementation, the destination gift card is linked to the target gift card. When the
14 user makes a purchase using his or her destination gift card, a charge request is sent to
15 the issuer of the target gift card to charge the value of the purchase (up to the equivalent
16 amount) to the target gift card. As such, the allocation and deallocation are ledger
17 entries made to track the exchange of the gift cards between users without actually
18 moving funds from one account to another. In some implementations, the payment
19 gateway may assist in the routing of the charge requests to the appropriate issuer or
20 issuer bank. At 1292, the MCB-Platform server may update the ledger entry balances for
21 the source, destination and target gift card, concluding the process at 1275.

22 **[00278]** Referring to FIGURE 12C, when there is no target gift card available for a
23 swap at 1279, or when the source gift card is determined to be an open loop card at

1 1268, the MCB-Platform server may determine the equivalent value of the source gift
2 card at 1269. In one implementation, the equivalent value may be 50% of the source gift
3 card value. In another implementation, the equivalent value may be determined using
4 similar method outlined in 1282-386 (FIGURE 12D). At 1270, the MCB-Platform server
5 may provide the equivalent value to client 1 and request acceptance of the transfer. At
6 1271, the user may input acceptance of the transfer and the client may provide the
7 acceptance message to the MCB-Platform server. In response, the MCB-Platform server
8 may deallocate the value of the source gift card at 1272, and may allocate the equivalent
9 value to an account at 1273. In one implementation, the user may select an account
10 where the equivalent value may be deposited. In an alternate implementation, the
11 equivalent amount be converted into MCB-Platform currency and the user's MCB-
12 Platform currency balance may be updated. At 1274, the value of the source gift card
13 account may be allocated to a MCB-Platform account or a MCB-Platform pool,
14 concluding the gift card exchange process. Examples on how to allocate and deallocate
15 are discussed with respect to processes e.g., 1289, 1291.

16 **[00279]** FIGURE 13 shows a data flow diagram illustrating source/destination
17 value exchange embodiment of the MCB-Platform. A user 1302 may launch a MCB-
18 Platform application or access a MCB-Platform web page and input login credentials
19 into a client 1304 at 1310. The client may generate and send an authentication request
20 1312 to the MCB-Platform server 1306. An example authentication request 1312,
21 substantially in the form of a HTTP(S) POST message including XML-formatted data, is
22 provided below:

```
23 POST /authreqyest.php HTTP/1.1  
24 Host: www.visa.com/uve  
25 Content-Type: Application/XML
```

```

1 Content-Length: 1384
2 <?XML version = "1.0" encoding = "UTF-8"?>
3 <auth_request>
4     <request_ID>45DSKFSWGG9</request_ID>
5     <timestamp>yyyy-mm-dd hh:mm:ss</timestamp>
6     <user_ID>JDoe@gmail.com</user_ID>
7     <password>kingoftheworlds1982</password>
8     <wallet_account_ID>6785456789763434</wallet_account_ID>
9     <client_details>
10        <client_IP>192.168.23.122</client_IP>
11        <client_type>smartphone</client_type>
12        <client_model>HTC Hero</client_model>
13        <OS>Android 2.2</OS>
14        <app_installed_flag>true</app_installed_flag>
15    </client_details>
16 </auth_request>
17

```

18 **[00280]** The MCB-Platform server may extract details from the authentication
19 request 1312 (e.g., XML data) to validate the authentication request. If the
20 authentication request cannot be verified, the user may be asked to re-enter login
21 credentials. The MCB-Platform server may identify all the loyalty programs that the user
22 is currently enrolled in at 1314. The MCB-Platform server may also identify the program
23 providers of the enrolled programs. In one implementation, the MCB-Platform may
24 query its user database to obtain a list of the user's enrolled programs. For example, the
25 MCB-Platform server may issue PHP/SQL commands to query a database table for
26 enrolled program data associated with the user. An example query, substantially in the
27 form of PHP/SQL commands, is provided below:

```

28 <?PHP
29 header('Content-Type: text/plain');
30 mysql_connect("254.93.179.112", $DBserver, $password); // access database server
31 mysql_select_db("MCB-Platform_DB.SQL"); // select database table to search
32 //create query
33 $query = "SELECT enrolled_program_list program_issuer FROM UserTable WHERE user
34 LIKE '%" $user_id";
35 $result = mysql_query($query); // perform the search query
36 mysql_close("MCB-Platform_DB.SQL"); // close database access
37 ?>

```

38 **[00281]**

39 **[00282]** In one implementation, the MCB-Platform server may query an issuer
40 database to obtain issuer balance/exchange rate request template to process the

1 exchange. The issuer template may include instructions, data, login URL, login API call
 2 template, rules and restrictions file, exchange rate file and/or the like for facilitating
 3 data exchange between the MCB-Platform server and the program issuer server. An
 4 example PHP/SQL command listing, illustrating substantive aspects of querying the
 5 database, is provided below:

```

6  <?PHP
7  header('Content-Type: text/plain');
8  mysql_connect("254.93.179.112",$DBserver,$password); // access database server
9  mysql_select_db("MCB-Platform.SQL"); // select database table to search
10 //create query
11 $query = "SELECT template FROM ProgramTable WHERE issuer LIKE '%"
12 $program_issuer";
13 $result = mysql_query($query); // perform the search query
14 mysql_close("MCB-Platform.SQL"); // close database access
15 ?>
16

```

17 **[00283]** In one implementation, the MCB-Platform may create and send a current
 18 points/currency balance and exchange rate request 1316 to the identified program
 19 provider servers 1308. The request 1316 may be in the form of an API/web service call in
 20 some implementations. The program provider servers may respond to the MCB-
 21 Platform server's request with the requested points/currency balance. For example, the
 22 program provider server may provide an HTTP(S) POST message, e.g., 1318, similar to
 23 the example below:

```

24 POST /balanceinfo.php HTTP/1.1
25 Host: www.uve.com
26 Content-Type: Application/XML
27 Content-Length: 1306
28 <?XML version = "1.0" encoding = "UTF-8"?>
29 <balance_notification>
30     <request_ID>4NFU5GG94</request_ID>
31     <timestamp>20xx-02-22 15:22:43</timestamp>
32     <member_ID>5645789643452367</member_ID>
33     <balance>100</balance>
34     <rate>10</rate>
35     <base_currency>dollars</base_currency>
36 </balance_notification>

```

37 **[00284]**

38 **[00285]** The MCB-Platform server may then provide program points/currency

1 balance message 1320 to the user's client 1304. In one implementation, the client may
 2 display the contents of the message 1320 to the user. The user may initiate a
 3 points/currency exchange transaction at 1322. In one implementation, the user may
 4 select a source program to initiate an exchange transaction. The client may generate and
 5 send a points/currency exchange request 1324 to the MCB-Platform server. In one
 6 implementation, the request 1324 may include user ID, source program ID, and/or the
 7 like. An example exchange request 1312, substantially in the form of a HTTP(S) POST
 8 request including XML-formatted data, is provided below:

```

9  POST /exchangerequest.php HTTP/1.1
10  Host: www.visa.com/uve
11  Content-Type: Application/XML
12  Content-Length: 1384
13  <?XML version = "1.0" encoding = "UTF-8"?>
14  <exchange_request>
15      <request_ID>45DSKFTGGG9</request_ID>
16      <timestamp>yyyy-mm-dd hh:mm:ss</timestamp>
17      <user_ID>JDoe@gmail.com</user_ID>
18      <source_details>
19          <member_ID>4444566897978766</member_ID>
20          <program_ID>MER675656</program_ID>
21          <issuer_ID>apple</issuer_ID>
22          <card_value>100</card_value>
23          <currency>usd</currency>
24      </source_details>
25      <client_details>
26          <client_IP>192.168.23.122</client_IP>
27          <client_type>smartphone</client_type>
28          <client_model>HTC Hero</client_model>
29          <OS>Android 2.2</OS>
30          <app_installed_flag>true</app_installed_flag>
31      </client_details>
32  </exchange_request>

```

33 **[00286]**

34 **[00287]** The MCB-Platform server may receive the exchange request and parse the
 35 request to obtain details (e.g., XML data). For example, the MCB-Platform server may
 36 identify the source program, and using the user ID, identify destination programs to
 37 which the source program points/currencies could be transferred. At 1326, the MCB-
 38 Platform server may query one or more databases and/or tables to determine rules and

1 restrictions for the source program. Further, in some implementations, the MCB-
2 Platform server may examine the rules and restrictions to determine potential
3 destinations programs that are available for exchange, unavailable for exchange and
4 preferred for exchange. FIGURES 5A-B provide additional detail on these
5 determinations. Upon identifying the potential destination programs, the MCB-
6 Platform server may send the client a request 1328 to select a destination program. The
7 request 1328 may include the list of the potential destination programs and indications
8 of whether they are unavailable, available or preferred destination program. For
9 example, the request 1328 may include XML formatted data similar to the example
10 below:

```
11 <selectdestination_request>
12     <request_ID>45DSJKTGGG9</request_ID>
13     <timestamp>yyyy-mm-dd hh:mm:ss</timestamp>
14     <source_details>
15         <member_ID>4444566897978766</member_ID>
16         <program_ID>MER675656</program_ID>
17         <issuer_ID>apple</issuer_ID>
18         <card_value>100</card_value>
19         <currency>usd</currency>
20     </source_details>
21     <destination_details>
22         <destination1>
23             <program_ID>MER567855</program_ID>
24             <issuer_ID>Hilton</issuer_ID>
25             <designation>available</designation>
26         </destination1>
27         <destination2>
28             <program_ID>MER598755</program_ID>
29             <issuer_ID>BestBuy</issuer_ID>
30             <designation>available</designation>
31         </destination2>
32         <destination3>
33             <program_ID>MER232855</program_ID>
34             <issuer_ID>Microsoft</issuer_ID>
35             <designation>unavailable</designation>
36         </destination3>
37         <destination4>
38             <program_ID>MER765555</program_ID>
39             <issuer_ID>Macworld</issuer_ID>
40             <designation>preferred</designation>
41         </destination4>
42         .
43         .
44         .
45 </selectdestination_request>
```

1 **[00288]**

2 **[00289]** The potential destination programs and their corresponding indications
3 may be displayed by the client. The client may specifically grey out unavailable
4 destination programs to indicate that the unavailable program cannot be selected by the
5 user for the exchange transaction. Further the client may highlight the preferred options
6 to draw the user's attention to the most optimal option for the exchange transaction. In
7 one implementations, potential destination programs that are neither unavailable nor
8 preferred may be displayed normally and may be available to the user for selection even
9 though the option may not be the most optimal.

10 **[00290]** At 1330 the user may select an available or preferred destination program.
11 Upon selection of the source program, the client may display an option for the user to
12 select or input an amount of the source program points/currency to exchange. In some
13 implementations, a default amount (e.g., available balance) may be pre-populated. The
14 client may package the user's input of the selected destination program and the amount
15 of the source program points/currency into an equivalent value request 1332 and send
16 the request to the MCB-Platform server. In one implementation, the equivalent value
17 request 1332 may include user ID, destination program ID, source program ID, source
18 program amount, and/or the like. The MCB-Platform server may receive the request
19 1332 and parse the request to identify the source program, destination program as well
20 as the amount to be exchanged. The MCB-Platform server may query one or more
21 databases and/or tables to determine the exchange rate between source program and
22 the destination program. The MCB-Platform server may then utilize the exchange rate
23 to calculate the equivalent value in destination points/currency at 1334. The MCB-

1 Platform server may send a request 1336 to the client to confirm exchange for the
2 equivalent destination points/currency. In one implementation, the request 1336 may
3 include user ID, source program ID, destination program ID, equivalent value, exchange
4 rate, validity time period, and/or the like. The user may view the equivalent value and
5 exchange rate and may agree to proceed with the exchange transaction at 1338. The
6 confirmation message 1340 may then be generated by the client and sent to the MCB-
7 Platform server. Upon receiving confirmation from the user, the MCB-Platform server
8 may send a payment request 1342 to the program provider to request payment for the
9 exchange transaction. In one implementation, the payment request 1342 may include
10 provider ID, source program ID, destination program ID, user ID, exchange rate,
11 equivalent value, points/currency amount for exchange, bill amount and/or the like. An
12 example payment request 1342, substantially in the form of a HTTP(S) POST request
13 including XML-formatted data, is provided below:

```
14 POST /paymentrequest.php HTTP/1.1
15 Host: www.programprovider.com/miles
16 Content-Type: Application/XML
17 Content-Length: 1384
18 <?XML version = "1.0" encoding = "UTF-8"?>
19 <payment_request>
20     <request_ID>45KGKFTGGG9</request_ID>
21     <timestamp>yyyy-mm-dd hh:mm:ss</timestamp>
22     <member_ID>4444566897978766</member_ID>
23     <program_ID>MER675656</program_ID>
24     <exchange_rate>10</exchange_rate>
25     <exchange_rate_startdate>yyyy-mm-dd</exchange_rate_startdate>
26     <exchange_rate_enddate>yyyy-mm-dd</exchange_rate_enddate>
27     <destination_program>BestBuy</destination_program>
28     <currency_amt>10000</currency_amt>
29     <currency>miles</currency>
30     <equivalent_amt>225</equivalent_amt>
31     <bill_owed>250</bill_owed>
32 </payment_request>
```

33 **[00291]**

34 **[00292]** The program provider may authorize payment and may send a payment
35 confirmation message 1344 to the MCB-Platform server. The payment confirmation

1 message may include provider ID, source program ID, destination program ID, user ID,
2 exchange rate, equivalent value, points/currency amount for exchange, payment ID, bill
3 amount and/or the like. In one implementation, both the source and destination
4 program providers may be billed for the services provided. Upon receiving the payment
5 confirmation message 1344, the MCB-Platform server may execute the exchange
6 transaction at 1346. In one implementation, executing the exchange transaction may
7 include decrementing the user's source program points/currency and incrementing the
8 destination program points/currency. Upon execution of the exchange transaction, the
9 source/destination gift card balances may be updated and the updated balance
10 information may be provided to the program providers via a balance message 1348.

11 **[00293]** FIGURES 14A and 14B show logic flow diagrams illustrating
12 source/destination value exchange component embodiment of the MCB-Platform.
13 Starting at 1406, a user may launch a MCB-Platform application on a client 1401 and
14 may provide login credentials at 1408. The login credentials are sent by the client to the
15 MCB-Platform server 1402. The MCB-Platform server may receive the login credentials
16 at 1410 and may authenticate the user. Once the user is authenticated, at 1412 the MCB-
17 Platform server may query one or more user databases and/or tables using for example
18 the user ID to identify loyalty programs in which the user is currently enrolled. At 1414,
19 the MCB-Platform server may communicate with the enrolled programs to ascertain
20 current points/balance and any exchange rate that they may have established. In one
21 implementation, the MCB-Platform server may communicate with the program service
22 provider servers 1403 using API/web service calls. The program provider servers may
23 receive the request from the MCB-Platform server, and at 1414, may validate that the
24 request is authentic. For example, the program provider may check the MCB-Platform

1 server credentials, user ID and/or the like to validate the request. At 1417, the program
2 service provider server may use the user ID in the received request to query their
3 databases and/or tables to determine the user's current points/currency balance. In a
4 further implementation, the program service provider may also look up the exchange
5 rate for the source program points/currency. At 1418, the program service providers
6 may return the obtained balance and exchange rate information to the MCB-Platform
7 server. At 1420, the MCB-Platform server may obtain the balance and exchange rate
8 information from each enrolled program. The MCB-Platform may also provide the
9 balance information, and in some implementations, the exchange rate for each of the
10 enrolled program to the client. In some implementations, the balance information may
11 be directly communicated to the client by the program service provider. Upon receiving
12 the points/currency balance, the client may display the balance at 1422 and inquire if
13 the user wishes to select a source program for an exchange transaction.

14 **[00294]** At 1424, the user may select a source currency/point program to initiate an
15 exchange transaction. The client may communicate the selected source program to the
16 MCB-Platform server which may receive the selection at 1426. At 1428, the MCB-
17 Platform server may parse the message received and may query the rules and
18 restrictions database to determine any rules and restrictions associated with the source
19 program.

20 **[00295]** In some implementations, each program may have rules and restrictions
21 associated therewith that allow certain exchanges to proceed while forbidding others.
22 Example rules and restrictions include: a minimum redemption group (e.g., redeem in
23 groups of 1400 miles), minimum redemption amount (e.g., users with 10,000 miles or

1 more can redeem), non-refundable exchange, exchange amount limit, number of
2 transactions per period limit, and/or the like.

3 **[00296]** At 1430, the MCB-Platform server may obtain the associated rules and
4 restrictions file and may evaluate each of the other enrolled programs against the source
5 program rules and restrictions. Referring to FIGURE 14B, at 1432, any program not
6 meeting the rules and restrictions may be identified. At 1434, one or more programs
7 that do not meet the source program rules and restrictions may be identified and
8 marked as unavailable for exchange, and the processing moves to 1440. If at 1432, all
9 programs are found to meet source program rules and restrictions, the processing
10 moves to 1436. At 1436, the MCB-Platform server may evaluate the exchange rates of
11 the programs that meet the rules and restrictions, and at 1438, based on the evaluation,
12 the MCB-Platform may determine and identify a preferred program for the exchange
13 transaction. In one implementation, for example, a preferred program may be a
14 program that has the most favorable exchange rate with the source program. With
15 regard to identifying, highlighting, marking, etc., value exchange program at e.g., 1434,
16 1436, 1438, 1440, the MCB-Platform may make entries as being preferred, not allowed
17 or restricted, allowed, etc., by updating a programs record 5119k appropriately. For
18 example, program record entry for, e.g., Delta Skymiles 850b of FIGURE 8L may appear
19 as follows:

```
20 <program1>
21     <program_name>Mileage Plus United</program_name>
22     <status>restricted</status>
23     <exchange_rate>NA</exchange_rate>
24 </program1>
25 <program2>
26     <program_name>Hilton HHnors Point</program_name>
27     <status>Preferred</status>
28     <exchange_rate>2:1</exchange_rate>
29 </program2>
30 <program3>
```

```
1         <program_name>BestBuy Rewards</program_name>
2         <status>allowed</status>
3         <exchange_rate>10:1</exchange_rate>
4     </program3>
5     .
6     .
7     .
```

8 **[00297]** In other implementations, the preferred program may have additional
9 rewards/points that may be obtained after the completion of the exchange. In yet other
10 implementations, preferred programs may be selected based upon other factors such as
11 acceptance, transaction history, and/or the like. Exchange rate evaluation and preferred
12 program determination are discussed in detail with respect to FIGURES 6A-B.

13 **[00298]** At 1440, the MCB-Platform server may provide to the client the identified
14 programs and indications whether each program is unavailable, available or preferred
15 for exchange with the source program. The client may receive the identified program
16 information and may display the unavailable program as an unselectable option at 1442.
17 In one implementation, the unavailable program may be grayed out to clearly identify
18 that the source program rules and restrictions forbid conversion of the source program
19 to the unavailable program. At 1444, the client may display the available programs as
20 options that can be selected. In a further implementation, the client may highlight the
21 preferred program so as to clearly identify that the highlighted program is the preferred
22 program to which the source program points/currency should be converted to.

23 **[00299]** The user may select a destination program from the available list of
24 programs and may input an amount of the source/currency points at 1446. The client
25 receives the input and sends the information to the MCB-Platform server which receives
26 the selected destination program and the amount of the source program
27 points/currency for exchange at 1448. At 1450, the MCB-Platform may determine
28 equivalent amount of destination currency/points for the selected amount of source

1 program currency points. In one implementation, the equivalent amount may be
2 calculated based on the exchange rate between the source and destination program
3 points/currency. In some implementations, the exchange rate of each program may be
4 with respect to a base currency/unit such as the MCB-Platform point, from which the
5 exchange rate between the two program points/currency may be determined. At 1452,
6 the MCB-Platform may provide the equivalent destination currency/points to the client
7 which may display the information at 1454. The client may also display controls for the
8 user to adjust or change the transaction. For example, the user may go back and change
9 the destination program or may adjust the source program points/currency amount. At
10 1456, the user may confirm the exchange, adjust or cancel the exchange transaction. At
11 1458, if the user does not confirm the transaction, the client may inquire if the user may
12 want to adjust the transaction. At 1472, if the user wants to adjust the transaction, the
13 process may move to 1446, where the user is provided an option to select another
14 destination program or adjust the amount for conversion. If at 1472, the user does not
15 wish to adjust the transaction, the client may notify the MCB-Platform server to cancel
16 the exchange transaction at 1474. The exchange transaction may then come to its
17 conclusion at 1468. On the other hand, if the user confirms the exchange at 1458, the
18 client sends a confirmation message 1459 to the MCB-Platform server. At 1460, the
19 MCB-Platform server may request payment from the program provider for exchange of
20 the amount of source points/currency. Referring to FIGURE 14A, the request may be
21 received by the program service providers at 1462. At 1463, the program service
22 providers may confirm payment or acknowledge the exchange transaction. Referring To
23 FIGURE 14B, the payment confirmation/acknowledgement may be received by the
24 MCB-Platform server at 1464. The MCB-Platform server may then update the

1 source/destination points/currency balance in one or more databases and/or tables. At
2 1469, the MCB-Platform server may provide the updated points/currency balances
3 and/or confirmation of the exchange transaction to the client and the program
4 providers. The client may receive the updated balances and confirmation and may
5 display a transaction summary at 1470. Referring to FIGURE 14A, the program
6 providers may receive the updated balance information and confirmation of the
7 transaction and may update their own records at 1466. The transaction processing may
8 then conclude at 1468.

9 **[00300]** FIGURES 15A and 15B show logic flow diagrams illustrating equivalent
10 value determination component embodiment of the MCB-Platform. Starting at 1552, the
11 component 1501 may receive as input a user request to exchange a source program
12 points/currency at 1502. At 1504, a determination may be made as to whether the
13 source program provider is a MCB-Platform partner. In one implementation, a MCB-
14 Platform partner is a program provider having an agreement with and enrolled in the
15 MCB-Platform. If the source program provider is a MCB-Platform partner, the user's
16 current program points/currency and exchange rate may be obtained from the provider
17 at 1506. At 1508, rules and restrictions tables and/or databases may be queried using the
18 provider's program ID to obtain rules and restrictions for the source program. At 1510,
19 programs that are restricted from participating in the requested currency/points
20 exchange are identified. A determination may be made at 1512 whether there are any
21 unrestricted programs. If there are any unrestricted programs, such programs are
22 identified at 1514. In one implementation, the exchange rates for each of the identified
23 unrestricted programs points/currency are obtained and compared at 1516b. For
24 example, as shown in 1516b, the exchange rate (e.g., points/dollars) of source,

1 destination 1, destination 2 and destination 3 are obtained. Using the same common
2 denominator (e.g., dollars), the exchange rate of the source with respect to each of the
3 destination programs may be calculated. By comparing the calculated exchange rate
4 ratios, the most favorable ratio may be selected. In the example shown, destination 3
5 has the most favorable ratio, and as such destination 3 may be selected as the preferred
6 program at 1518. In another implementation, the component 1501 may obtain exchange
7 rate data relating to the source/destination programs for at least the last three
8 consecutive time periods. As shown in 1516a example, the exchange rate trend for each
9 of the destination programs may be evaluated and a preferred exchange rate determined
10 based on the trend analysis. In the example shown, destination 2 may be determined to
11 be a preferred program for exchange. Upon determining a preferred program at 1518,
12 the user may be requested to select a destination program from the list of preferred and
13 other unrestricted programs at 1520. In one implementation, the restricted programs
14 may also be displayed along with the preferred and/or unrestricted destination
15 programs. In a further implementation, the restricted programs may be de-highlighted
16 or grayed out to indicate that these programs may not be selected as destination
17 programs. In one implementation, the preferred program(s) may be highlighted to
18 clearly distinguish it from other options. In some implementations, the highlighting and
19 de-highlighting may be mandated by exchange rate analyses (e.g., 1516a, 1516b). In a
20 further implementation, one or more destination programs may be given preferential
21 treatment based on user preferences. For example, the user may specify a ranking of his
22 or her rewards programs. In such a case, the MCB-Platform server may present as
23 preferred a destination program that the user prefers provided that the destination
24 program is not restricted by the rules and conditions. In yet another implementation,

1 bilateral relationship or affiliation between the source and a destination program may
2 be taken into account while determining a preferred destination program.

3 **[00301]** The user selection of a destination program and an amount of the
4 points/currency may be obtained at 1522. In one implementation, a determination may
5 be made whether the user selected amount meets the source program rules/restrictions
6 at 1524. For example, the source program rules and restrictions may require the source
7 amount to be selected in groups of 500. As another example, a user may have to have
8 select a minimum amount of points/currency or may not select more than a maximum
9 amount of points/currency. If the user selected amount does not meet the rules and
10 restrictions, the amount may be automatically adjusted at 530 by rounding up or down.
11 If the user selected amount meets the rules and restrictions, or once the user selected
12 amount has been adjusted to meet the rules and restrictions, transaction fees and/or
13 payment for the points/currency may be billed to (or deducted from) the
14 source/destination program providers at 1526. At 1528, the user may be provided the
15 equivalent destination points/currency, completing the transaction at 1550.

16 **[00302]** In one implementation, when the source program provider is not a MCB-
17 Platform partner (as determined at 1504) or when there are no unrestricted programs
18 (as determined at 1512), referring to FIGURE 15B, the exchange may be limited to MCB-
19 Platform points/currency and/or cash at 1532. At 1534, the component may examine
20 transaction history to assess the demand for the source program points/currency. In one
21 implementation, a method similar to the risk exposure thresholds and weights shown in
22 Table 1 may be utilized to determine demand or risk exposure. At 1536, the exchange
23 rate may be set based on the weighted demand/risk exposure. At 1538, the user may be

1 provided an option to select cash and/or MCB-Platform points as a destination
2 program. At 1540, the component may obtain the user selected destination program and
3 an amount of source points/currency for conversion. At 1542, a determination may be
4 made whether to adjust the exchange rates based on the amount. For example, the
5 amount selected by the user may be too high, increasing the risk exposure and therefore
6 may require adjustment of the exchange rate. If an adjustment is required, at 1544, the
7 exchange rate may be adjusted and the process moves on to 1546. At 1546, equivalent
8 destination program amount may be determined using the original or adjusted
9 exchange rates. At 1548, the equivalent amount may be provided to the user for
10 confirmation. In some implementations a transaction fee may be levied for the exchange
11 transaction. The process may conclude at 1550.

12 **[00303]** FIGURE 16 shows a logic flow diagram illustrating cross-ecosystem
13 exchange component embodiment of the MCB-Platform. Exemplary aspects of
14 transforming value equivalent exchange instructions into cross-ecosystem currency
15 exchanges in some embodiments of the MCB-Platform are discussed. In some
16 implementations, a universal value exchange controller may obtain one or more cross-
17 ecosystem currency exchange instructions, e.g., 1604. For example, such instructions
18 may specify currency source details and currency destination details such as those
19 discussed above. The universal value exchange controller may parse the obtained
20 instructions, and determine the identities of the ecosystems acting as sources and
21 destinations of the currencies, e.g., 1606. The universal value exchange controller may
22 utilize the identities of the source and destination ecosystems to determine the currency
23 types associated with each of the source and destination currency ecosystems, e.g., 1608.
24 Using the currency types, the universal value exchange controller may determine an

1 exchange rate of each of the source and destination currencies relative to a standard
2 currency, e.g., 1610. For example, the universal value exchange controller may look up
3 the currency exchange rates of the currency types of the currency sources in a relational
4 database using a hypertext preprocessor (PHP) script utilizing Structured Query
5 Language (SQL) commands. In some implementations, the universal value exchange
6 controller may similarly determine the currency exchange rates of the currency types of
7 the currency destinations, e.g., 1618. In some implementations, the universal value
8 exchange controller may parse the cross-ecosystem currency exchange instructions, and
9 obtain account information (e.g., account name, account number, routing number,
10 password, security codes, CVV number, etc.) for the source currencies, e.g., 1616. For
11 example, the universal value exchange controller may utilize such information to obtain
12 access to the purchasing power retained in the currency sources. In some
13 implementations, the universal value exchange controller may parse the cross-
14 ecosystem currency exchange instructions, and obtain account information for the
15 destination currencies, e.g., 1614. For example, the universal value exchange controller
16 may utilize such information to obtain access to the currency destinations for depositing
17 purchasing power into the currency destinations.

18 **[00304]** In some implementations, the universal value exchange controller may
19 also determine whether there are any restrictions and/or conditions at each of the
20 sources of the currencies, as well as the destinations of the currencies. For example, the
21 universal value exchange controller may query a database to obtain the restrictions
22 and/or conditions for the sources and/or destinations. In some implementations, the
23 universal value exchange controller may generate, e.g., 1620, a currency exchange flow
24 path based on the restrictions and/or conditions at the currency sources and/or

1 destinations. Upon generating the currency exchange flow path, the universal value
2 exchange controller may, in some implementations, if an API is available, e.g., 1624,
3 initiate currency exchange along the generated currency exchange flow path, for
4 example, by providing request messages to the components in the currency exchange
5 flow path to provide and/or accept currency value, based on the generated currency
6 exchange flow path. The universal value exchange controller may monitor the currency
7 exchange flow among the components in the currency exchange flow path until the
8 currency exchange is complete, e.g., 1628-1630. Alternatively if an API is not available,
9 e.g., 1624, the MCB-Platform controller may deallocate a specified value from the source
10 account e.g., 1638 and allocate an equivalent value calculated using the valuation rate to
11 the destination account, e.g., 1640. Upon completing the currency withdrawal and/or
12 deposits into each of the currency accounts involved in the cross-ecosystem currency
13 exchange, the universal value exchange controller may provide notifications, e.g., 1632,
14 for the users of the universal value exchange controller notifying them of completion of
15 the requested cross-ecosystem currency transaction. In some implementations, the
16 universal value exchange controller may determine whether there are more cross-
17 ecosystem currency exchange instructions remaining to be processed (e.g., 1634, option
18 “Y”), and perform the cross-ecosystem currency exchanges until all the cross-ecosystem
19 currency exchange instructions have been processed (e.g., 1634, option “N”).

20 **[00305]** FIGURES 17A-D show screenshot diagrams illustrating exchange mode
21 embodiments of the MCB-Platform. In some implementations, the exchange mode UIs
22 may include various options for user selections. Referring to FIGURE 17A for example,
23 the left UI shows exchange 1701, today's exchange rate 1702, manage my cards 1703, my
24 MCB-Platform points 1704 and settings 1705 for user selection. Each of the options are

1 discussed in further detail below.

2 **[00306]** When the exchange option 1701 is selected from the left UI, the exchange
3 UI (right) may be displayed. The exchange UI may display various options for selecting a
4 source currency. For example, a user may select the loyalty tab 1706a as a source
5 currency. When the loyalty tab is selected a loyalty panel 1706b may be displayed. As
6 shown, the loyalty panel may include a listing of loyalty cards or accounts. The user may
7 select one or more of these loyalty accounts as a source currency. Further for each
8 selected account, the user may view the total available points/currency as well as select
9 the amount of currency the user would like to exchange. Also shown in the right UI is a
10 value equivalent selection panel 1706c. The user may select any of the options as the
11 destination into which the loyalty currencies may be converted to. The back button
12 1706d allows the user to go back to the left UI, while the exchange button 1706e allows
13 the user to initiate the exchange.

14 **[00307]** Referring to FIGURE 17B, when the virtual games tab 1708a is selected,
15 the virtual games panel 1708b is displayed. As shown, a list of the user's virtual
16 currencies are populated. The user may select one or more of these virtual currencies as
17 source currencies and may specify for each currency the amount to be converted.
18 Referring to the right UI, the monetary tab 1710a is selected. The UI shows the monetary
19 panel 1710b and a list of monetary accounts. These accounts may be imported from the
20 user's electronic wallet. Alternately, these monetary accounts may be added by the user
21 to the MCB-Platform application/account. As shown, one or more of these monetary
22 accounts may be selected, and the user may specify for each selected account, the
23 amount to be converted (e.g., \$52). Referring to FIGURE 17C, when the MCB-Platform

1 points tab 1712a is selected, the MCB-Platform points panel 1712b may be displayed. As
2 shown, the UI may display the amount of points available (e.g., 5000) and allow the
3 user to select the amount of points to be converted (e.g., 2000 points). As shown in the
4 right UI, any of the options in the value equivalent panel may be selected. As shown the
5 BestBuy rewards points option is selected. The panel 1714 displays the user selected
6 source currencies (e.g., United Airline Miles and Hilton points selected at panel 1706b,
7 Farmville cash selected at 1708b, Discover *5678 account selected at 1710b and MCB-
8 Platform points selected at 1712b), as well as equivalent of the selected source currencies
9 in BestBuy rewards points. In one implementation, when a source currency cannot be
10 converted into a selected currency, the conversion of that currency is skipped, and the
11 rest of the currencies are converted. As shown in the right UI, the user may view the
12 value equivalents, and if it is acceptable to the user, the user may confirm exchange by
13 selecting the exchange button 1716. Referring to FIGURE 17D, once the exchange is
14 performed, a summary 1718 of the remaining points/currency balance in the programs
15 may be displayed. For example, as shown, the UI may display the currencies that were
16 converted 1718a-d along with the remaining balance. In some other implementations,
17 the display 1718 may show the amount of currencies converted and the effective
18 exchange rate.

19 **[00308]** FIGURE 17E shows screenshot diagrams illustrating exchange rate mode
20 embodiment of the MCB-Platform. As shown in the left UI, the user may select view
21 today's exchange rate 1702. The right UI 1720 as shown displays a summary of the deals
22 or exchanges available in the display panel 1720a. In one implementation, these
23 exchange messages may be provided by the program providers to encourage
24 points/currency redemption. In other implementations, the messages may be provided

1 as an offer to gain points/currency by performing an online or offline activity.

2 **[00309]** FIGURES 17F-I show screenshot diagrams illustrating management mode
3 embodiment of the MCB-Platform. In one implementation, the selection of the option
4 manage my cards 1703 from the left UI may display the right UI. As shown, the right UI
5 displays various cards or accounts 1722a-i added to the MCB-Platform application or
6 account. In one implementation, the user may select one of the card accounts, e.g.,
7 1722i. Referring to FIGURE 17G, the left UI may be displayed to the user in response to
8 his or her selection of a card account 1722i. The user may have a number of options e.g.,
9 1726, 1728 and 1730 for selection. The about option 1726 may provide a brief description
10 about the program provider or card account. Selection of the enrollment option 1728
11 may lead to the right UI which may display enrollment status 1728a and enrollment
12 information such as name 1728b, email address 1728c, member ID 1728d, notification
13 setting 1728e, and/or the like. The enrollment information may be provided at the time
14 the card is added to the MCB-Platform account. As shown, the user may uncheck or
15 unselect the enrolled option 1728a to unenroll from the selected program 1722i. When
16 the usage preferences option 1730 is selected from the left UI, the left UI 1730a of
17 FIGURE 17H may be displayed. In this UI, the user may specify how the program
18 points/currency may be used. As shown, the user may select options 1730b-e. The user
19 may also add his or her own category for usage 630f. In some implementations, the user
20 may also specify priority or order of usage. Referring to the right UI of FIGURE 17H, the
21 user may select option 1724 to add a new card or program account. As shown in
22 FIGURE 17I, the user may enter information such as name 1732a, email 1732b, member
23 ID 1732c, username 1732d, password 1732e, short name 1732f, and/or the like to add a
24 program account to the MCB-Platform. The user may select the add card button 1734 to

1 add the program to the MCB-Platform or the cancel button 1736 to cancel.

2 **[00310]** FIGUREs 17J-K show screenshot diagrams illustrating MCB-Platform
3 point mode embodiment of the MCB-Platform. When the user selects my MCB-Platform
4 points option 1704 from the left UI, the right UI may be displayed. As shown, various
5 options e.g., 1738, 1744, 1745, 1746, etc., may be available. The user may select the about
6 option 1738 to read information about the MCB-Platform points. The enrollment option
7 1744 may be selected to view the left UI as shown in FIGURE 17K. The enrollment UI
8 shows the enrolled status 1744a, along with identifying information such as name 1744b,
9 email address 1744c, phone 1744d, payment device number, 1744e, security code 1744f,
10 billing zip code 1744g, and/or the like. The user may unenroll from the MCB-Platform
11 points program by unchecking the enrolled option 1744a. Referring back to FIGURE
12 17J, when the user selects the statement option 1745, a statement UI may be displayed.
13 The statement UI (not shown) may allow the user to select a time period and obtain a
14 statement summary of exchanges, MCB-Platform points balance, and/or the like. The
15 usage preferences option 1746 may be selected to view the right UI shown in FIGURE
16 17K. As shown, the user may select usage preferences for specific purchases 1746a-d for
17 the MCB-Platform points. The user may also add his or her own category 646e.

18 **[00311]** FIGUREs 17L-N show screenshot diagrams illustrating source/destination
19 exchange mode embodiment of the MCB-Platform. In one embodiment of the MCB-
20 Platform, a source UI 1750 may display a list of selectable options 1750a-g as possible
21 sources for an exchange. When the source 1750b (e.g., Delta Skymiles) is selected, the
22 destination UI 1752 may be displayed. The destination UI may display a list of possible
23 destination currencies 1752a-852g into which the source currency 1750b may be

1 converted. The destination UI may also highlight or de-highlight certain options to
2 indicate preference or restriction, For example, the destination UI shows "Mileage Plus
3 United" 1752a and "Cash" 1752e as grayed out indicating that these two options cannot
4 be selected as destination currencies. As a further example, "Hilton HHonors Point"
5 1752c option is highlighted (e.g., bold, large font) to indicate that the exchange of the
6 Delta Skymiles 1750b for Hilton HHonors Point is the most favorable or optimal
7 exchange. Other options 1752d-g that are selectable as destination programs may also be
8 shown, but without any emphasis, to indicate that these options are neither preferred
9 nor restricted.

10 **[00312]** Referring to FIGURE 17M, the user may select the preferred destination
11 program 1754c. The terms UI 1756(right) may then be displayed showing the details of
12 the exchange to be conducted. For example, the terms UI shows the exchange rate 1756a
13 that indicates that 2 Delta Skymiles is equivalent to 1 Hilton HHonors Points. The UI
14 may also display a slide control 1756b to allow the user to select the amount of Delta
15 Skymiles that the user wishes to convert. In a further implementation, the UI may also
16 display the equivalent Hilton HHonors Point 1756c that the selected amount of Delta
17 Skymiles would be converted to. Upon viewing the terms of the exchange, the user may
18 select the transfer button 1756d to initiate the exchange. In some implementations, the
19 user may also select the add to exchange cart button 1756e to add the exchange
20 transaction to cart and execute at a later time. The user may also set up other exchanges
21 and add those to the cart to simultaneously execute multiple exchanges.

22 **[00313]** Referring to FIGURE 17N, the user may select a destination program
23 1754d (e.g., BestBuy Rewards) that is not preferred, but is available for exchange. When

1 such an option is selected, the terms UI 1758 may display the terms of the exchange as
2 shown. In contrast to the preferred exchange shown in FIGURE 17M where the
3 exchange rate ratio was 2:1, in this case the exchange rate ratio 1758a may be worse
4 (e.g., 10:1). The user may specify the amount of the source program points to use via the
5 slider 1758b. The equivalent value destination program points may be displayed at
6 1758c. The user may execute the transfer by selecting the button 1758d or may postpone
7 it till later by selecting add to exchange cart button 1758e.

8 **MCB-Platform Centralized Personal Information Platform**

9 **[00314]** FIGURE 18 shows a block diagram illustrating example aspects of a
10 centralized personal information platform in some embodiments of the MCB-Platform.
11 In various scenarios, originators 1811 such as merchants 1811b, consumers 1811c,
12 account issuers, acquirers 1811a, and/or the like, desire to utilize information from
13 payment network systems for enabling various features for consumers. Such features
14 may include application services 1812 such as alerts 1812a, offers 1812c, money transfers
15 1812n, fraud detection 1812b, and/or the like. In some embodiments of the MCB-
16 Platform, such originators may request data to enable application services from a
17 common, secure, centralized information platform including a consolidated, cross-entity
18 profile-graph database 1801. For example, the originators may submit complex queries
19 to the MCB-Platform in a structure format, such as the example below. In this example,
20 the query includes a query to determine a location (e.g., of a user), determine the
21 weather associated with the location, perform analyses on the weather data, and provide
22 an exploded graphical view of the results of the analysis:

23 <int

```

1      Model_id  ="1"
2      environment_type="RT"
3      meta_data="./fModels/robotExample.meta"
4      tumblr_location="./fModels/robotExample.tumblr.location"
5      input_format="JSON"
6      pmmls="AUTONOMOUS_AGENTS.PMML"
7      Model_type ="AUTONOMOUS_AGENTS"
8  >
9  <vault  >
10
11  <door:LOCATION>
12      <lock name="DETERMINE LOCATION"
13          inkey="INPUT" inkeyname="lat"
14          inkey2="INPUT" inkeyname2="long"
15          function="ROUND"
16          fnct1-prec="-2"
17          function-1="JOIN"
18          fnct2-delim=":"
19          tumblr='LAT_LONG.key'
20          outkey="TEMP"  outkeyname="location"
21          type="STRING"
22      />
23      <lock name="DETERMINE WEATHER"
24          inkey="TEMP" inkeyname="location"
25          mesh='MESHRT.RECENTWEATHER'
26          mesh-query='HASH'
27          outkey="TEMP"  outkeyname="WEATHERDATA"
28          type="ARRAY"
29      />
30      <lock name="EXPLODE DATA"
31          inkey="TEMP" inkeyname="WEATHERDATA"
32          function="EXPLODE"
33          fnct-delim=":"
34          outkey="MODELDATA"  outkeystartindex=1
35      />
36      <lock name="USER SETTINGS"
37          inkey="INPUT" inkeyname="USERID"
38          mesh='MESHRT.AUTONOMOUSAGENT.SETTINGS'
39          mesh-query='HASH'
40          outkey="TEMP"  outkeyname="USERSETTINGS"
41          type="ARRAY"
42      />
43      <lock name="EXPLODE USER"
44          inkey="TEMP" inkeyname="USERSETTINGS"
45          function="EXPLODE"
46          fnct-delim=":"
47          outkey="USERDATA"  outkeystartindex=1
48      />
49      <lock name="RUN MODELE"
50          inkey="MODELDATA"
51          inkey1="USERDATA"
52          function="TREE"
53          fnc-pmml="AUTONOMOUS_AGENTS.PMML"
54          outkey="OUTPUT"  outkeyname="WEATHER"
55          type="NUMERIC"
56      />
57  </door>
58  </vault>
59
60

```

1 **[00315]** A non-limiting, example listing of data that the MCB-Platform may return
 2 based on a query is provided below. In this example, a user may log into a website via a
 3 computing device. The computing device may provide a IP address, and a timestamp to
 4 the MCB-Platform. In response, the MCB-Platform may identify a profile of the user
 5 from its database, and based on the profile, return potential merchants for offers or
 6 coupons:

```

7 -----
8 ----- Use Case 3 -----
9 -- User log into a website
10 -- Only IP address, GMT and day of week is passed to Mesh
11 -- Mesh matches profile based on Affinity Group
12 -- Mesh returns potential Merchants for offers or coupons based on tempory
13 model using suppression rules
14 -----
15 -- Test case 1 IP:24:227:206 Hour:9 Day:3
16 -- Test case 2 IP:148:181:75 Hour:4 Day:5
17 -----
18 ----- AffinityGroup Lookup -----
19 -----
20 Look up test case 1
21 [OrderedDict([('ISACTIVE', 'True'), ('ENTITYKEY', '24:227:206:3:1'), ('XML',
22 None), ('AFFINITYGROUPNAME', '24:227:206:3:1'), ('DESCRIPTION', None),
23 ('TYPEOF', None), ('UUID', '5f8df970b9ff11e09ab9270cf67eca90'))],
24 OrderedDict([('ISACTIVE', 'True'), ('BASEUUID',
25 '4fbea327b9ff11e094f433b5d7c45677'), ('TOKENENTITYKEY',
26 '4fbea327b9ff11e094f433b5d7c45677:TOKEN:349:F'), ('BASETYPE',
27 'MODEL_002_001_00'), ('STATUS', 'ACTIVE'), ('ISSUEDDATE', None), ('WEIGHT',
28 '349'), ('CATEGORY', 'F'), ('DOUBLELINKED', None), ('UUID',
29 '6b6aab39b9ff11e08d850dc270e3ea06'))], OrderedDict([('ISACTIVE', 'True'),
30 ('BASEUUID', '4fbea328b9ff11e0a5f833b5d7c45677'), ('TOKENENTITYKEY',
31 '4fbea328b9ff11e0a5f833b5d7c45677:TOKEN:761:1'), ('BASETYPE',
32 'MODEL_003_001_00'), ('STATUS', 'ACTIVE'), ('ISSUEDDATE', None), ('WEIGHT',
33 '761'), ('CATEGORY', '1'), ('DOUBLELINKED', None), ('UUID',
34 '68aaca40b9ff11e0ac799fd4e415d9de'))], OrderedDict([('ISACTIVE', 'True'),
35 ('BASEUUID', '4fbea328b9ff11e0a5f833b5d7c45677'), ('TOKENENTITYKEY',
36 '4fbea328b9ff11e0a5f833b5d7c45677:TOKEN:637:2'), ('BASETYPE',
37 'MODEL_003_001_00'), ('STATUS', 'ACTIVE'), ('ISSUEDDATE', None), ('WEIGHT',
38 '637'), ('CATEGORY', '2'), ('DOUBLELINKED', None), ('UUID',
39 '6b6d1c38b9ff11e08ce10dc270e3ea06'))], OrderedDict([('ISACTIVE', 'True'),
40 ('BASEUUID', '4fbea328b9ff11e0a5f833b5d7c45677'), ('TOKENENTITYKEY',
41 '4fbea328b9ff11e0a5f833b5d7c45677:TOKEN:444:3'), ('BASETYPE',
42 'MODEL_003_001_00'), ('STATUS', 'ACTIVE'), ('ISSUEDDATE', None), ('WEIGHT',
43 '444'), ('CATEGORY', '3'), ('DOUBLELINKED', None), ('UUID',
44 '6342aa53b9ff11e0bcd9fd4e415d9de'))], OrderedDict([('ISACTIVE', 'True'),
45 ('BASEUUID', '4fbea328b9ff11e0a5f833b5d7c45677'), ('TOKENENTITYKEY',
46 '4fbea328b9ff11e0a5f833b5d7c45677:TOKEN:333:4'), ('BASETYPE',
47 'MODEL_003_001_00'), ('STATUS', 'ACTIVE'), ('ISSUEDDATE', None), ('WEIGHT',
48 '333'), ('CATEGORY', '4'), ('DOUBLELINKED', None), ('UUID',
49 '62bd26a2b9ff11e0bc239fd4e415d9de'))], OrderedDict([('ISACTIVE', 'True'),
50 ('BASEUUID', '4fbea328b9ff11e0a5f833b5d7c45677'), ('TOKENENTITYKEY',
51 '4fbea328b9ff11e0a5f833b5d7c45677:TOKEN:307:5'), ('BASETYPE',
52 'MODEL_003_001_00'), ('STATUS', 'ACTIVE'), ('ISSUEDDATE', None), ('WEIGHT',

```

```

1  '307'), ('CATEGORY', '5'), ('DOUBLELINKED', None), ('UUID',
2  '6b6d1c39b9ff11e0986c0dc270e3ea06'))], OrderedDict([('ISACTIVE', 'True'),
3  ('BASEUUID', '4fbea32db9ff11e09f3e33b5d7c45677'), ('TOKENENTITYKEY',
4  '4fbea32db9ff11e09f3e33b5d7c45677:TOKEN:801:Spend'), ('BASETYPE',
5  'MODEL_008_001_00'), ('STATUS', 'ACTIVE'), ('ISSUEDDATE', None), ('WEIGHT',
6  '801'), ('CATEGORY', 'Spend'), ('DOUBLELINKED', None), ('UUID',
7  '6b6d1c3ab9ff11e0a4ec0dc270e3ea06'))], OrderedDict([('ISACTIVE', 'True'),
8  ('BASEUUID', '4fbea32eb9ff11e0b55133b5d7c45677'), ('TOKENENTITYKEY',
9  '4fbea32eb9ff11e0b55133b5d7c45677:TOKEN:1:Volume'), ('BASETYPE',
10 'MODEL_009_001_00'), ('STATUS', 'ACTIVE'), ('ISSUEDDATE', None), ('WEIGHT',
11 '1'), ('CATEGORY', 'Volume'), ('DOUBLELINKED', None), ('UUID',
12 '62a09df3b9ff11e090d79fd4e415d9de'))]]
13 Found a direct match
14 148:181:75:1:2
15 -- Failed to find a direct match
16 -- Try again with only IP address and hour
17 [OrderedDict([('ISACTIVE', 'True'), ('ENTITYKEY', '148:181:75:1:1'), ('XML',
18 None), ('AFFINITYGROUPNAME', '148:181:75:1:1'), ('DESCRIPTION', None),
19 ('TYPEOF', None)])]
20 -- Found match for case 2
21 -----
22 ----- Temporary model rules-----
23 -----
24 {1: {'LOWER': 10, 'BASETYPE': ['MODEL_002_001_00', 'MODEL_003_001_00'],
25 'attribute': 'WEIGHT', 'rule': 'NEAR', 'OP': 'PROX', 'type': 'TOKENENTITY',
26 'HIGHER': 10}, 2: {'type': ['MERCHANT'], 'rule': 'FOLLOW'}, 3: {'rule':
27 'RESTRICTSUBTYPE', 'BASETYPE': ['MODEL_002_001_00', 'MODEL_003_001_00']}}
28 -----
29 ----- Temporary Model Output-----
30 ----- For Use Case 1 -----
31 -----
32 -- Number of Nodes:102
33 _____LIVRARIASICILIAN
34 _____GDPCOLTD
35 _____GOODWILLINDUSTRIES
36 _____DISCOUNTDE
37 _____BARELANCHOE
38 _____BLOOMINGDALES
39 _____PARCWORLDTENNIS
40 _____STRIDERITEOUTLET
41 _____PARCCEANOR
42 _____PONTOFRIO
43 _____FNACPAULISTA
44 _____FINISHLINE
45 _____WALMARTCENTRAL
46 _____BESNIINTERLARGOS
47 _____PARCLOJASCOLOMBO
48 _____SHOPTIMEINTER
49 _____BEDBATHBEYOND
50 _____MACYSWEST
51 _____PARCRIACHUELOFILIAL
52 _____JCPENNEYCORPINC
53 _____PARCLOJASRENNERFL
54 _____PARCPAQUETAESPORTES
55 _____MARISALJ
56 _____PARCLEADERMAGAZINE
57 _____INTERFLORA
58 _____DECATHLON
59 _____PERNAMBUCANASFL
60 _____KARSTADTDE
61 _____PARCCEAMCO
62 _____CHAMPS
63 _____ACCESSORIZE

```

1 BLOOMINGDALESDVRS
2 PARCLIVRARIACULTURA
3 PARCEALOJA
4 ARQUIBANCADA
5 KITBAG
6 FREDERICKSOFHLWD
7 WALMART
8 PARCLOJASINSINUANTE
9 WALMARTCONTAGEM
10 FOOTLOCKER
11 PARCSANTALOLLA
12 RICARDOELETRO
13 PARCPONTOFRIO
14 DOTPAYPLPOLSKA
15 CAMICADO
16 KARSTADT
17 PARCRAMSONS
18 PARCGREGORY
19 GREMIOFBPA
20 WALMARTSJC
21 PRODIRECTSOCCERLTD
22 LAVIEENROSE
23 PARCMARISALJ
24 ORDERS
25 PARCNSNNATALNORTE
26 LOJASINSINUANTE
27 B
28 CITYCOUNTY
29 WALMARTPACAEMBU
30 SOHO
31 WALMARTOSASCO
32 FOSSILSTORESINC
33 MENARDSCLIO
34 PARCPEQUENTE
35 BEALLS
36 THEHOMEDEPOT
37 VIAMIA
38 PARCLOJASRIACHUELO
39 PARCLOJASMILANO
40 NORDSTROM
41 WAILANACOFFEEHOUSE
42 LANCHOEBELLA
43 PUKET
44 WALMARTSTORESINC
45 PARCPERNAMBUCANASFL
46 SMARTSHOPPER
47 PARCMAGAZINELUIZASP
48 COLUMBIASPORTSWEARCO
49 BARELANCESTADA
50 DONATEEBAY
51 PARCRICARDOELETRO
52 PARCDISANTINNI
53 SCHUHCOUK
54 CEANOR
55 PARCCAMICADO
56 PARCCENTAUROCE
57 PARCMARLUIJOIAS
58 ALBADAH
59 MARTINEZ
60 MONEYBOOKERSLTD
61 MACYS
62 PARCRIOCENTER
63 PARCCASASBAHIA

```

1  _____ PARCSUBMARINOLOJA
2  _____ INC
3  _____ SUBMARINOLOJA
4  _____ LOJASRENNERFL
5  _____ RIACHUELOFILIAL
6  _____ PARCSOHNODOSPES
7  _____ PINKBIJU
8  _____ PARCEAMRB
9  -----
10 ----- Temporary model Output -----
11 ----- For Use Case 2 -----
12 -----
13 -- Number of Nodes:3
14 _____ KITBAG
15 _____ COLUMBIASPORTSWEARCO
16 _____ GREMIOFBPA
17 -----
18 ----- End of Example Use Case -----
19 -----
20
21

```

[00316] In some embodiments, the MCB-Platform may provide access to information on a need-to-know basis to ensure the security of data of entities on which the MCB-Platform stores information. Thus, in some embodiments, access to information from the centralized platform may be restricted based on the originator as well as application services for which the data is requested. In some embodiments, the MCB-Platform may thus allow a variety of flexible application services to be built on a common database infrastructure, while preserving the integrity, security, and accuracy of entity data. In some implementations, the MCB-Platform may generate, update, maintain, store and/or provide profile information on entities, as well as a social graph that maintains and updates interrelationships between each of the entities stored within the MCB-Platform. For example, the MCB-Platform may store profile information on an issuer bank 1802a (see profile 1803a), a acquirer bank 1802b (see profile 1803b), a consumer 1802c (see profile 1803c), a user 1802d (see profile 1803d), a merchant 1802e (see profile 1803e), a second merchant 1802f (see profile 1803f). The MCB-Platform may also store relationships between such entities. For example, the MCB-Platform may store information on a relationship of the issuer bank 1802a to the consumer 1802c

1 shopping at merchant 1802e, who in turn may be related to user 1802d, who might bank
2 at the bank 1802b that serves as acquirer for merchant 1802f.

3 **[00317]** FIGURES 19A-F show block diagrams illustrating example aspects of data
4 models within a centralized personal information platform in some embodiments of the
5 MCB-Platform. In various embodiments, the MCB-Platform may store a variety of
6 attributes of entities according to various data models. A few non-limiting example data
7 models are provided below. In some embodiments, the MCB-Platform may store user
8 profile attributes. For example, a user profile model may store user identifying
9 information 1901, user aliases 1902, email addresses 1903, phone numbers 1904,
10 addresses 1905, email address types 1906, address types 1907, user alias types 1908,
11 notification statuses 1909, ISO country 1910, phone number types 1911, contract
12 information with the MCB-Platform 1912, user authorization status 1913, user profile
13 status 1914, security answer 1915, security questions 1916, language 1917, time zone
14 1918, and/or the like, each of the above field types including one or more fields and field
15 values. As another example, a user financial attributes model may store user identifying
16 information 1920, user financial account information 1921, account contract
17 information 1922, user financial account role 1923, financial account type 1924,
18 financial account identifying information 1925, contract information 1926, financial
19 account validation 1927, financial account validation type 1928, and/or the like. As
20 another example, a user payment card attributes data model may include field types
21 such as, but not limited to: user identifying information 1930, user financial account
22 information 1931, user financial account role 1932, account consumer applications 1933,
23 user consumer application 1934, financial account type 1935, financial account
24 validation type 1936, financial account information 1937, consumer application

1 information 1938, consumer application provider information 1939, and/or the like. As
2 another example, a user services attributes data model may include field types such as,
3 but not limited to: user identifying information 1940, user alias 1941, consumer
4 application user alias status 1942, user alias status 1943, status change reason code
5 1944, user contract 1945, contract information 1946, user service attribute value 1947,
6 consumer application attributes 1948, account service attribute value, account contract
7 1950, user profile status 1961, contract business role 1952, contract business 1953, client
8 information 1954, contract role 1955, consumer application 1956, user activity audit
9 1957, login results 1958, and/or the like. As another example, a user services usage
10 attributes data model may include field types such as, but not limited to: user
11 identifying information 1960, user alias 1961, consumer application user alias status
12 1962, status change reason code 1963, user alias status 1964, user consumer application
13 1965, user login audit 1966, login result 1967, account service attribute value 1968,
14 account consumer application 1969, consumer application 1970, consumer application
15 provider 1971, login result 1972, and/or the like. As another example, a user graph
16 attributes data model may include field types such as, but not limited to: user
17 identifying information 1980, user contact 1981, consumer application user alias status
18 1982, relationship 1983, and/or the like. In some embodiments, the MCB-Platform may
19 store each object (e.g., user, merchant, issuer, acquirer, IP address, household, etc.) as a
20 node in graph database, and store data with respect to each node in a format such as the
21 example format provided below:

```
22 <Nodes Data>
23 ID, Nodes, Label
24 2fdc7e3fbd1c11e0be645528b00e8d0e, 2fdc7e3fbd1c11e0be645528b00e8d0e, AFFINITYGROUP
25 NAME: 49:95:0:3:1
26 32b1d53ebd1c11e094172557fb829fdf, 32b1d53ebd1c11e094172557fb829fdf, TOKENENTITYKE
27 Y:2b8494f0bd1c11e09c856d888c43f7c2:TOKEN:0:F
```

```
1 2e6381e4bd1c11e0b9ffc929a54bb0fd,2e6381e4bd1c11e0b9ffc929a54bb0fd,MERCHANTNAME:
2      MERCHANT_ABC
3 2fdc7e3dbd1c11e0a22d5528b00e8d0e,2fdc7e3dbd1c11e0a22d5528b00e8d0e,AFFINITYGROUP
4 NAME:49:95:0:1:1
5 2e6381e7bd1c11e091b7c929a54bb0fd,2e6381e7bd1c11e091b7c929a54bb0fd,MERCHANTNAME:
6      MERCHANT_XYZ
7 2cf8cbabb1c11e0894a5de4f9281135,2cf8cbabb1c11e0894a5de4f9281135,USERNAME:0000
8 60FF6557F103
9 2e6381debd1c11e0b336c929a54bb0fd,2e6381debd1c11e0b336c929a54bb0fd,MERCHANTNAME:
10     MERCHANT_123
11 2e6381e0bd1c11e0b4e8c929a54bb0fd,2e6381e0bd1c11e0b4e8c929a54bb0fd,MERCHANTNAME:
12     MERCHANT_FGH
13 2cf681c1bd1c11e0b8815de4f9281135,2cf681c1bd1c11e0b8815de4f9281135,USERNAME:0000
14 30C57080FFE8
15 2b8494f1bd1c11e0acbd6d888c43f7c2,2b8494f1bd1c11e0acbd6d888c43f7c2,MODELNAME:MOD
16 EL_003_001_00
17 32b44638bd1c11e0b01c2557fb829fdf,32b44638bd1c11e0b01c2557fb829fdf,TOKENENTITYKE
18 Y:2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:1
19 2fdc7e40bd1c11e094675528b00e8d0e,2fdc7e40bd1c11e094675528b00e8d0e,AFFINITYGROUP
20 NAME:49:95:0:4:1
21 2b8494f0bd1c11e09c856d888c43f7c2,2b8494f0bd1c11e09c856d888c43f7c2,MODELNAME:MOD
22 EL_002_001_00
23 32b44639bd1c11e0b15b2557fb829fdf,32b44639bd1c11e0b15b2557fb829fdf,TOKENENTITYKE
24 Y:2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:2
25 32ce84febd1c11e0b0112557fb829fdf,32ce84febd1c11e0b0112557fb829fdf,TOKENENTITYKE
26 Y:2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:4
27 2e6381e3bd1c11e095b1c929a54bb0fd,2e6381e3bd1c11e095b1c929a54bb0fd,MERCHANTNAME:
28     MERCHANT_789
29 34582a87bd1c11e080820167449bc60f,34582a87bd1c11e080820167449bc60f,TOKENENTITYKE
30 Y:2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:778:5
31 2e6381e5bd1c11e0b62cc929a54bb0fd,2e6381e5bd1c11e0b62cc929a54bb0fd,MERCHANTNAME:
32     MERCHANT_456
33 2fdc7e3ebd1c11e088b55528b00e8d0e,2fdc7e3ebd1c11e088b55528b00e8d0e,AFFINITYGROUP
34 NAME:49:95:0:2:1
35 32c4e80dbd1c11e09e442557fb829fdf,32c4e80dbd1c11e09e442557fb829fdf,TOKENENTITYKE
36 Y:2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:774:5
37 2e6381e1bd1c11e0bf28c929a54bb0fd,2e6381e1bd1c11e0bf28c929a54bb0fd,MERCHANTNAME:
38     MERCHANT_WER
39 2cf681b8bd1c11e08be85de4f9281135,2cf681b8bd1c11e08be85de4f9281135,USERNAME:0000
40 2552FC930FF8
41 2cf8cba8bd1c11e09fbc5de4f9281135,2cf8cba8bd1c11e09fbc5de4f9281135,USERNAME:0000
42 570FF1B46A24
43 32b4463abd1c11e0bdaa2557fb829fdf,32b4463abd1c11e0bdaa2557fb829fdf,TOKENENTITYKE
44 Y:2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:3
45 2cf8cbaebd1c11e0b6515de4f9281135,2cf8cbaebd1c11e0b6515de4f9281135,USERNAME:0000
46 64A20FF962D4
47 2e6381e6bd1c11e08087c929a54bb0fd,2e6381e6bd1c11e08087c929a54bb0fd,MERCHANTNAME:
48     MERCHANT_496
49 2e6381e2bd1c11e0941dc929a54bb0fd,2e6381e2bd1c11e0941dc929a54bb0fd,MERCHANTNAME:
50     MERCHANT_SDF
51 <Edge Data>Source,Target,label, Weight
52 32ce84febd1c11e0b0112557fb829fdf,2e6381e6bd1c11e08087c929a54bb0fd,MODEL_003_001
53 _00,2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
54 2fdc7e3ebd1c11e088b55528b00e8d0e,32ce84febd1c11e0b0112557fb829fdf,MODEL_003_001
55 _00,2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
56 2e6381e2bd1c11e0941dc929a54bb0fd,34582a87bd1c11e080820167449bc60f,MODEL_003_001
57 _00,2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:778:5,778
58 2b8494f1bd1c11e0acbd6d888c43f7c2,34582a87bd1c11e080820167449bc60f,MODEL_003_001
59 _00,2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:778:5,778
60 2e6381e1bd1c11e0bf28c929a54bb0fd,32b44639bd1c11e0b15b2557fb829fdf,MODEL_003_001
61 _00,2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:2,0
62 2e6381e0bd1c11e0b4e8c929a54bb0fd,32ce84febd1c11e0b0112557fb829fdf,MODEL_003_001
63 _00,2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
```

1 32b44639bdlc11e0b15b2557fb829fdf, 2e6381e6bdlc11e08087c929a54bb0fd, MODEL_003_001
2 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:2,0
3 2e6381e1bdlc11e0bf28c929a54bb0fd, 32ce84febdlc11e0b0112557fb829fdf, MODEL_003_001
4 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
5 2e6381debd1c11e0b336c929a54bb0fd, 32ce84febdlc11e0b0112557fb829fdf, MODEL_003_001
6 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
7 2e6381e3bdlc11e095b1c929a54bb0fd, 34582a87bdlc11e080820167449bc60f, MODEL_003_001
8 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:778:5,778
9 2fdc7e40bdlc11e094675528b00e8d0e, 32b44639bdlc11e0b15b2557fb829fdf, MODEL_003_001
10 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:2,0
11 2b8494f1bdlc11e0acbd6d888c43f7c2, 32b4463abdlc11e0bdaa2557fb829fdf, MODEL_003_001
12 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:3,0
13 2e6381e3bdlc11e095b1c929a54bb0fd, 32b4463abdlc11e0bdaa2557fb829fdf, MODEL_003_001
14 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:3,0
15 2e6381e3bdlc11e095b1c929a54bb0fd, 32b1d53ebdlc11e094172557fb829fdf, MODEL_002_001
16 _00, 2b8494f0bdlc11e09c856d888c43f7c2:TOKEN:0:F,0
17 2e6381e5bdlc11e0b62cc929a54bb0fd, 34582a87bdlc11e080820167449bc60f, MODEL_003_001
18 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:778:5,778
19 2cf8cbabbd1c11e0894a5de4f9281135, 32b44638bdlc11e0b01c2557fb829fdf, MODEL_003_001
20 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:1000:1,1000
21 2cf681b8bdlc11e08be85de4f9281135, 32b1d53ebdlc11e094172557fb829fdf, MODEL_002_001
22 _00, 2b8494f0bdlc11e09c856d888c43f7c2:TOKEN:0:F,0
23 32b4463abdlc11e0bdaa2557fb829fdf, 2e6381e6bdlc11e08087c929a54bb0fd, MODEL_003_001
24 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:3,0
25 2e6381debd1c11e0b336c929a54bb0fd, 32b44639bdlc11e0b15b2557fb829fdf, MODEL_003_001
26 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:2,0
27 2e6381e1bdlc11e0bf28c929a54bb0fd, 32b44638bdlc11e0b01c2557fb829fdf, MODEL_003_001
28 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:1000:1,1000
29 2e6381e5bdlc11e0b62cc929a54bb0fd, 32ce84febdlc11e0b0112557fb829fdf, MODEL_003_001
30 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
31 2e6381e1bdlc11e0bf28c929a54bb0fd, 32b4463abdlc11e0bdaa2557fb829fdf, MODEL_003_001
32 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:3,0
33 2e6381e2bdlc11e0941dc929a54bb0fd, 32b44639bdlc11e0b15b2557fb829fdf, MODEL_003_001
34 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:2,0
35 2b8494f1bdlc11e0acbd6d888c43f7c2, 32c4e80dbdlc11e09e442557fb829fdf, MODEL_003_001
36 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:774:5,774
37 2e6381e2bdlc11e0941dc929a54bb0fd, 32b1d53ebdlc11e094172557fb829fdf, MODEL_002_001
38 _00, 2b8494f0bdlc11e09c856d888c43f7c2:TOKEN:0:F,0
39 2e6381e4bdlc11e0b9ffc929a54bb0fd, 32b4463abdlc11e0bdaa2557fb829fdf, MODEL_003_001
40 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:3,0
41 2fdc7e3fbd1c11e0be645528b00e8d0e, 32b4463abdlc11e0bdaa2557fb829fdf, MODEL_003_001
42 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:3,0
43 2e6381e1bdlc11e0bf28c929a54bb0fd, 32b1d53ebdlc11e094172557fb829fdf, MODEL_002_001
44 _00, 2b8494f0bdlc11e09c856d888c43f7c2:TOKEN:0:F,0
45 2fdc7e40bdlc11e094675528b00e8d0e, 32ce84febdlc11e0b0112557fb829fdf, MODEL_003_001
46 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
47 2cf8cba8bdlc11e09fbc5de4f9281135, 32c4e80dbdlc11e09e442557fb829fdf, MODEL_003_001
48 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:774:5,774
49 2e6381e2bdlc11e0941dc929a54bb0fd, 32b44638bdlc11e0b01c2557fb829fdf, MODEL_003_001
50 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:1000:1,1000
51 2e6381e4bdlc11e0b9ffc929a54bb0fd, 32b1d53ebdlc11e094172557fb829fdf, MODEL_002_001
52 _00, 2b8494f0bdlc11e09c856d888c43f7c2:TOKEN:0:F,0
53 2e6381e5bdlc11e0b62cc929a54bb0fd, 32b44639bdlc11e0b15b2557fb829fdf, MODEL_003_001
54 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:2,0
55 32b1d53ebdlc11e094172557fb829fdf, 2e6381e6bdlc11e08087c929a54bb0fd, MODEL_002_001
56 _00, 2b8494f0bdlc11e09c856d888c43f7c2:TOKEN:0:F,0
57 2b8494f1bdlc11e0acbd6d888c43f7c2, 32b44639bdlc11e0b15b2557fb829fdf, MODEL_003_001
58 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:2,0
59 2e6381e3bdlc11e095b1c929a54bb0fd, 32b44638bdlc11e0b01c2557fb829fdf, MODEL_003_001
60 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:1000:1,1000
61 2fdc7e3dbdlc11e0a22d5528b00e8d0e, 32ce84febdlc11e0b0112557fb829fdf, MODEL_003_001
62 _00, 2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:1000:4,1000

1 2cf681c1bd1c11e0b8815de4f9281135, 32b44638bd1c11e0b01c2557fb829fdf, MODEL_003_001
2 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:1,1000
3 2cf681c1bd1c11e0b8815de4f9281135, 32b1d53ebd1c11e094172557fb829fdf, MODEL_002_001
4 _00, 2b8494f0bd1c11e09c856d888c43f7c2:TOKEN:0:F,0
5 2e6381e3bd1c11e095b1c929a54bb0fd, 32b44639bd1c11e0b15b2557fb829fdf, MODEL_003_001
6 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:2,0
7 2fdc7e3fbd1c11e0be645528b00e8d0e, 32b1d53ebd1c11e094172557fb829fdf, MODEL_002_001
8 _00, 2b8494f0bd1c11e09c856d888c43f7c2:TOKEN:0:F,0
9 32b44638bd1c11e0b01c2557fb829fdf, 2e6381e6bd1c11e08087c929a54bb0fd, MODEL_003_001
10 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:1,1000
11 2cf8cbaebd1c11e0b6515de4f9281135, 32ce84feb1c11e0b0112557fb829fdf, MODEL_003_001
12 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
13 2e6381e6bd1c11e08087c929a54bb0fd, 32b1d53ebd1c11e094172557fb829fdf, MODEL_002_001
14 _00, 2b8494f0bd1c11e09c856d888c43f7c2:TOKEN:0:F,0
15 2e6381e7bd1c11e091b7c929a54bb0fd, 34582a87bd1c11e080820167449bc60f, MODEL_003_001
16 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:778:5,778
17 2e6381e1bd1c11e0bf28c929a54bb0fd, 34582a87bd1c11e080820167449bc60f, MODEL_003_001
18 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:778:5,778
19 2e6381e5bd1c11e0b62cc929a54bb0fd, 32b1d53ebd1c11e094172557fb829fdf, MODEL_002_001
20 _00, 2b8494f0bd1c11e09c856d888c43f7c2:TOKEN:0:F,0
21 2b8494f0bd1c11e09c856d888c43f7c2, 32b1d53ebd1c11e094172557fb829fdf, MODEL_002_001
22 _00, 2b8494f0bd1c11e09c856d888c43f7c2:TOKEN:0:F,0
23 2b8494f1bd1c11e0acbd6d888c43f7c2, 32b44638bd1c11e0b01c2557fb829fdf, MODEL_003_001
24 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:1,1000
25 2e6381e6bd1c11e08087c929a54bb0fd, 32b4463abd1c11e0bdaa2557fb829fdf, MODEL_003_001
26 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:3,0
27 2b8494f1bd1c11e0acbd6d888c43f7c2, 32ce84feb1c11e0b0112557fb829fdf, MODEL_003_001
28 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
29 2cf681c1bd1c11e0b8815de4f9281135, 32b44639bd1c11e0b15b2557fb829fdf, MODEL_003_001
30 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:2,0
31 2cf681c1bd1c11e0b8815de4f9281135, 32b4463abd1c11e0bdaa2557fb829fdf, MODEL_003_001
32 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:3,0
33 2e6381e2bd1c11e0941dc929a54bb0fd, 32b4463abd1c11e0bdaa2557fb829fdf, MODEL_003_001
34 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:3,0
35 2e6381e3bd1c11e095b1c929a54bb0fd, 32ce84feb1c11e0b0112557fb829fdf, MODEL_003_001
36 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
37 2e6381e6bd1c11e08087c929a54bb0fd, 32ce84feb1c11e0b0112557fb829fdf, MODEL_003_001
38 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
39 2e6381e6bd1c11e08087c929a54bb0fd, 34582a87bd1c11e080820167449bc60f, MODEL_003_001
40 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:778:5,778
41 2e6381e6bd1c11e08087c929a54bb0fd, 32b44638bd1c11e0b01c2557fb829fdf, MODEL_003_001
42 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:1,1000
43 2fdc7e3ebd1c11e088b55528b00e8d0e, 32b44639bd1c11e0b15b2557fb829fdf, MODEL_003_001
44 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:2,0
45 2e6381e5bd1c11e0b62cc929a54bb0fd, 32b4463abd1c11e0bdaa2557fb829fdf, MODEL_003_001
46 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:3,0
47 2e6381e4bd1c11e0b9ffc929a54bb0fd, 34582a87bd1c11e080820167449bc60f, MODEL_003_001
48 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:778:5,778
49 2e6381e4bd1c11e0b9ffc929a54bb0fd, 32b44638bd1c11e0b01c2557fb829fdf, MODEL_003_001
50 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:1,1000
51 34582a87bd1c11e080820167449bc60f, 2e6381e6bd1c11e08087c929a54bb0fd, MODEL_003_001
52 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:778:5,778
53 2e6381e6bd1c11e08087c929a54bb0fd, 32b44639bd1c11e0b15b2557fb829fdf, MODEL_003_001
54 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:2,0
55 2e6381e5bd1c11e0b62cc929a54bb0fd, 32b44638bd1c11e0b01c2557fb829fdf, MODEL_003_001
56 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:1,1000
57 2fdc7e3fbd1c11e0be645528b00e8d0e, 32b44638bd1c11e0b01c2557fb829fdf, MODEL_003_001
58 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:1000:1,1000
59 2cf681b8bd1c11e08be85de4f9281135, 32b44639bd1c11e0b15b2557fb829fdf, MODEL_003_001
60 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:2,0
61 2e6381e4bd1c11e0b9ffc929a54bb0fd, 32b44639bd1c11e0b15b2557fb829fdf, MODEL_003_001
62 _00, 2b8494f1bd1c11e0acbd6d888c43f7c2:TOKEN:0:2,0

```

1  2cf681b8bdlc11e08be85de4f9281135,32b4463abdlc11e0bdaa2557fb829fdf,MODEL_003_001
2  _00,2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:3,0
3  2e6381e4bdlc11e0b9ffc929a54bb0fd,32ce84febd1c11e0b0112557fb829fdf,MODEL_003_001
4  _00,2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
5  2e6381e2bdlc11e0941dc929a54bb0fd,32ce84febd1c11e0b0112557fb829fdf,MODEL_003_001
6  _00,2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:1000:4,1000
7  2fdc7e3dbdlc11e0a22d5528b00e8d0e,32b44639bdlc11e0b15b2557fb829fdf,MODEL_003_001
8  _00,2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:0:2,0
9  2cf681b8bdlc11e08be85de4f9281135,32b44638bdlc11e0b01c2557fb829fdf,MODEL_003_001
10 _00,2b8494f1bdlc11e0acbd6d888c43f7c2:TOKEN:1000:1,1000
11
12
13
14

```

[00318] In alternate examples, the MCB-Platform may store data in a JavaScript Object Notation (“JSON”) format. The stored information may include data regarding the object, such as, but not limited to: commands, attributes, group information, payment information, account information, etc., such as in the example below:

```

19  {'MERCHANT': {'TYPEOFTYPES': ['MERCHANTS', 'SYNTHETICNETWORKS'], 'FUNCTIONS':
20  {'ENTITYCREATION': 'putNetwork'}
21  , 'UNIQUEATTRIBUTES': ['MERCHANTNAME'], 'TOKENENTITIESRELATIONSHIPS': [],
22  'ATTRIBUTES': {'MERCHANT': (2, 'STRING', 0, 'VALUE'), 'MERCH_ZIP_CD': (7,
23  'STRING', 0, 'VALUE'), 'MERCH_NAME': (8, 'STRING', 0, 'VALUE'), 'MERCHANTNAME':
24  (3, 'STRING', 0, 'VALUE'), 'ACQ_CTRY_NUM': (4, 'STRING', 0, 'VALUE'),
25  'ACQ_PCR': (6, 'STRING', 0, 'VALUE'), 'ACQ_REGION_NUM': (5, 'STRING', 0,
26  'VALUE'), 'ISACTIVE': (0, 'BOOL', 1, 'VALUE'), 'ENTITYKEY': (1, 'STRING', 0,
27  'VALUE')}}
28  }
29  , 'AFFINITYGROUP': {'TYPEOFTYPES': ['AFFINITYGROUPS'], 'FUNCTIONS':
30  {'ENTITYCREATION': 'putNetwork'}
31  , 'UNIQUEATTRIBUTES': ['AFFINITYGROUPNAME'], 'TOKENENTITIESRELATIONSHIPS': [],
32  'ATTRIBUTES': {'XML': (2, 'STRING', 0, 'VALUE'), 'DESCRIPTION': (4, 'STRING',
33  0, 'VALUE'), 'ENTITYKEY': (1, 'STRING', 0, 'VALUE'), 'TYPEOF': (5, 'STRING', 0,
34  'VALUE'), 'AFFINITYGROUPNAME': (3, 'STRING', 0, 'VALUE'), 'ISACTIVE': (0,
35  'BOOL', 1, 'VALUE')}}
36  }
37  , 'CASCADINGPAYMENT': {'TYPEOFTYPES': ['CASCADINGPAYMENT'], 'FUNCTIONS':
38  {'ENTITYCREATION': 'putNetwork'}
39  , 'UNIQUEATTRIBUTES': ['CASCADINGPAYMENTNAME'], 'TOKENENTITIESRELATIONSHIPS':
40  ['GROUP'], 'ATTRIBUTES': {'STATUS': (2, 'STRING', 0, 'VALUE'), 'EXPDT': (6,
41  'DATETIME', 0, 'VALUE'), 'GROUP': (3, 'STRING', 0, 'VALUE'), 'RESTRICTIONS':
42  (7, 'DICT', 0, 'VALUE'), 'CASCADINGPAYMENTNAME': (4, 'STRING', 0, 'VALUE'),
43  'STARTDT': (5, 'DATETIME', 0, 'VALUE'), 'ISACTIVE': (0, 'BOOL', 1, 'VALUE'),
44  'ENTITYKEY': (1, 'STRING', 0, 'VALUE')}}
45  }
46  , 'GROUP': {'TYPEOFTYPES': [], 'FUNCTIONS': {'ENTITYCREATION': 'putNetwork'}
47  , 'UNIQUEATTRIBUTES': ['GROUPNAME'], 'TOKENENTITIESRELATIONSHIPS': {}
48  , 'ATTRIBUTES': {'GROUPNAME': (2, 'STRING', 0, 'VALUE'), 'DESCRIPTION': (2,
49  'STRING', 0, 'VALUE'), 'ISACTIVE': (0, 'BOOL', 1, 'VALUE'), 'ENTITYKEY': (1,
50  'STRING', 0, 'VALUE')}}
51  }
52  , 'USERS': {'TYPEOFTYPES': [], 'FUNCTIONS': {'ENTITYCREATION': 'putNetwork'}
53  , 'UNIQUEATTRIBUTES': ['USERSID'], 'TOKENENTITIESRELATIONSHIPS': {}
54  , 'ATTRIBUTES': {'USERSID': (2, 'STRING', 0, 'VALUE'), 'ISACTIVE': (0, 'BOOL',
55  1, 'VALUE'), 'ENTITYKEY': (1, 'STRING', 0, 'VALUE')}}
56  }

```

```

1  , 'TWITTERUSER': {'TYPEOFTYPES': ['TOKENENTITY'], 'FUNCTIONS':
2  {'ENTITYCREATION': 'putWGTNetwork'}
3  , 'UNIQUEATTRIBUTES': ['USERNAME'], 'TOKENENTITIESRELATIONSHIPS': ['USER'],
4  'ATTRIBUTES': {'USERNAME': (2, 'STRING', 0, 'VALUE'), 'CITY': (5, 'STRING', 0,
5  'VALUE'), 'ENTITYKEY': (1, 'STRING', 0, 'VALUE'), 'USERLINK': (6, 'STRING', 0,
6  'VALUE'), 'FULLNAME': (4, 'STRING', 0, 'VALUE'), 'USERTAG': (3, 'STRING', 0,
7  'VALUE'), 'ISACTIVE': (0, 'BOOL', 1, 'VALUE')}
8  }
9  , 'COUPON': {'TYPEOFTYPES': ['COUPON'], 'FUNCTIONS': {'ENTITYCREATION':
10 'putNetwork'}
11 , 'UNIQUEATTRIBUTES': ['COUPONNAME'], 'TOKENENTITIESRELATIONSHIPS':
12 ['MERCHANT'], 'ATTRIBUTES': {'STATUS': (2, 'STRING', 0, 'VALUE'), 'MERCHANT':
13 (3, 'STRING', 0, 'VALUE'), 'TITLE': (5, 'STRING', 0, 'VALUE'), 'NOTES': (7,
14 'STRING', 0, 'VALUE'), 'UPDATEDBY': (11, 'STRING', 0, 'VALUE'), 'ENTITYKEY':
15 (1, 'STRING', 0, 'VALUE'), 'DESCRIPTION': (6, 'STRING', 0, 'VALUE'),
16 'CREATEDBY': (10, 'STRING', 0, 'VALUE'), 'LASTUPDATEDT': (9, 'DATETIME', 0,
17 'VALUE'), 'EXPDT': (13, 'DATETIME', 0, 'VALUE'), 'RESTRICTIONS': (14, 'DICT',
18 0, 'VALUE'), 'COUPONNAME': (4, 'STRING', 0, 'VALUE'), 'CREATIONDT': (8,
19 'DATETIME', 0, 'VALUE'), 'STARTDT': (12, 'DATETIME', 0, 'VALUE'), 'ISACTIVE':
20 (0, 'BOOL', 1, 'VALUE')}
21 }
22 , 'MEMBERSHIP': {'TYPEOFTYPES': ['MEMBERSHIPS'], 'FUNCTIONS':
23 {'ENTITYCREATION': 'putNetwork'}
24 , 'UNIQUEATTRIBUTES': ['MEMBERSHIPNAME'], 'TOKENENTITIESRELATIONSHIPS':
25 ['MERCHANT'], 'ATTRIBUTES': {'STATUS': (2, 'STRING', 0, 'VALUE'), 'MERCHANT':
26 (3, 'STRING', 0, 'VALUE'), 'RESTRICTIONS': (7, 'DICT', 0, 'VALUE'),
27 'MEMBERSHIPNAME': (4, 'STRING', 0, 'VALUE'), 'STARTDT': (5, 'DATETIME', 0,
28 'VALUE'), 'EXPDT': (6, 'DATETIME', 0, 'VALUE'), 'ISACTIVE': (0, 'BOOL', 1,
29 'VALUE'), 'ENTITYKEY': (1, 'STRING', 0, 'VALUE')}
30 }
31 , 'USERSECURITY': {'TYPEOFTYPES': ['SECURITY'], 'FUNCTIONS': {'ENTITYCREATION':
32 'putNetwork'}
33 , 'UNIQUEATTRIBUTES': ['USERSECURITYNAME'], 'TOKENENTITIESRELATIONSHIPS':
34 ['USER'], 'ATTRIBUTES': {'STATUS': (2, 'STRING', 0, 'VALUE'), 'EXPDT': (6,
35 'DATETIME', 0, 'VALUE'), 'USERSECURITYNAME': (4, 'STRING', 0, 'VALUE'), 'USER':
36 (3, 'STRING', 0, 'VALUE'), 'RESTRICTIONS': (7, 'DICT', 0, 'VALUE'), 'STARTDT':
37 (5, 'DATETIME', 0, 'VALUE'), 'ISACTIVE': (0, 'BOOL', 1, 'VALUE'), 'ENTITYKEY':
38 (1, 'STRING', 0, 'VALUE')}
39 }
40 , 'MCC': {'TYPEOFTYPES': ['MCC'], 'FUNCTIONS': {'ENTITYCREATION':
41 'putWGTNetwork'}
42 , 'UNIQUEATTRIBUTES': ['MCCNAME', 'MCC'], 'TOKENENTITIESRELATIONSHIPS':
43 ['MCCSEG'], 'ATTRIBUTES': {'MCCSEG': (4, 'STRING', 0, 'VALUE'), 'MCC': (2,
44 'STRING', 0, 'VALUE'), 'MCCNAME': (3, 'STRING', 0, 'VALUE'), 'ISACTIVE': (0,
45 'BOOL', 1, 'VALUE'), 'ENTITYKEY': (1, 'STRING', 0, 'VALUE')}
46 }
47 , 'ZIPCODE': {'TYPEOFTYPES': ['LOCATION'], 'FUNCTIONS': {'ENTITYCREATION':
48 'putNetwork'}
49 , 'UNIQUEATTRIBUTES': ['ZIPCODE'], 'TOKENENTITIESRELATIONSHIPS': [],
50 'ATTRIBUTES': {'STATE': (4, 'STRING', 0, 'VALUE'), 'POPULATION': (3, 'STRING',
51 0, 'VALUE'), 'ZIPCODE': (2, 'STRING', 0, 'VALUE'), 'ISACTIVE': (0, 'BOOL', 1,
52 'VALUE'), 'ENTITYKEY': (1, 'STRING', 0, 'VALUE')}
53 }
54 , 'PAYMENTCARD': {'TYPEOFTYPES': ['PAYMENTCARDS'], 'FUNCTIONS':
55 {'ENTITYCREATION': 'putNetwork'}
56 , 'UNIQUEATTRIBUTES': ['CARDNUMBER'], 'TOKENENTITIESRELATIONSHIPS': ['USER'],
57 'ATTRIBUTES': {'EXPDATE': (5, 'DATETIME', 0, 'VALUE'), 'ENTITYKEY': (1,
58 'STRING', 0, 'VALUE'), 'CARDTYPE': (4, 'STRING', 0, 'VALUE'), 'CARDNUMBER': (2,
59 'STRING', 0, 'VALUE'), 'USER': (3, 'STRING', 0, 'VALUE'), 'ISACTIVE': (0,
60 'BOOL', 1, 'VALUE')}
61 }
62 , 'GENERICTOKEN': {'TYPEOFTYPES': ['COUPON'], 'FUNCTIONS': {'ENTITYCREATION':
63 'putNetwork'}

```

```

1  , 'UNIQUEATTRIBUTES': ['GENERTOKENNAME'], 'TOKENENTITIESRELATIONSHIPS':
2  ['MERCHANT'], 'ATTRIBUTES': {'STATUS': (2, 'STRING', 0, 'VALUE'), 'MERCHANT':
3  (3, 'STRING', 0, 'VALUE'), 'TITLE': (5, 'STRING', 0, 'VALUE'), 'NOTES': (7,
4  'STRING', 0, 'VALUE'), 'UPDATEDBY': (11, 'STRING', 0, 'VALUE'), 'ENTITYKEY':
5  (1, 'STRING', 0, 'VALUE'), 'DESCRIPTION': (6, 'STRING', 0, 'VALUE'),
6  'CREATEDBY': (10, 'STRING', 0, 'VALUE'), 'LASTUPDATEDT': (9, 'DATETIME', 0,
7  'VALUE'), 'EXPDT': (13, 'DATETIME', 0, 'VALUE'), 'RESTRICTIONS': (14, 'DICT',
8  0, 'VALUE'), 'STARTDT': (12, 'DATETIME', 0, 'VALUE'), 'CREATIONDT': (8,
9  'DATETIME', 0, 'VALUE'), 'GENERTOKENNAME': (4, 'STRING', 0, 'VALUE'),
10 'ISACTIVE': (0, 'BOOL', 1, 'VALUE')}
11 }
12 , 'USER': {'TYPEOFTYPES': ['USERS', 'SYNTHETICNETWORKS'], 'FUNCTIONS':
13 {'ENTITYCREATION': 'putNetwork'}}
14 , 'UNIQUEATTRIBUTES': ['USERNAME'], 'TOKENENTITIESRELATIONSHIPS': ['USERS'],
15 'ATTRIBUTES': {'USERNAME': (5, 'STRING', 0, 'VALUE'), 'USERS': (2, 'STRING', 0,
16 'VALUE'), 'FIRSTNAME': (3, 'STRING', 0, 'VALUE'), 'LASTNAME': (4, 'STRING', 0,
17 'VALUE'), 'ENTITYKEY': (1, 'STRING', 0, 'VALUE'), 'ISACTIVE': (0, 'BOOL', 1,
18 'VALUE')}
19 }
20 , 'TWEETS': {'TYPEOFTYPES': ['TOKENENTITY'], 'FUNCTIONS': {'ENTITYCREATION':
21 'putWGTNetwork'}}
22 , 'UNIQUEATTRIBUTES': ['TWEETID'], 'TOKENENTITIESRELATIONSHIPS':
23 ['TWITTERUSER'], 'ATTRIBUTES': {'Title': (4, 'STRING', 0, 'VALUE'), 'RawTweet':
24 (5, 'STRING', 0, 'VALUE'), 'DATETIME': (3, 'STRING', 0, 'VALUE'),
25 'CLEANEDTWEET': (6, 'STRING', 0, 'VALUE'), 'ENTITYKEY': (1, 'STRING', 0,
26 'VALUE'), 'TWEETID': (2, 'STRING', 0, 'VALUE'), 'ISACTIVE': (0, 'BOOL', 1,
27 'VALUE')}
28 }
29 , 'MODEL': {'TYPEOFTYPES': ['MODELS'], 'FUNCTIONS': {'ENTITYCREATION':
30 'putNetwork'}}
31 , 'UNIQUEATTRIBUTES': ['MODELNAME'], 'TOKENENTITIESRELATIONSHIPS': ['USER',
32 'MERCHANT', 'PAYMENTCARD'], 'ATTRIBUTES': {'XML': (2, 'STRING', 0, 'VALUE'),
33 'MODELNAME': (3, 'STRING', 0, 'VALUE'), 'DESCRIPTION': (4, 'STRING', 0,
34 'VALUE'), 'ENTITYKEY': (1, 'STRING', 0, 'VALUE'), 'TYPEOF': (5, 'STRING', 0,
35 'VALUE'), 'ISACTIVE': (0, 'BOOL', 1, 'VALUE')}
36 }
37 , 'MCCSEG': {'TYPEOFTYPES': ['MCCSEG'], 'FUNCTIONS': {'ENTITYCREATION':
38 'putWGTNetwork'}}
39 , 'UNIQUEATTRIBUTES': ['MCCSEGID'], 'TOKENENTITIESRELATIONSHIPS': {}
40 , 'ATTRIBUTES': {'MCCSEGID': (2, 'STRING', 0, 'VALUE'), 'MCCSEGNAME': (3,
41 'STRING', 0, 'VALUE'), 'ISACTIVE': (0, 'BOOL', 1, 'VALUE'), 'ENTITYKEY': (1,
42 'STRING', 0, 'VALUE')}
43 }
44 , 'TOKENENTITY': {'TYPEOFTYPES': ['TOKENENTITY'], 'FUNCTIONS':
45 {'ENTITYCREATION': 'putWGTNetwork'}}
46 , 'UNIQUEATTRIBUTES': ['TOKENENTITYKEY'], 'TOKENENTITIESRELATIONSHIPS': {}
47 , 'ATTRIBUTES': {'STATUS': (4, 'STRING', 0, 'VALUE'), 'ISSUEDDATE': (5,
48 'STRING', 0, 'VALUE'), 'DOUBLELINKED': (8, 'BOOL', 1, 'VALUE'), 'BASEUUID': (1,
49 'STRING', 0, 'VALUE'), 'WEIGHT': (6, 'STRING', 0, 'VALUE'), 'BASETYPE': (3,
50 'STRING', 0, 'VALUE'), 'CATEGORY': (7, 'STRING', 0, 'VALUE'), 'ISACTIVE': (0,
51 'BOOL', 1, 'VALUE'), 'TOKENENTITYKEY': (2, 'STRING', 0, 'VALUE')}
52 }
53 }
54
55

```

56 **[00319]** FIGURE 20 shows a block diagram illustrating example MCB-Platform
57 component configurations in some embodiments of the MCB-Platform. In some
58 embodiments, the MCB-Platform may aggregate data from a variety of sources to

1 generate centralized personal information. The may also aggregate various types of data
2 in order to generate the centralized personal information. For example, the MCB-
3 Platform may utilize search results aggregation component(s) 2001 (e.g., such as
4 described in FIGS. 21-22) to aggregate search results from across a wide range of
5 computer networked systems, e.g., the Internet. As another example, the MCB-Platform
6 may utilize transaction data aggregation component(s) 2002 (e.g., such as described in
7 FIGS. 23-26) to aggregate transaction data, e.g., from transaction processing procedure
8 by a payment network. As another example, the MCB-Platform may utilize service
9 usage data aggregation component(s) 2003 (e.g., such as described in FIGS. 23-26) to
10 aggregate data on user's usage of various services associated with the MCB-Platform. As
11 another example, the MCB-Platform may utilize enrollment data component(s) 2004
12 (e.g., such as described in FIGS. 23-26) to aggregate data on user's enrollment into
13 various services associated with the MCB-Platform. As another example, the MCB-
14 Platform may utilize social data aggregation component(s) 2003 (e.g., such as described
15 in FIGS. 27-28) to aggregate data on user's usage of various social networking services
16 accessible by the MCB-Platform.

17 **[00320]** In some embodiments, the MCB-Platform may acquire the aggregated
18 data, and normalize the data into formats that are suitable for uniform storage,
19 indexing, maintenance, and/or further processing via data record normalization
20 component(s) 2006 (e.g., such as described in FIG. 31). The MCB-Platform may extract
21 data from the normalized data records, and recognize data fields, e.g., the MCB-
22 Platform may identify the attributes of each field of data included in the normalized data
23 records via data field recognition component(s) 2007 (e.g., such as described in FIG.
24 32). For example, the MCB-Platform may identify names, user ID(s), addresses,

1 network addresses, comments and/or specific words within the comments, images, blog
2 posts, video, content within the video, and/or the like from the aggregated data. In
3 some embodiments, for each field of data, the MCB-Platform may classify entity types
4 associated with the field of data, as well as entity identifiers associated with the field of
5 data, e.g., via component(s) 2008 (e.g., such as described in FIG. 33). For example, the
6 MCB-Platform may identify an Internet Protocol (IP) address data field to be associated
7 with a user ID john.q.public (consumer entity type), a user John Q. Public (consumer
8 entity type), a household (the Public household - a multi-consumer entity type /
9 household entity type), a merchant entity type with identifier Acme Merchant Store, Inc.
10 from which purchases are made from the IP address, an Issuer Bank type with identifier
11 First National Bank associated with the purchases made from the IP address, and/or the
12 like. In some embodiments, the MCB-Platform may utilize the entity types and entity
13 identifiers to correlate entities across each other, e.g., via cross-entity correlation
14 component(s) 2009 (e.g., such as described in FIG. 34). For example, the MCB-
15 Platform may identify, from the aggregated data, that a household entity with identifier
16 H123 may include a user entity with identifier John Q. Public and social identifier
17 john.q.public@facebook.com, a second user entity with identifier Jane P. Doe with
18 social identifier jpdoe@twitter.com, a computer entity with identifier IP address
19 192.168.4.5, a card account entity with identifier ****1234, a bank issuer entity with
20 identifier AB23145, a merchant entity with identifier Acme Stores, Inc. where the
21 household sub-entities make purchases, and/or the like. In some embodiments, the
22 MCB-Platform may utilize the entity identifiers, data associated with each entity and/or
23 correlated entities to identify associations to other entities, e.g., via entity attribute
24 association component(s) 2010 (e.g., such as described in FIG. 35). For example, the

1 MCB-Platform may identify specific purchases made via purchase transactions by
2 members of the household, and thereby identify attributes of members of the household
3 on the basis of the purchases in the purchase transactions made by members of the
4 household. Based on such correlations and associations, the MCB-Platform may update
5 a profile for each entity identified from the aggregated data, as well as a social graph
6 interrelating the entities identified in the aggregated data, e.g., via entity profile-graph
7 updating component(s) 2011 (e.g., such as described in FIG. 36). In some
8 embodiments, the updating of profile and/or social graphs for an entity may trigger a
9 search for additional data that may be relevant to the newly identified correlations and
10 associations for each entity, e.g., via search term generation component(s) 2013-2014
11 (e.g., such as described in FIG. 37). For example, the updating of a profile and/or social
12 graph may trigger searches across the Internet, social networking websites, transaction
13 data from payment networks, services enrolled into and/or utilized by the entities,
14 and/or the like. In some embodiments, such updating of entity profiles and/or social
15 graphs may be performed continuously, periodically, on-demand, and/or the like.

16 **[00321]** FIGURE 21 shows a data flow diagram illustrating an example search
17 result aggregation procedure in some embodiments of the MCB-Platform. In some
18 implementations, the pay network server may obtain a trigger to perform a search. For
19 example, the pay network server may periodically perform a search update of its
20 aggregated search database, e.g., 2110, with new information available from a variety of
21 sources, such as the Internet. As another example, a request for on-demand search
22 update may be obtained as a result of a user wishing to enroll in a service, for which the
23 pay network server may facilitate data entry by providing an automated web form filling
24 system using information about the user obtained from the search update. In some

1 implementations, the pay network server may parse the trigger to extract keywords
 2 using which to perform an aggregated search. The pay network server may generate a
 3 query for application programming interface (API) templates for various search engines
 4 (e.g., Google™, Bing®, AskJeeves, market data search engines, etc.) from which to
 5 collect data for aggregation. The pay network server may query, e.g., 2112, a pay
 6 network database, e.g., 2107, for search API templates for the search engines. For
 7 example, the pay network server may utilize PHP/SQL commands similar to the
 8 examples provided above. The database may provide, e.g., 2113, a list of API templates
 9 in response. Based on the list of API templates, the pay network server may generate
 10 search requests, e.g., 2114. The pay network server may issue the generated search
 11 requests, e.g., 2115a-c, to the search engine servers, e.g., 2101a-c. For example, the pay
 12 network server may issue PHP commands to request the search engine for search
 13 results. An example listing of commands to issue search requests 2115a-c, substantially
 14 in the form of PHP commands, is provided below:

```

15  <?PHP
16  // API URL with access key
17  $url = ["https://ajax.googleapis.com/ajax/services/search/web?v=1.0&"
18         . "q=" $keywords "&key=1234567890987654&userip=datagraph.cpip.com"];
19
20  // Send Search Request
21  $ch = curl_init();
22  curl_setopt($ch, CURLOPT_URL, $url);
23  curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
24  curl_setopt($ch, CURLOPT_REFERER, "datagraph.cpip.com");
25  $body = curl_exec($ch);
26  curl_close($ch);
27
28  // Obtain, parse search results
29  $json = json_decode($body);
30  ?>
31
32

```

33 **[00322]** In some embodiments, the search engine servers may query, e.g., 2117a-c,
 34 their search databases, e.g., 2102a-c, for search results falling within the scope of the
 35 search keywords. In response to the search queries, the search databases may provide

1 search results, e.g., 2118a-c, to the search engine servers. The search engine servers may
 2 return the search results obtained from the search databases, e.g., 2119a-c, to the pay
 3 network server making the search requests. An example listing of search results 2119a-
 4 c, substantially in the form of JavaScript Object Notation (JSON)-formatted data, is
 5 provided below:

```

6  {"responseData": {
7    "results": [
8      {
9        "GsearchResultClass": "GwebSearch",
10       "unescapedUrl": "http://en.wikipedia.org/wiki/John_Q_Public",
11       "url": "http://en.wikipedia.org/wiki/John_Q_Public",
12       "visibleUrl": "en.wikipedia.org",
13       "cacheUrl":
14       "http://www.google.com/search?q\u003dcache:TwrPfhd22hYJ:en.wikipedia.org",
15       "title": "\u003cb\u003eJohn Q. Public\u003c/b\u003e - Wikipedia, the free
16       encyclopedia",
17       "titleNoFormatting": "John Q. Public - Wikipedia, the free encyclopedia",
18       "content": "\[1\] In 2006, he served as Chief Technology Officer..."
19     },
20     {
21       "GsearchResultClass": "GwebSearch",
22       "unescapedUrl": "http://www.imdb.com/name/nm0385296/",
23       "url": "http://www.imdb.com/name/nm0385296/",
24       "visibleUrl": "www.imdb.com",
25       "cacheUrl":
26       "http://www.google.com/search?q\u003dcache:li34KkqnssooJ:www.imdb.com",
27       "title": "\u003cb\u003eJohn Q. Public\u003c/b\u003e",
28       "titleNoFormatting": "John Q. Public",
29       "content": "Self: Zolander. Socialite \u003cb\u003eJohn Q.
30       Public\u003c/b\u003e..."
31     },
32     ...
33   ],
34   "cursor": {
35     "pages": [
36       { "start": "0", "label": 1 },
37       { "start": "4", "label": 2 },
38       { "start": "8", "label": 3 },
39       { "start": "12", "label": 4 }
40     ],
41     "estimatedResultCount": "59600000",
42     "currentPageIndex": 0,
43     "moreResultsUrl":
44     "http://www.google.com/search?oe\u003dutf8\u0026ie\u003dutf8..."
45   }
46 },
47 "responseDetails": null, "responseStatus": 200}
48
49
```

50 **[00323]** In some embodiments, the pay network server may store the aggregated
 51 search results, e.g., 2120, in an aggregated search database, e.g., 2110.

1 **[00324]** FIGURE 22 shows a logic flow diagram illustrating example aspects of
2 aggregating search results in some embodiments of the MCB-Platform, e.g., a Search
3 Results Aggregation (“SRA”) component 2200. In some implementations, the pay
4 network server may obtain a trigger to perform a search, e.g., 2201. For example, the
5 pay network server may periodically perform a search update of its aggregated search
6 database with new information available from a variety of sources, such as the Internet.
7 As another example, a request for on-demand search update may be obtained as a result
8 of a user wishing to enroll in a service, for which the pay network server may facilitate
9 data entry by providing an automated web form filling system using information about
10 the user obtained from the search update. In some implementations, the pay network
11 server may parse the trigger, e.g., 2202, to extract keywords using which to perform an
12 aggregated search. The pay network server may determine the search engines to search,
13 e.g., 2203, using the extracted keywords. Then, the pay network server may generate a
14 query for application programming interface (API) templates for the various search
15 engines (e.g., Google™, Bing®, AskJeeves, market data search engines, etc.) from which
16 to collect data for aggregation, e.g., 2204. The pay network server may query, e.g., 2205,
17 a pay network database for search API templates for the search engines. For example,
18 the pay network server may utilize PHP/SQL commands similar to the examples
19 provided above. The database may provide, e.g., 2205, a list of API templates in
20 response. Based on the list of API templates, the pay network server may generate
21 search requests, e.g., 2206. The pay network server may issue the generated search
22 requests to the search engine servers. The search engine servers may parse the obtained
23 search results(s), e.g., 2207, and query, e.g., 2208, their search databases for search
24 results falling within the scope of the search keywords. In response to the search

1 queries, the search databases may provide search results, e.g., 2209, to the search
2 engine servers. The search engine servers may return the search results obtained from
3 the search databases, e.g., 2210, to the pay network server making the search requests.
4 The pay network server may generate, e.g., 2211, and store the aggregated search results,
5 e.g., 2212, in an aggregated search database.

6 **[00325]** FIGURES 23A-D show data flow diagrams illustrating an example card-
7 based transaction execution procedure in some embodiments of the MCB-Platform. In
8 some implementations, a user, e.g., 2301, may desire to purchase a product, service,
9 offering, and/or the like (“product”), from a merchant. The user may communicate with
10 a merchant server, e.g., 2303, via a client such as, but not limited to: a personal
11 computer, mobile device, television, point-of-sale terminal, kiosk, ATM, and/or the like
12 (e.g., 2302). For example, the user may provide user input, e.g., purchase input 2311,
13 into the client indicating the user’s desire to purchase the product. In various
14 implementations, the user input may include, but not be limited to: keyboard entry, card
15 swipe, activating a RFID/NFC enabled hardware device (e.g., electronic card having
16 multiple accounts, smartphone, tablet, etc.), mouse clicks, depressing buttons on a
17 joystick/game console, voice commands, single/multi-touch gestures on a touch-
18 sensitive interface, touching user interface elements on a touch-sensitive display, and/or
19 the like. For example, the user may direct a browser application executing on the client
20 device to a website of the merchant, and may select a product from the website via
21 clicking on a hyperlink presented to the user via the website. As another example, the
22 client may obtain track 1 data from the user’s card (e.g., credit card, debit card, prepaid
23 card, charge card, etc.), such as the example track 1 data provided below:

1 %B123456789012345^PUBLIC/J.Q.^99011200000000000000**901*****?
 2 (wherein '123456789012345' is the card number of 'J.Q. Public' and has a CVV
 3 number of 901. '990112' is a service code, and *** represents decimal digits
 4 which change randomly each time the card is used.)
 5
 6

7 **[00326]** In some implementations, the client may generate a purchase order
 8 message, e.g., 2312, and provide, e.g., 2313, the generated purchase order message to
 9 the merchant server. For example, a browser application executing on the client may
 10 provide, on behalf of the user, a (Secure) Hypertext Transfer Protocol ("HTTP(S)") GET
 11 message including the product order details for the merchant server in the form of data
 12 formatted according to the eXtensible Markup Language ("XML"). Below is an example
 13 HTTP(S) GET message including an XML-formatted purchase order message for the
 14 merchant server:

```

15  GET /purchase.php HTTP/1.1
16  Host: www.merchant.com
17  Content-Type: Application/XML
18  Content-Length: 1306
19  <?XML version = "1.0" encoding = "UTF-8"?>
20  <purchase_order>
21      <order_ID>4NFU4RG94</order_ID>
22      <timestamp>2011-02-22 15:22:43</timestamp>
23      <user_ID>john.q.public@gmail.com</user_ID>
24      <client_details>
25          <client_IP>192.168.23.126</client_IP>
26          <client_type>smartphone</client_type>
27          <client_model>HTC Hero</client_model>
28          <OS>Android 2.2</OS>
29          <app_installed_flag>>true</app_installed_flag>
30      </client_details>
31      <purchase_details>
32          <num_products>1</num_products>
33          <product>
34              <product_type>book</product_type>
35              <product_params>
36                  <product_title>XML for dummies</product_title>
37                  <ISBN>938-2-14-168710-0</ISBN>
38                  <edition>2nd ed.</edition>
39                  <cover>hardbound</cover>
40                  <seller>bestbuybooks</seller>
41              </product_params>
42              <quantity>1</quantity>
43          </product>
44      </purchase_details>
45      <account_params>
46          <account_name>John Q. Public</account_name>
47          <account_type>credit</account_type>
48          <account_num>123456789012345</account_num>

```

```

1         <billing_address>123 Green St., Norman, OK
2 98765</billing_address>
3         <phone>123-456-7809</phone>
4         <sign>/jpg</sign>
5         <confirm_type>email</confirm_type>
6         <contact_info>john.q.public@gmail.com</contact_info>
7     </account_params>
8     <shipping_info>
9         <shipping_address>same as billing</shipping_address>
10        <ship_type>expedited</ship_type>
11        <ship_carrier>FedEx</ship_carrier>
12        <ship_account>123-45-678</ship_account>
13        <tracking_flag>true</tracking_flag>
14        <sign_flag>false</sign_flag>
15    </shipping_info>
16 </purchase_order>
17
18

```

[00327] In some implementations, the merchant server may obtain the purchase order message from the client, and may parse the purchase order message to extract details of the purchase order from the user. The merchant server may generate a card query request, e.g., 2314 to determine whether the transaction can be processed. For example, the merchant server may attempt to determine whether the user has sufficient funds to pay for the purchase in a card account provided with the purchase order. The merchant server may provide the generated card query request, e.g., 2315, to an acquirer server, e.g., 2304. For example, the acquirer server may be a server of an acquirer financial institution (“acquirer”) maintaining an account of the merchant. For example, the proceeds of transactions processed by the merchant may be deposited into an account maintained by the acquirer. In some implementations, the card query request may include details such as, but not limited to: the costs to the user involved in the transaction, card account details of the user, user billing and/or shipping information, and/or the like. For example, the merchant server may provide a HTTP(S) POST message including an XML-formatted card query request similar to the example listing provided below:

```

35 POST /cardquery.php HTTP/1.1
36 Host: www.acquirer.com

```



```

1 Content-Type: Application/XML
2 Content-Length: 624
3 <?XML version = "1.0" encoding = "UTF-8"?>
4 <card_query_request>
5     <query_ID>VNEI39FK</query_ID>
6     <timestamp>2011-02-22 15:22:44</timestamp>
7     <purchase_summary>
8         <num_products>1</num_products>
9         <product>
10             <product_summary>Book - XML for
11 dummies</product_summary>
12             <product_quantity>1</product_quantity?
13         </product>
14     </purchase_summary>
15     <transaction_cost>$34.78</transaction_cost>
16     <account_params>
17         <account_name>John Q. Public</account_name>
18         <account_type>credit</account_type>
19         <account_num>123456789012345</account_num>
20         <billing_address>123 Green St., Norman, OK
21 98765</billing_address>
22         <phone>123-456-7809</phone>
23         <sign>/jpg</sign>
24     </account_params>
25     <merchant_params>
26         <merchant_id>3FBCR4INC</merchant_id>
27         <merchant_name>Books & Things, Inc.</merchant_name>
28         <merchant_auth_key>1NNF484MCP59CHB27365</merchant_auth_key>
29     </merchant_params>
30 </card_query_request>
31
32

```

33 **[00328]** In some implementations, the acquirer server may generate a card
34 authorization request, e.g., 2316, using the obtained card query request, and provide the
35 card authorization request, e.g., 2317, to a pay network server, e.g., 2305. For example,
36 the acquirer server may redirect the HTTP(S) POST message in the example above from
37 the merchant server to the pay network server.

38 **[00329]** In some implementations, the pay network server may determine whether
39 the user has enrolled in value-added user services. For example, the pay network server
40 may query 2318 a database, e.g., pay network database 2307, for user service enrollment
41 data. For example, the server may utilize PHP/SQL commands similar to the example
42 provided above to query the pay network database. In some implementations, the
43 database may provide the user service enrollment data, e.g., 2319. The user enrollment
44 data may include a flag indicating whether the user is enrolled or not, as well as

1 instructions, data, login URL, login API call template and/or the like for facilitating
 2 access of the user-enrolled services. For example, in some implementations, the pay
 3 network server may redirect the client to a value-add server (e.g., such as a social
 4 network server where the value-add service is related to social networking) by providing
 5 a HTTP(S) REDIRECT 300 message, similar to the example below:

```

6  HTTP/1.1 300 Multiple Choices
7  Location:
8  https://www.facebook.com/dialog/oauth?client_id=snpa_app_ID&redirect_uri=
9  www.paynetwork.com/purchase.php
10 <html>
11     <head><title>300 Multiple Choices</title></head>
12     <body><h1>Multiple Choices</h1></body>
13 </html>
14
15

```

16 **[00330]** In some implementations, the pay network server may provide payment
 17 information extracted from the card authorization request to the value-add server as
 18 part of a value add service request, e.g., 2320. For example, the pay network server may
 19 provide a HTTP(S) POST message to the value-add server, similar to the example below:

```

20 POST /valueservices.php HTTP/1.1
21 Host: www.valueadd.com
22 Content-Type: Application/XML
23 Content-Length: 1306
24 <?XML version = "1.0" encoding = "UTF-8"?>
25 <service_request>
26     <request_ID>4NFU4RG94</order_ID>
27     <timestamp>2011-02-22 15:22:43</timestamp>
28     <user_ID>john.q.public@gmail.com</user_ID>
29     <client_details>
30         <client_IP>192.168.23.126</client_IP>
31         <client_type>smartphone</client_type>
32         <client_model>HTC Hero</client_model>
33         <OS>Android 2.2</OS>
34         <app_installed_flag>true</app_installed_flag>
35     </client_details>
36     <account_params>
37         <account_name>John Q. Public</account_name>
38         <account_type>credit</account_type>
39         <account_num>123456789012345</account_num>
40         <billing_address>123 Green St., Norman, OK
41 98765</billing_address>
42         <phone>123-456-7809</phone>
43         <sign>jqp</sign>
44         <confirm_type>email</confirm_type>
45         <contact_info>john.q.public@gmail.com</contact_info>
46     </account_params>
47     <!--optional-->

```

```

1      <merchant>
2          <merchant_id>CQN3Y42N</merchant_id>
3          <merchant_name>Acme Tech, Inc.</merchant_name>
4          <user_name>john.q.public</user_name>
5          <cardlist> www.acme.com/user/john.q.public/cclist.xml<cardlist>
6          <user_account_preference>1 3 2 4 7 6 5<user_account_preference>
7      </merchant>
8  </service_request>
9
10

```

[00331] In some implementations, the value-add server may provide a service input request, e.g., 2321, to the client. For example, the value-add server may provide a HTML input/login form to the client. The client may display, e.g., 2322, the login form for the user. In some implementations, the user may provide login input into the client, e.g., 2323, and the client may generate a service input response, e.g., 2324, for the value-add server. In some implementations, the value-add server may provide value-add services according to user value-add service enrollment data, user profile, etc., stored on the value-add server, and based on the user service input. Based on the provision of value-add services, the value-add server may generate a value-add service response, e.g., 2326, and provide the response to the pay network server. For example, the value-add server may provide a HTTP(S) POST message similar to the example below:

```

22  POST /serviceresponse.php HTTP/1.1
23  Host: www.paynet.com
24  Content-Type: Application/XML
25  Content-Length: 1306
26  <?XML version = "1.0" encoding = "UTF-8"?>
27  <service_response>
28      <request_ID>4NFU4RG94</order_ID>
29      <timestamp>2011-02-22 15:22:43</timestamp>
30      <result>serviced</result>
31      <servcode>943528976302-45569-003829-04</servcode>
32  </service_response>
33
34

```

[00332] In some implementations, upon receiving the value-add service response from the value-add server, the pay network server may extract the enrollment service data from the response for addition to a transaction data record. In some implementations, the pay network server may forward the card authorization request to

1 an appropriate pay network server, e.g., 2328, which may parse the card authorization
2 request to extract details of the request. Using the extracted fields and field values, the
3 pay network server may generate a query, e.g., 2329, for an issuer server corresponding
4 to the user's card account. For example, the user's card account, the details of which the
5 user may have provided via the client-generated purchase order message, may be linked
6 to an issuer financial institution ("issuer"), such as a banking institution, which issued
7 the card account for the user. An issuer server, e.g., 2308a-n, of the issuer may
8 maintain details of the user's card account. In some implementations, a database, e.g.,
9 pay network database 2307, may store details of the issuer servers and card account
10 numbers associated with the issuer servers. For example, the database may be a
11 relational database responsive to Structured Query Language ("SQL") commands. The
12 pay network server may execute a hypertext preprocessor ("PHP") script including SQL
13 commands to query the database for details of the issuer server. An example PHP/SQL
14 command listing, illustrating substantive aspects of querying the database, is provided
15 below:

```
16 <?PHP
17 header('Content-Type: text/plain');
18 mysql_connect("254.93.179.112",$DBserver,$password); // access database server
19 mysql_select_db("ISSUERS.SQL"); // select database table to search
20 //create query for issuer server data
21 $query = "SELECT issuer_name issuer_address issuer_id ip_address mac_address
22 auth_key port_num security_settings_list FROM IssuerTable WHERE account_num
23 LIKE '%" $accountnum";
24 $result = mysql_query($query); // perform the search query
25 mysql_close("ISSUERS.SQL"); // close database access
26 ?>
27
28
```

29 **[00333]** In response to obtaining the issuer server query, e.g., 2329, the pay
30 network database may provide, e.g., 2330, the requested issuer server data to the pay
31 network server. In some implementations, the pay network server may utilize the issuer
32 server data to generate a forwarding card authorization request, e.g., 2331, to redirect

1 the card authorization request from the acquirer server to the issuer server. The pay
2 network server may provide the card authorization request, e.g., 2332a-n, to the issuer
3 server. In some implementations, the issuer server, e.g., 2308a-n, may parse the card
4 authorization request, and based on the request details may query 2333a-n database,
5 e.g., user profile database 2309a-n, for data of the user's card account. For example, the
6 issuer server may issue PHP/SQL commands similar to the example provided below:

```
7  <?PHP
8  header('Content-Type: text/plain');
9  mysql_connect("254.93.179.112",$DBserver,$password); // access database server
10 mysql_select_db("USERS.SQL"); // select database table to search
11 //create query for user data
12 $query = "SELECT user_id user_name user_balance account_type FROM UserTable
13 WHERE account_num LIKE '%" . $accountnum . "';
14 $result = mysql_query($query); // perform the search query
15 mysql_close("USERS.SQL"); // close database access
16 ?>
```

17
18 **[00334]** In some implementations, on obtaining the user data, e.g., 2334a-n, the
19 issuer server may determine whether the user can pay for the transaction using funds
20 available in the account, e.g., 2335a-n. For example, the issuer server may determine
21 whether the user has a sufficient balance remaining in the account, sufficient credit
22 associated with the account, and/or the like. If the issuer server determines that the
23 user can pay for the transaction using the funds available in the account, the server may
24 provide an authorization message, e.g., 2336a-n, to the pay network server. For
25 example, the server may provide a HTTP(S) POST message similar to the examples
26 above.

27
28 **[00335]** In some implementations, the pay network server may obtain the
29 authorization message, and parse the message to extract authorization details. Upon
30 determining that the user possesses sufficient funds for the transaction, the pay network
31 server may generate a transaction data record from the card authorization request it

1 received, and store, e.g., 2339, the details of the transaction and authorization relating
 2 to the transaction in a database, e.g., pay network database 2307. For example, the pay
 3 network server may issue PHP/SQL commands similar to the example listing below to
 4 store the transaction data in a database:

```

5  <?PHP
6  header('Content-Type: text/plain');
7  mysql_connect("254.92.185.103",$DBserver,$password); // access database server
8  mysql_select("TRANSACTIONS.SQL"); // select database to append
9  mysql_query("INSERT INTO PurchasesTable (timestamp, purchase_summary_list,
10 num_products, product_summary, product_quantity, transaction_cost,
11 account_params_list, account_name, account_type, account_num, billing_addres,
12 zipcode, phone, sign, merchant_params_list, merchant_id, merchant_name,
13 merchant_auth_key)
14 VALUES (time(), $purchase_summary_list, $num_products, $product_summary,
15 $product_quantity, $transaction_cost, $account_params_list, $account_name,
16 $account_type, $account_num, $billing_addres, $zipcode, $phone, $sign,
17 $merchant_params_list, $merchant_id, $merchant_name, $merchant_auth_key)"); //
18 add data to table in database
19 mysql_close("TRANSACTIONS.SQL"); // close connection to database
20 ?>
21
22

```

23 **[00336]** In some implementations, the pay network server may forward the
 24 authorization message, e.g., 2340, to the acquirer server, which may in turn forward the
 25 authorization message, e.g., 2340, to the merchant server. The merchant may obtain
 26 the authorization message, and determine from it that the user possesses sufficient
 27 funds in the card account to conduct the transaction. The merchant server may add a
 28 record of the transaction for the user to a batch of transaction data relating to
 29 authorized transactions. For example, the merchant may append the XML data
 30 pertaining to the user transaction to an XML data file comprising XML data for
 31 transactions that have been authorized for various users, e.g., 2341, and store the XML
 32 data file, e.g., 2342, in a database, e.g., merchant database 2304. For example, a batch
 33 XML data file may be structured similar to the example XML data structure template
 34 provided below:

```

35  <?XML version = "1.0" encoding = "UTF-8"?>

```

```

1  <merchant_data>
2      <merchant_id>3FBCR4INC</merchant_id>
3      <merchant_name>Books & Things, Inc.</merchant_name>
4      <merchant_auth_key>1NNF484MCP59CHB27365</merchant_auth_key>
5      <account_number>123456789</account_number>
6  </merchant_data>
7  <transaction_data>
8      <transaction 1>
9          ...
10     </transaction 1>
11     <transaction 2>
12         ...
13     </transaction 2>
14     .
15     .
16     .
17     <transaction n>
18         ...
19     </transaction n>
20 </transaction_data>
21
22

```

[00337] In some implementations, the server may also generate a purchase receipt, e.g., 2343, and provide the purchase receipt to the client. The client may render and display, e.g., 2344, the purchase receipt for the user. For example, the client may render a webpage, electronic message, text / SMS message, buffer a voicemail, emit a ring tone, and/or play an audio message, etc., and provide output including, but not limited to: sounds, music, audio, video, images, tactile feedback, vibration alerts (e.g., on vibration-capable client devices such as a smartphone etc.), and/or the like.

[00338] With reference to FIGURE 23C, in some implementations, the merchant server may initiate clearance of a batch of authorized transactions. For example, the merchant server may generate a batch data request, e.g., 2345, and provide the request, e.g., 2346, to a database, e.g., merchant database 2304. For example, the merchant server may utilize PHP/SQL commands similar to the examples provided above to query a relational database. In response to the batch data request, the database may provide the requested batch data, e.g., 2347. The server may generate a batch clearance request, e.g., 2348, using the batch data obtained from the database, and provide, e.g., 2341, the

1 batch clearance request to an acquirer server, e.g., 2310. For example, the merchant
2 server may provide a HTTP(S) POST message including XML-formatted batch data in
3 the message body for the acquirer server. The acquirer server may generate, e.g., 2350,
4 a batch payment request using the obtained batch clearance request, and provide the
5 batch payment request to the pay network server, e.g., 2351. The pay network server
6 may parse the batch payment request, and extract the transaction data for each
7 transaction stored in the batch payment request, e.g., 2352. The pay network server
8 may store the transaction data, e.g., 2353, for each transaction in a database, e.g., pay
9 network database 2307. For each extracted transaction, the pay network server may
10 query, e.g., 2354-2355, a database, e.g., pay network database 2307, for an address of an
11 issuer server. For example, the pay network server may utilize PHP/SQL commands
12 similar to the examples provided above. The pay network server may generate an
13 individual payment request, e.g., 2356, for each transaction for which it has extracted
14 transaction data, and provide the individual payment request, e.g., 2357, to the issuer
15 server, e.g., 2308. For example, the pay network server may provide a HTTP(S) POST
16 request similar to the example below:

```
17 POST /requestpay.php HTTP/1.1
18 Host: www.issuer.com
19 Content-Type: Application/XML
20 Content-Length: 788
21 <?XML version = "1.0" encoding = "UTF-8"?>
22 <pay_request>
23     <request_ID>CNI4ICNW2</request_ID>
24     <timestamp>2011-02-22 17:00:01</timestamp>
25     <pay_amount>$34.78</pay_amount>
26     <account_params>
27         <account_name>John Q. Public</account_name>
28         <account_type>credit</account_type>
29         <account_num>123456789012345</account_num>
30         <billing_address>123 Green St., Norman, OK
31 98765</billing_address>
32         <phone>123-456-7809</phone>
33         <sign>/jqp</sign>
34     </account_params>
35     <merchant_params>
36         <merchant_id>3FBCR4INC</merchant_id>
```



```

1         <merchant_name>Books & Things, Inc.</merchant_name>
2         <merchant_auth_key>1NNF484MCP59CHB27365</merchant_auth_key>
3     </merchant_params>
4     <purchase_summary>
5         <num_products>1</num_products>
6         <product>
7             <product_summary>Book - XML for
8 dummies</product_summary>
9             <product_quantity>1</product_quantity>
10        </product>
11    </purchase_summary>
12 </pay_request>
13
14

```

15 **[00339]** In some implementations, the issuer server may generate a payment
16 command, e.g., 2358. For example, the issuer server may issue a command to deduct
17 funds from the user's account (or add a charge to the user's credit card account). The
18 issuer server may issue a payment command, e.g., 2359, to a database storing the user's
19 account information, e.g., user profile database 2308. The issuer server may provide a
20 funds transfer message, e.g., 2360, to the pay network server, which may forward, e.g.,
21 2361, the funds transfer message to the acquirer server. An example HTTP(S) POST
22 funds transfer message is provided below:

```

23 POST /clearance.php HTTP/1.1
24 Host: www.acquirer.com
25 Content-Type: Application/XML
26 Content-Length: 206
27 <?XML version = "1.0" encoding = "UTF-8"?>
28 <deposit_ack>
29     <request_ID>CNI4ICNW2</request_ID>
30     <clear_flag>true</clear_flag>
31     <timestamp>2011-02-22 17:00:02</timestamp>
32     <deposit_amount>$34.78</deposit_amount>
33 </deposit_ack>
34
35

```

36 **[00340]** In some implementations, the acquirer server may parse the funds
37 transfer message, and correlate the transaction (e.g., using the request_ID field in the
38 example above) to the merchant. The acquirer server may then transfer the funds
39 specified in the funds transfer message to an account of the merchant, e.g., 2362.

1 **[00341]** FIGURES 24A-E show logic flow diagrams illustrating example aspects of
2 card-based transaction execution, resulting in generation of card-based transaction data
3 and service usage data, in some embodiments of the MCB-Platform, e.g., a Card-Based
4 Transaction Execution (“CTE”) component 2400. In some implementations, a user may
5 provide user input, e.g., 2401, into a client indicating the user’s desire to purchase a
6 product from a merchant. The client may generate a purchase order message, e.g.,
7 2402, and provide the generated purchase order message to the merchant server. In
8 some implementations, the merchant server may obtain, e.g., 2403, the purchase order
9 message from the client, and may parse the purchase order message to extract details of
10 the purchase order from the user. Example parsers that the merchant client may utilize
11 are discussed further below with reference to FIGURE 51. The merchant may generate a
12 product data query, e.g., 2404, for a merchant database, which may in response provide
13 the requested product data, e.g., 2405. The merchant server may generate a card query
14 request using the product data, e.g., 2404, to determine whether the transaction can be
15 processed. For example, the merchant server may process the transaction only if the
16 user has sufficient funds to pay for the purchase in a card account provided with the
17 purchase order. The merchant server may optionally provide the generated card query
18 request to an acquirer server. The acquirer server may generate a card authorization
19 request using the obtained card query request, and provide the card authorization
20 request to a pay network server.

21 **[00342]** In some implementations, the pay network server may determine whether
22 the user has enrolled in value-added user services. For example, the pay network server
23 may query a database, e.g., 2407, for user service enrollment data. For example, the
24 server may utilize PHP/SQL commands similar to the example provided above to query

1 the pay network database. In some implementations, the database may provide the user
2 service enrollment data, e.g., 2408. The user enrollment data may include a flag
3 indicating whether the user is enrolled or not, as well as instructions, data, login URL,
4 login API call template and/or the like for facilitating access of the user-enrolled
5 services. For example, in some implementations, the pay network server may redirect
6 the client to a value-add server (e.g., such as a social network server where the value-add
7 service is related to social networking) by providing a HTTP(S) REDIRECT 300
8 message. In some implementations, the pay network server may provide payment
9 information extracted from the card authorization request to the value-add server as
10 part of a value add service request, e.g., 2410.

11 **[00343]** In some implementations, the value-add server may provide a service
12 input request, e.g., 2411, to the client. The client may display, e.g., 2412, the input
13 request for the user. In some implementations, the user may provide input into the
14 client, e.g., 2413, and the client may generate a service input response for the value-add
15 server. In some implementations, the value-add server may provide value-add services
16 according to user value-add service enrollment data, user profile, etc., stored on the
17 value-add server, and based on the user service input. Based on the provision of value-
18 add services, the value-add server may generate a value-add service response, e.g., 2417,
19 and provide the response to the pay network server. In some implementations, upon
20 receiving the value-add service response from the value-add server, the pay network
21 server may extract the enrollment service data from the response for addition to a
22 transaction data record, e.g., 2419-2420.

1 **[00344]** With reference to FIGURE 24B, in some implementations, the pay
2 network server may obtain the card authorization request from the acquirer server, and
3 may parse the card authorization request to extract details of the request, e.g., 2420.
4 Using the extracted fields and field values, the pay network server may generate a query,
5 e.g., 2421-2422, for an issuer server corresponding to the user's card account. In
6 response to obtaining the issuer server query the pay network database may provide,
7 e.g., 2422, the requested issuer server data to the pay network server. In some
8 implementations, the pay network server may utilize the issuer server data to generate a
9 forwarding card authorization request, e.g., 2423, to redirect the card authorization
10 request from the acquirer server to the issuer server. The pay network server may
11 provide the card authorization request to the issuer server. In some implementations,
12 the issuer server may parse, e.g., 2424, the card authorization request, and based on the
13 request details may query a database, e.g., 2425, for data of the user's card account. In
14 response, the database may provide the requested user data. On obtaining the user
15 data, the issuer server may determine whether the user can pay for the transaction using
16 funds available in the account, e.g., 2426. For example, the issuer server may determine
17 whether the user has a sufficient balance remaining in the account, sufficient credit
18 associated with the account, and/or the like, but comparing the data from the database
19 with the transaction cost obtained from the card authorization request. If the issuer
20 server determines that the user can pay for the transaction using the funds available in
21 the account, the server may provide an authorization message, e.g., 2427, to the pay
22 network server.

23 **[00345]** In some implementations, the pay network server may obtain the
24 authorization message, and parse the message to extract authorization details. Upon

1 determining that the user possesses sufficient funds for the transaction (e.g., 2430,
2 option “Yes”), the pay network server may extract the transaction card from the
3 authorization message and/or card authorization request, e.g., 2433, and generate a
4 transaction data record using the card transaction details. The pay network server may
5 provide the transaction data record for storage, e.g., 2434, to a database. In some
6 implementations, the pay network server may forward the authorization message, e.g.,
7 2435, to the acquirer server, which may in turn forward the authorization message, e.g.,
8 2436, to the merchant server. The merchant may obtain the authorization message, and
9 parse the authorization message o extract its contents, e.g., 2437. The merchant server
10 may determine whether the user possesses sufficient funds in the card account to
11 conduct the transaction. If the merchant server determines that the user possess
12 sufficient funds, e.g., 2438, option “Yes,” the merchant server may add the record of the
13 transaction for the user to a batch of transaction data relating to authorized
14 transactions, e.g., 2439-2440. The merchant server may also generate a purchase
15 receipt, e.g., 2441, for the user. If the merchant server determines that the user does not
16 possess sufficient funds, e.g., 2438, option “No,” the merchant server may generate an
17 “authorization fail” message, e.g., 2442. The merchant server may provide the purchase
18 receipt or the “authorization fail” message to the client. The client may render and
19 display, e.g., 2443, the purchase receipt for the user.

20 **[00346]** In some implementations, the merchant server may initiate clearance of a
21 batch of authorized transactions by generating a batch data request, e.g., 2444, and
22 providing the request to a database. In response to the batch data request, the database
23 may provide the requested batch data, e.g., 2445, to the merchant server. The server
24 may generate a batch clearance request, e.g., 2446, using the batch data obtained from

1 the database, and provide the batch clearance request to an acquirer server. The
2 acquirer server may generate, e.g., 2448, a batch payment request using the obtained
3 batch clearance request, and provide the batch payment request to a pay network server.
4 The pay network server may parse, e.g., 2449, the batch payment request, select a
5 transaction stored within the batch data, e.g., 2450, and extract the transaction data for
6 the transaction stored in the batch payment request, e.g., 2451. The pay network server
7 may generate a transaction data record, e.g., 2452, and store the transaction data, e.g.,
8 2453, the transaction in a database. For the extracted transaction, the pay network
9 server may generate an issuer server query, e.g., 2454, for an address of an issuer server
10 maintaining the account of the user requesting the transaction. The pay network server
11 may provide the query to a database. In response, the database may provide the issuer
12 server data requested by the pay network server, e.g., 2455. The pay network server may
13 generate an individual payment request, e.g., 2456, for the transaction for which it has
14 extracted transaction data, and provide the individual payment request to the issuer
15 server using the issuer server data from the database.

16 **[00347]** In some implementations, the issuer server may obtain the individual
17 payment request, and parse, e.g., 2457, the individual payment request to extract details
18 of the request. Based on the extracted data, the issuer server may generate a payment
19 command, e.g., 2458. For example, the issuer server may issue a command to deduct
20 funds from the user's account (or add a charge to the user's credit card account). The
21 issuer server may issue a payment command, e.g., 2459, to a database storing the user's
22 account information. In response, the database may update a data record
23 corresponding to the user's account to reflect the debit / charge made to the user's

1 account. The issuer server may provide a funds transfer message, e.g., 2460, to the pay
2 network server after the payment command has been executed by the database.

3 **[00348]** In some implementations, the pay network server may check whether
4 there are additional transactions in the batch that need to be cleared and funded. If
5 there are additional transactions, e.g., 2461, option “Yes,” the pay network server may
6 process each transaction according to the procedure described above. The pay network
7 server may generate, e.g., 2462, an aggregated funds transfer message reflecting transfer
8 of all transactions in the batch, and provide, e.g., 2463, the funds transfer message to
9 the acquirer server. The acquirer server may, in response, transfer the funds specified in
10 the funds transfer message to an account of the merchant, e.g., 2464.

11 **[00349]** FIGURE 25 shows a data flow diagram illustrating an example procedure
12 to aggregate card-based transaction data in some embodiments of the MCB-Platform.
13 In some implementations, the pay network server may determine a scope of data
14 aggregation required to perform the analysis, e.g., 2511. The pay network server may
15 initiate data aggregation based on the determined scope. The pay network server may
16 generate a query for addresses of server storing transaction data within the determined
17 scope. The pay network server may query, e.g., 2512, a pay network database, e.g.,
18 2507a, for addresses of pay network servers that may have stored transaction data
19 within the determined scope of the data aggregation. For example, the pay network
20 server may utilize PHP/SQL commands similar to the examples provided above. The
21 database may provide, e.g., 2513, a list of server addresses in response to the pay
22 network server’s query. Based on the list of server addresses, the pay network server
23 may generate transaction data requests, e.g., 2514. The pay network server may issue

1 the generated transaction data requests, e.g., 2515a-c, to the other pay network servers,
2 e.g., 2505b-d. The other pay network servers may query, e.g., 2517a-c, their pay network
3 database, e.g., 2507a-d, for transaction data falling within the scope of the transaction
4 data requests. In response to the transaction data queries, the pay network databases
5 may provide transaction data, e.g., 2518a-c, to the other pay network servers. The other
6 pay network servers may return the transaction data obtained from the pay network
7 databases, e.g., 2519a-c, to the pay network server making the transaction data requests,
8 e.g., 2505a. The pay network server, e.g., 2505a, may store the aggregated transaction
9 data, e.g., 2520, in an aggregated transactions database, e.g., 2510a.

10 **[00350]** FIGURE 26 shows a logic flow diagram illustrating example aspects of
11 aggregating card-based transaction data in some embodiments of the MCB-Platform,
12 e.g., a Transaction Data Aggregation (“TDA”) component 2600. In some
13 implementations, a pay network server may obtain a trigger to aggregate transaction
14 data, e.g., 2601. For example, the server may be configured to initiate transaction data
15 aggregation on a regular, periodic, basis (e.g., hourly, daily, weekly, monthly, quarterly,
16 semi-annually, annually, etc.). As another example, the server may be configured to
17 initiate transaction data aggregation on obtaining information that the U.S. Government
18 (e.g., Department of Commerce, Office of Management and Budget, etc) has released
19 new statistical data related to the U.S. business economy. As another example, the
20 server may be configured to initiate transaction data aggregation on-demand, upon
21 obtaining a user investment strategy analysis request for processing. The pay network
22 server may determine a scope of data aggregation required to perform the analysis, e.g.,
23 2602. For example, the scope of data aggregation may be pre-determined. As another
24 example, the scope of data aggregation may be determined based on a received user

1 investment strategy analysis request. The pay network server may initiate data
2 aggregation based on the determined scope. The pay network server may generate a
3 query for addresses of server storing transaction data within the determined scope, e.g.,
4 2603. The pay network server may query a database for addresses of pay network
5 servers that may have stored transaction data within the determined scope of the data
6 aggregation. The database may provide, e.g., 2604, a list of server addresses in response
7 to the pay network server's query. Based on the list of server addresses, the pay network
8 server may generate transaction data requests, e.g., 2605. The pay network server may
9 issue the generated transaction data requests to the other pay network servers. The
10 other pay network servers may obtain and parse the transaction data requests, e.g.,
11 2606. Based on parsing the data requests, the other pay network servers may generate
12 transaction data queries, e.g., 2607, and provide the transaction data queries to their
13 pay network databases. In response to the transaction data queries, the pay network
14 databases may provide transaction data, e.g., 2608, to the other pay network servers.
15 The other pay network servers may return, e.g., 2609, the transaction data obtained
16 from the pay network databases to the pay network server making the transaction data
17 requests. The pay network server may generate aggregated transaction data records
18 from the transaction data received from the other pay network servers, e.g., 2610, and
19 store the aggregated transaction data in a database, e.g., 2611.

20 **[00351]** FIGURE 27 shows a data flow diagram illustrating an example social data
21 aggregation procedure in some embodiments of the MCB-Platform. In some
22 implementations, the pay network server may obtain a trigger to perform a social data
23 search. For example, the pay network server may periodically perform an update of its
24 aggregated social database, e.g., 2710, with new information available from a variety of

1 sources, such as the social networking services operating on the Internet. As another
2 example, a request for on-demand social data update may be obtained as a result of a
3 user wishing to enroll in a service, for which the pay network server may facilitate data
4 entry by providing an automated web form filling system using information about the
5 user obtained from the social data update. In some implementations, the pay network
6 server may parse the trigger to extract keywords using which to perform an aggregated
7 social data update. The pay network server may generate a query for application
8 programming interface (API) templates for various social networking services (e.g.,
9 Facebook®, Twitter™, etc.) from which to collect social data for aggregation. The pay
10 network server may query, e.g., 2712, a pay network database, e.g., 2707, for social
11 network API templates for the social networking services. For example, the pay network
12 server may utilize PHP/SQL commands similar to the examples provided above. The
13 database may provide, e.g., 2713, a list of API templates in response. Based on the list of
14 API templates, the pay network server may generate social data requests, e.g., 2714. The
15 pay network server may issue the generated social data requests, e.g., 2715a-c, to the
16 social network servers, e.g., 2701a-c. For example, the pay network server may issue
17 PHP commands to request the social network servers for social data. An example listing
18 of commands to issue social data requests 2715a-c, substantially in the form of PHP
19 commands, is provided below:

```
20 <?PHP
21 header('Content-Type: text/plain');
22
23 // Obtain user ID(s) of friends of the logged-in user
24 $friends =
25 json_decode(file_get_contents('https://graph.facebook.com/me/friends?access
26 token='.$cookie['oauth_access_token']), true);
27 $friend_ids = array_keys($friends);
28
29 // Obtain message feed associated with the profile of the logged-in user
```

```

1  $feed =
2  json_decode(file_get_contents('https://graph.facebook.com/me/feed?access_token=
3  '$cookie['oauth_access_token']), true);
4
5  // Obtain messages by the user's friends
6  $result = mysql_query('SELECT * FROM content WHERE uid IN (
7  .implode($friend_ids, ',') . ')');
8  $friend_content = array();
9  while ($row = mysql_fetch_assoc($result))
10  $friend_content [] $row;
11  ?>
12
13

```

[00352] In some embodiments, the social network servers may query, e.g., 2717a-c, their databases, e.g., 2702a-c, for social data results falling within the scope of the social keywords. In response to the queries, the databases may provide social data, e.g., 2718a-c, to the search engine servers. The social network servers may return the social data obtained from the databases, e.g., 2719a-c, to the pay network server making the social data requests. An example listing of social data 2719a-c, substantially in the form of JavaScript Object Notation (JSON)-formatted data, is provided below:

```

21
22  [ "data": [
23      {      "name": "Tabatha Orloff",
24              "id": "483722"},
25      {      "name": "Darren Kinnaman",
26              "id": "86S743"},
27      {      "name": "Sharron Jutras",
28              "id": "091274"}
29  ] }
30
31

```

[00353] In some embodiments, the pay network server may store the aggregated search results, e.g., 2720, in an aggregated search database, e.g., 2710.

[00354] FIGURE 28 shows a logic flow diagram illustrating example aspects of aggregating social data in some embodiments of the MCB-Platform, e.g., a Social Data Aggregation (“SDA”) component 2800. In some implementations, the pay network server may obtain a trigger to perform a social search, e.g., 2801. For example, the pay network server may periodically perform an update of its aggregated social database

1 with new information available from a variety of sources, such as the Internet. As
2 another example, a request for on-demand social data update may be obtained as a
3 result of a user wishing to enroll in a service, for which the pay network server may
4 facilitate data entry by providing an automated web form filling system using
5 information about the user obtained from the social data update. In some
6 implementations, the pay network server may parse the trigger, e.g., 2802, to extract
7 keywords and/or user ID(s) using which to perform an aggregated search for social data.
8 The pay network server may determine the social networking services to search, e.g.,
9 2803, using the extracted keywords and/or user ID(s). Then, the pay network server
10 may generate a query for application programming interface (API) templates for the
11 various social networking services (e.g., Facebook®, Twitter™, etc.) from which to
12 collect social data for aggregation, e.g., 2804. The pay network server may query, e.g.,
13 2805, a pay network database for search API templates for the social networking
14 services. For example, the pay network server may utilize PHP/SQL commands similar
15 to the examples provided above. The database may provide, e.g., 2805, a list of API
16 templates in response. Based on the list of API templates, the pay network server may
17 generate social data requests, e.g., 2806. The pay network server may issue the
18 generated social data requests to the social networking services. The social network
19 servers may parse the obtained search results(s), e.g., 2807, and query, e.g., 2808, their
20 databases for social data falling within the scope of the search keywords. In response to
21 the social data queries, the databases may provide social data, e.g., 2809, to the social
22 networking servers. The social networking servers may return the social data obtained
23 from the databases, e.g., 2810, to the pay network server making the social data

1 requests. The pay network server may generate, e.g., 2811, and store the aggregated
2 social data, e.g., 2812, in an aggregated social database.

3 **[00355]** FIGURE 29 shows a data flow diagram illustrating an example procedure
4 for enrollment in value-add services in some embodiments of the MCB-Platform. In
5 some implementations, a user, e.g., 2901, may desire to enroll in a value-added service.
6 Let us consider an example wherein the user desires to enroll in social network
7 authenticated purchase payment as a value-added service. It is to be understood that
8 any other value-added service may take the place of the below-described value-added
9 service. The user may communicate with a pay network server, e.g., 2903, via a client
10 such as, but not limited to: a personal computer, mobile device, television, point-of-sale
11 terminal, kiosk, ATM, and/or the like (e.g., 2902). For example, the user may provide
12 user input, e.g., enroll input 2911, into the client indicating the user's desire to enroll in
13 social network authenticated purchase payment. In various implementations, the user
14 input may include, but not be limited to: a single tap (e.g., a one-tap mobile app
15 purchasing embodiment) of a touchscreen interface, keyboard entry, card swipe,
16 activating a RFID/NFC enabled hardware device (e.g., electronic card having multiple
17 accounts, smartphone, tablet, etc.) within the user device, mouse clicks, depressing
18 buttons on a joystick/game console, voice commands, single/multi-touch gestures on a
19 touch-sensitive interface, touching user interface elements on a touch-sensitive display,
20 and/or the like. For example, the user may swipe a payment card at the client 2902. In
21 some implementations, the client may obtain track 1 data from the user's card as enroll
22 input 2911 (e.g., credit card, debit card, prepaid card, charge card, etc.), such as the
23 example track 1 data provided below:

1 %B123456789012345^PUBLIC/J.Q.^99011200000000000000**901*****?*
 2 (wherein '123456789012345' is the card number of 'J.Q. Public' and has a CVV
 3 number of 901. '990112' is a service code, and *** represents decimal digits
 4 which change randomly each time the card is used.)
 5
 6

7 **[00356]** In some implementations, using the user's input, the client may generate
 8 an enrollment request, e.g., 2912, and provide the enrollment request, e.g., 2913, to the
 9 pay network server. For example, the client may provide a (Secure) Hypertext Transfer
 10 Protocol ("HTTP(S)") POST message including data formatted according to the
 11 eXtensible Markup Language ("XML"). Below is an example HTTP(S) POST message
 12 including an XML-formatted enrollment request for the pay network server:

```

13  POST /enroll.php HTTP/1.1
14  Host: www.merchant.com
15  Content-Type: Application/XML
16  Content-Length: 718
17  <?XML version = "1.0" encoding = "UTF-8"?>
18  <enrollment_request>
19      <cart_ID>4NFU4RG94</order_ID>
20      <timestamp>2011-02-22 15:22:43</timestamp>
21      <user_ID>john.q.public@gmail.com</user_ID>
22      <client_details>
23          <client_IP>192.168.23.126</client_IP>
24          <client_type>smartphone</client_type>
25          <client_model>HTC Hero</client_model>
26          <OS>Android 2.2</OS>
27          <app_installed_flag>true</app_installed_flag>
28      </client_details>
29      <!--account_params> <optional>
30          <account_name>John Q. Public</account_name>
31          <account_type>credit</account_type>
32          <account_num>123456789012345</account_num>
33          <billing_address>123 Green St., Norman, OK
34  98765</billing_address>
35          <phone>123-456-7809</phone>
36          <sign>/jqp/</sign>
37          <confirm_type>email</confirm_type>
38          <contact_info>john.q.public@gmail.com</contact_info>
39      </account_params-->
40      <checkout_purchase_details>
41          <num_products>1</num_products>
42          <product>
43              <product_type>book</product_type>
44              <product_params>
45                  <product_title>XML for dummies</product_title>
46                  <ISBN>938-2-14-168710-0</ISBN>
47                  <edition>2nd ed.</edition>
48                  <cover>hardbound</cover>
49                  <seller>bestbuybooks</seller>
50              </product_params>
51              <quantity>1</quantity>
52          </product>

```

```

1         </checkout_purchase_details>
2     </enrollment_request>
3
4

```

[00357] In some implementations, the pay network server may obtain the enrollment request from the client, and extract the user's payment detail (e.g., XML data) from the enrollment request. For example, the pay network server may utilize a parser such as the example parsers described below in the discussion with reference to FIGURE 51. In some implementations, the pay network server may query, e.g., 2914, a pay network database, e.g., 2904, to obtain a social network request template, e.g., 2915, to process the enrollment request. The social network request template may include instructions, data, login URL, login API call template and/or the like for facilitating social network authentication. For example, the database may be a relational database responsive to Structured Query Language ("SQL") commands. The merchant server may execute a hypertext preprocessor ("PHP") script including SQL commands to query the database for product data. An example PHP/SQL command listing, illustrating substantive aspects of querying the database, e.g., 2914-2915, is provided below:

```

18     <?PHP
19     header('Content-Type: text/plain');
20     mysql_connect("254.93.179.112",$DBserver,$password); // access database server
21     mysql_select_db("SOCIALAUTH.SQL"); // select database table to search
22     //create query
23     $query = "SELECT template FROM EnrollTable WHERE network LIKE '%" . $socialnet . "';
24     $result = mysql_query($query); // perform the search query
25     mysql_close("SOCIALAUTH.SQL"); // close database access
26     ?>
27
28

```

[00358] In some implementations, the pay network server may redirect the client to a social network server by providing a HTTP(S) REDIRECT 300 message, similar to the example below:

```

32     HTTP/1.1 300 Multiple Choices
33     Location:
34     https://www.facebook.com/dialog/oauth?client_id=snpa_app_ID&redirect_uri=
35     www.paynetwork.com/enroll.php

```

```

1  <html>
2      <head><title>300 Multiple Choices</title></head>
3      <body><h1>Multiple Choices</h1></body>
4  </html>
5
6

```

7 **[00359]** In some implementations, the pay network server may provide payment
8 information extracted from the card authorization request to the social network server
9 as part of a social network authentication enrollment request, e.g., 2917. For example,
10 the pay network server may provide a HTTP(S) POST message to the social network
11 server, similar to the example below:

```

12  POST /authenticate_enroll.php HTTP/1.1
13  Host: www.socialnet.com
14  Content-Type: Application/XML
15  Content-Length: 1306
16  <?XML version = "1.0" encoding = "UTF-8"?>
17  <authenticate_enrollment_request>
18      <request_ID>4NFU4RG94</order_ID>
19      <timestamp>2011-02-22 15:22:43</timestamp>
20      <user_ID>john.q.public@gmail.com</user_ID>
21      <client_details>
22          <client_IP>192.168.23.126</client_IP>
23          <client_type>smartphone</client_type>
24          <client_model>HTC Hero</client_model>
25          <OS>Android 2.2</OS>
26          <app_installed_flag>true</app_installed_flag>
27      </client_details>
28      <account_params>
29          <account_name>John Q. Public</account_name>
30          <account_type>credit</account_type>
31          <account_num>123456789012345</account_num>
32          <billing_address>123 Green St., Norman, OK
33  98765</billing_address>
34          <phone>123-456-7809</phone>
35          <sign>/jq</sign>
36          <confirm_type>email</confirm_type>
37          <contact_info>john.q.public@gmail.com</contact_info>
38      </account_params>
39  </authenticate_enrollment_request>
40
41

```

42 **[00360]** In some implementations, the social network server may provide a social
43 network login request, e.g., 2918, to the client. For example, the social network server
44 may provide a HTML input form to the client. The client may display, e.g., 2919, the
45 login form for the user. In some implementations, the user may provide login input into
46 the client, e.g., 2920, and the client may generate a social network login response, e.g.,

1 2921, for the social network server. In some implementations, the social network server
2 may authenticate the login credentials of the user, and access payment account
3 information of the user stored within the social network, e.g., in a social network
4 database. Upon authentication, the social network server may generate an
5 authentication data record for the user, e.g., 2922, and provide an enrollment
6 notification, e.g., 2924, to the pay network server. For example, the social network
7 server may provide a HTTP(S) POST message similar to the example below:

```
8 POST /enrollnotification.php HTTP/1.1
9 Host: www.paynet.com
10 Content-Type: Application/XML
11 Content-Length: 1306
12 <?XML version = "1.0" encoding = "UTF-8"?>
13 <enroll_notification>
14     <request_ID>4NFU4RG94</order_ID>
15     <timestamp>2011-02-22 15:22:43</timestamp>
16     <result>enrolled</result>
17 </enroll_notification>
18
19
```

20 **[00361]** Upon receiving notification of enrollment from the social network server,
21 the pay network server may generate, e.g., 2925, a user enrollment data record, and
22 store the enrollment data record in a pay network database, e.g., 2926, to complete
23 enrollment. In some implementations, the enrollment data record may include the
24 information from the enrollment notification 2924.

25 **[00362]** FIGURE 30 shows a logic flow diagram illustrating example aspects of
26 enrollment in a value-added service in some embodiments of the MCB-Platform, e.g., a
27 Value-Add Service Enrollment (“VASE”) component 3000. In some implementations, a
28 user, e.g., 2901, may desire to enroll in a value-added service. Let us consider an
29 example wherein the user desires to enroll in social network authenticated purchase
30 payment as a value-added service. It is to be understood that any other value-added
31 service may take the place of the below-described value-added service. The user may

1 communicate with a pay network server via a client. For example, the user may provide
2 user input, e.g., 3001, into the client indicating the user's desire to enroll in social
3 network authenticated purchase payment. In various implementations, the user input
4 may include, but not be limited to: a single tap (e.g., a one-tap mobile app purchasing
5 embodiment) of a touchscreen interface, keyboard entry, card swipe, activating a
6 RFID/NFC enabled hardware device (e.g., electronic card having multiple accounts,
7 smartphone, tablet, etc.) within the user device, mouse clicks, depressing buttons on a
8 joystick/game console, voice commands, single/multi-touch gestures on a touch-
9 sensitive interface, touching user interface elements on a touch-sensitive display, and/or
10 the like. In some implementations, using the user's input, the client may generate an
11 enrollment request, e.g., 3002, and provide the enrollment request to the pay network
12 server. In some implementations, the SNPA may provide an enrollment button that
13 may take the user to an enrollment webpage where account info may be entered into
14 web form fields. In some implementations, the pay network server may obtain the
15 enrollment request from the client, and extract the user's payment detail from the
16 enrollment request. For example, the pay network server may utilize a parser such as
17 the example parsers described below in the discussion with reference to FIGURE 51. In
18 some implementations, the pay network server may query, e.g., 3004, a pay network
19 database to obtain a social network request template, e.g., 3005, to process the
20 enrollment request. The social network request template may include instructions,
21 data, login URL, login API call template and/or the like for facilitating social network
22 authentication. In some implementations, the pay network server may provide payment
23 information extracted from the card authorization request to the social network server
24 as part of a social network authentication enrollment request, e.g., 3006. In some

1 implementations, the social network server may provide a social network login request,
2 e.g., 3007, to the client. For example, the social network server may provide a HTML
3 input form to the client. The client may display, e.g., 3008, the login form for the user.
4 In some implementations, the user may provide login input into the client, e.g., 3009,
5 and the client may generate a social network login response for the social network
6 server. In some implementations, the social network server may authenticate the login
7 credentials of the user, and access payment account information of the user stored
8 within the social network, e.g., in a social network database. Upon authentication, the
9 social network server may generate an authentication data record for the user, e.g., 3011,
10 and provide an enrollment notification to the pay network server, e.g., 3013. Upon
11 receiving notification of enrollment from the social network server, the pay network
12 server may generate, e.g., 3014, a user enrollment data record, and store the enrollment
13 data record in a pay network database, e.g., 3015, to complete enrollment. The pay
14 network server may provide an enrollment confirmation, and provide the enrollment
15 confirmation to the client, which may display, e.g., 3017, the confirmation for the user.

16 **[00363]** FIGURES 31A-B show flow diagrams illustrating example aspects of
17 normalizing aggregated search, enrolled, service usage, transaction and/or other
18 aggregated data into a standardized data format in some embodiments of the MCB-
19 Platform, e.g., a Aggregated Data Record Normalization (“ADRN”) component 3100.
20 With reference to FIGURE 31A, in some implementations, a pay network server
21 (“server”) may attempt to convert any aggregated data records stored in an aggregated
22 records database it has access to in a normalized data format. For example, the
23 database may have a transaction data record template with predetermined, standard
24 fields that may store data in pre-defined formats (e.g., long integer / double float / 4

1 digits of precision, etc.) in a pre-determined data structure. A sample XML transaction
 2 data record template is provided below:

```

3  <?XML version = "1.0" encoding = "UTF-8"?>
4  <transaction_record>
5      <record_ID>00000000</record_ID>
6      <norm_flag>>false</norm_flag>
7      <timestamp>yyyy-mm-dd hh:mm:ss</timestamp>
8      <transaction_cost>$0,000,000,00</transaction_cost>
9      <merchant_params>
10         <merchant_id>00000000</merchant_id>
11         <merchant_name>TBD</merchant_name>
12         <merchant_auth_key>0000000000000000</merchant_auth_key>
13     </merchant_params>
14     <merchant_products>
15         <num_products>000</num_products>
16         <product>
17             <product_type>TBD</product_type>
18             <product_name>TBD</product_name>
19             <class_labels_list>TBD<class_labels_list>
20             <product_quantity>000</product_quantity>
21             <unit_value>$0,000,000.00</unit_value>
22             <sub_total>$0,000,000.00</sub_total>
23             <comment>normalized transaction data record
24 template</comment>
25         </product>
26     </merchant_products>
27     <user_account_params>
28         <account_name>JTBD</account_name>
29         <account_type>TBD</account_type>
30         <account_num>0000000000000000</account_num>
31         <billing_line1>TBD</billing_line1>
32         <billing_line2>TBD</billing_line2>
33         <zipcode>TBD</zipcode>
34         <state>TBD</state>
35         <country>TBD</country>
36         <phone>00-00-000-000-0000</phone>
37         <sign>TBD</sign>
38     </user_account_params>
39 </transaction_record>
40
41

```

42 **[00364]** In some implementations, the server may query a database for a
 43 normalized data record template, e.g., 3101. The server may parse the normalized data
 44 record template, e.g., 3102. Based on parsing the normalized data record template, the
 45 server may determine the data fields included in the normalized data record template,
 46 and the format of the data stored in the fields of the data record template, e.g., 3103.
 47 The server may obtain transaction data records for normalization. The server may
 48 query a database, e.g., 3104, for non-normalized records. For example, the server may

1 issue PHP/SQL commands to retrieve records that do not have the 'norm_flag' field
2 from the example template above, or those where the value of the 'norm_flag' field is
3 'false'. Upon obtaining the non-normalized transaction data records, the server may
4 select one of the non-normalized transaction data records, e.g., 3105. The server may
5 parse the non-normalized transaction data record, e.g., 3106, and determine the fields
6 present in the non-normalized transaction data record, e.g., 3107. For example, the
7 server may utilize a procedure similar to one described below with reference to FIGURE
8 32. The server may compare the fields from the non-normalized transaction data record
9 with the fields extracted from the normalized transaction data record template. For
10 example, the server may determine whether the field identifiers of fields in the non-
11 normalized transaction data record match those of the normalized transaction data
12 record template, (e.g., via a dictionary, thesaurus, etc.), are identical, are synonymous,
13 are related, and/or the like. Based on the comparison, the server may generate a 1:1
14 mapping between fields of the non-normalized transaction data record match those of
15 the normalized transaction data record template, e.g., 3109. The server may generate a
16 copy of the normalized transaction data record template, e.g., 3110, and populate the
17 fields of the template using values from the non-normalized transaction data record,
18 e.g., 3111. The server may also change the value of the 'norm_flag' field to 'true' in the
19 example above. The server may store the populated record in a database (for example,
20 replacing the original version), e.g., 3112. The server may repeat the above procedure
21 for each non-normalized transaction data record (see e.g., 3113), until all the non-
22 normalized transaction data records have been normalized.

23 **[00365]** With reference to FIGURE 31B, in some embodiments, the server may
24 utilize metadata (e.g., easily configurable data) to drive an analytics and rule engine that

1 may convert any structured data into a standardized XML format (“encryptmatics”
2 XML). The encryptmatics XML may then be processed by an encryptmatics engine that
3 is capable of parsing, transforming and analyzing data to generate decisions based on
4 the results of the analysis. Accordingly, in some embodiments, the server may
5 implement a metadata-based interpretation engine that parses structured data,
6 including, but not limited to: web content (see e.g., 3121), graph databases (see e.g.,
7 3122), micro blogs, images or software code (see e.g., 3124), and converts the structured
8 data into commands in the encryptmatics XML file format. For example, the structured
9 data may include, without limitation, software code, images, free text, relational
10 database queries, graph queries, sensory inputs (see e.g., 3123, 3125), and/or the like. A
11 metadata based interpretation engine engine, e.g., 3126, may populate a data/command
12 object, e.g., 3127, based on a given record using configurable metadata, e.g., 3128. The
13 configurable metadata may define an action for a given glyph or keyword contained
14 within a data record. The engine may then process the object to export its data structure
15 as a collection of encryptmatics vaults in a standard encryptmatics XML file format, e.g.,
16 3129. The encryptmatics XML file may then be processed to provide various features by
17 an encryptmatics engine, e.g., 3130.

18 **[00366]** In some embodiments, the server may obtain the structured data, and
19 perform a standardization routine using the structured data as input (e.g., including
20 script commands, for illustration). For example, the server may remove extra line
21 breaks, spaces, tab spaces, etc. from the structured data, e.g. 3131. The server may
22 determine and load a metadata library, e.g., 3132, using which the server may parse
23 subroutines or functions within the script, based on the metadata, e.g., 3133-3134. In
24 some embodiments, the server may pre-parse conditional statements based on the

1 metadata, e.g., 3135-3136. The server may also parse data 3137 to populate a
2 data/command object based on the metadata and prior parsing, e.g., 3138. Upon
3 finalizing the data/command object, the server may export 3139 the data/command
4 object as XML in standardized encryptmatics format.

5 **[00367]** FIGURE 32 shows a logic flow diagram illustrating example aspects of
6 recognizing data fields in normalized aggregated data records in some embodiments of
7 the MCB-Platform, e.g., a Data Field Recognition (“DFR”) component 3200. In some
8 implementations, a server may recognize the type of data fields included in a data
9 record, e.g., date, address, zipcode, name, user ID, email address, payment account
10 number (PAN), CVV2 numbers, and/or the like. The server may select an unprocessed
11 data record for processing, e.g., 3201. The server may parse the data record rule, and
12 extract data fields from the data record, e.g., 3202. The server may query a database for
13 data field templates, e.g., 3203. For example, the server may compare the format of the
14 fields from the data record to the data record templates to identify a match between one
15 of the data field templates and each field within the data record, thus identifying the
16 type of each field within the data record. The server may thus select an extracted data
17 field from the data record, e.g., 3204. The server may select a data field template for
18 comparison with the selected data field, e.g., 3205, and compare the data field template
19 with the selected data field, e.g., 3206, to determine whether format of extracted data
20 field matches format of data field template, e.g., 3207. If the format of the selected
21 extracted data field matches the format of the data field template, e.g., 3208, option
22 “Yes,” the server may assign the type of data field template to the selected data field, e.g.,
23 3209. If the format of the extracted data field does not match the format of the data
24 field template, e.g., 3208, option “No,” the server may try another data field template

1 until no more data field templates are available for comparison, see e.g., 3210. If no
2 match is found, the server may assign “unknown” string as the type of the data field,
3 e.g., 3211. The server may store the updated data record in the database, e.g., 3212. The
4 server may perform such data field recognition for each data field in the data record
5 (and also for each data record in the database), see e.g., 3213.

6 **[00368]** FIGURE 33 shows a logic flow diagram illustrating example aspects of
7 classifying entity types in some embodiments of the MCB-Platform, e.g., an Entity Type
8 Classification (“ETC”) component 3300. In some implementations, a server may apply
9 one or more classification labels to each of the data records. For example, the server
10 may classify the data records according to entity type, according to criteria such as, but
11 not limited to: geo-political area, number of items purchased, and/or the like. The
12 server may obtain transactions from a database that are unclassified, e.g., 3301, and
13 obtain rules and labels for classifying the records, e.g., 3302. For example, the database
14 may store classification rules, such as the exemplary illustrative XML-encoded
15 classification rule provided below:

```
16 <rule>
17     <id>PURCHASE_44_45</id>
18     <name>Number of purchasers</name>
19     <inputs>num_purchasers</inputs>
20     <operations>
21         <1>label = 'null'</1>
22         <2>IF (num_purchasers > 1) label = 'household'</2>
23     </operations>
24     <outputs>label</outputs>
25 </rule>
```

26
27
28 **[00369]** The server may select an unclassified data record for processing, e.g.,
29 3303. The server may also select a classification rule for processing the unclassified
30 data record, e.g., 3304. The server may parse the classification rule, and determine the
31 inputs required for the rule, e.g., 3305. Based on parsing the classification rule, the

1 server may parse the normalized data record template, e.g., 3306, and extract the values
2 for the fields required to be provided as inputs to the classification rule. The server may
3 parse the classification rule, and extract the operations to be performed on the inputs
4 provided for the rule processing, e.g., 3307. Upon determining the operations to be
5 performed, the server may perform the rule-specified operations on the inputs provided
6 for the classification rule, e.g., 3308. In some implementations, the rule may provide
7 threshold values. For example, the rule may specify that if the number of products in
8 the transaction, total value of the transaction, average luxury rating of the products sold
9 in the transaction, etc. may need to cross a threshold in order for the label(s) associated
10 with the rule to be applied to the transaction data record. The server may parse the
11 classification rule to extract any threshold values required for the rule to apply, e.g.,
12 3309. The server may compare the computed values with the rule thresholds, e.g., 3310.
13 If the rule threshold(s) is crossed, e.g., 3311, option “Yes,” the server may apply one or
14 more labels to the transaction data record as specified by the classification rule, e.g.,
15 3312. For example, the server may apply a classification rule to an individual product
16 within the transaction, and/or to the transaction as a whole. In some implementations,
17 the server may process the transaction data record using each rule (see, e.g., 3313).
18 Once all classification rules have been processed for the transaction record, e.g., 3313,
19 option “No,” the server may store the transaction data record in a database, e.g., 3314.
20 The server may perform such processing for each transaction data record until all
21 transaction data records have been classified (see, e.g., 3315).

22 **[00370]** FIGURE 34 shows a logic flow diagram illustrating example aspects of
23 identifying cross-entity correlation in some embodiments of the MCB-Platform, e.g., a
24 Cross-Entity Correlation (“CEC”) component 3400. In some implementations, a server

1 may recognize that two entities in the the MCB-Platform share common or related data
2 fields, e.g., date, address, zipcode, name, user ID, email address, payment account
3 number (PAN), CVV2 numbers, and/or the like, and thus identify the entities as being
4 correlated. The server may select a data record for cross-entity correlation, e.g., 3401.
5 The server may parse the data record rule, and extract data fields from the data record,
6 e.g., 3402-3403. The server may select an extracted data field from the data record, e.g.,
7 3404, and query a database for other data records having the same data field as the
8 extracted data field, e.g., 3405. From the list of retrieved data records from the database
9 query, the server may select a record for further analysis. The server may identify, e.g.,
10 3407, an entity associated with the retrieved data record, e.g., using the ETC 3300
11 component discussed above in the description with reference to FIGURE 33. The server
12 may add a data field to the data record obtained for cross-entity correlation specifying
13 the correlation to the retrieved selected data record, e.g., 3408. In some embodiments,
14 the server may utilize each data field in the data record obtained for cross-entity
15 correlation to identify correlated entities, see e.g., 3409. The server may add, once
16 complete, a “correlated” flag to the data record obtained for cross-entity correlation,
17 e.g., 3410, e.g., along with a timestamp specifying the time at which the cross-entity
18 correlation was performed. For example, such a timestamp may be used to determine at
19 a later time whether the data record should be processed again for cross-entity
20 correlation. The server may store the updated data record in a database.

21 **[00371]** FIGURE 35 shows a logic flow diagram illustrating example aspects of
22 associating attributes to entities in some embodiments of the MCB-Platform, e.g., an
23 Entity Attribute Association (“EAA”) component 3500. In some implementations, a
24 server may associate attributes to an entity, e.g., if the entity is a person, the server may

1 identify a demographic (e.g., male/female), a spend character, a purchase preferences
2 list, a merchants preference list, and/or the like, based on field values of data fields in
3 data records that are related to the entity. In some implementations, a server may
4 obtain a data record for entity attribute association, e.g., 3501. The server may parse the
5 data record rule, and extract data fields from the data record, e.g., 3502-3503. The
6 server may select an extracted data field from the data record, e.g., 3504, and identify a
7 field value for the selected extracted data field from the data record, e.g., 3505. The
8 server may query a database for demographic data, behavioral data, and/or the like, e.g.,
9 3506, using the field value and field type. In response, the database may provide a list
10 of potential attributes, as well as a confidence level in those attribute associations to the
11 entity, see e.g., 3507. The server may add data fields to the data record obtained for
12 entity attribute association specifying the potentially associated attributes and their
13 associated confidence levels, e.g., 3508. In some embodiments, the server may utilize
14 each data field in the data record obtained for cross-entity correlation to identify
15 correlated entities, see e.g., 3509. The server may store the updated data record in a
16 database, e.g., 3510.

17 **[00372]** FIGURE 36 shows a logic flow diagram illustrating example aspects of
18 updating entity profile-graphs in some embodiments of the MCB-Platform, e.g., an
19 Entity Profile-Graph Updating (“EPGU”) component 3600. In some implementations, a
20 server may generate/update a profile for an entity whose data is stored within the MCB-
21 Platform. The server may obtain an entity profile record for updating, e.g., 3601. The
22 server may parse the entity profile record, and extract an entity identifier data field from
23 the data record, e.g., 3602. The server may query a database for other data records that
24 are related to the same entity, e.g., 3603, using the value for the entity identifier data

1 field. In response, the database may provide a list of other data records for further
2 processing. The server may select one of the other data records to update the entity
3 profile record, e.g., 3604. The server may parse the data record, and extract all
4 correlations, associations, and new data from the other record, e.g., 3605. The server
5 may compare the correlations, attributes, associations, etc., from the other data record
6 with the correlations, associations and attributes from the entity profile. Based on this
7 comparison, the server may identify any new correlations, associations, etc., and
8 generate an updated entity profile record using the new correlations, associations; flag
9 new correlations, associations for further processing, e.g., 3607. In some embodiments,
10 the server may utilize each data record obtained for updating the entity profile record as
11 well as its social graph (e.g., as given by the correlations and associations for the entity),
12 see e.g., 3609. The server may store the updated entity profile record in a database, e.g.,
13 3608.

14 **[00373]** FIGURE 37 shows a logic flow diagram illustrating example aspects of
15 generating search terms for profile-graph updating in some embodiments of the MCB-
16 Platform, e.g., a Search Term Generation (“STG”) component 3700. In some
17 implementations, a server may generate/update a profile for an entity whose data is
18 stored within the MCB-Platform, by performing search for new data, e.g., across the
19 Internet and social networking services. The server may obtain an entity profile record
20 for updating, e.g., 3701. The server may parse the entity profile record, and extract data
21 field types and field values from the entity profile record, e.g., 3702. The server may
22 query a database for other data records that are related to the same entity, e.g., 3703,
23 using the values for the extracted data fields. In response, the database may provide a
24 list of other data records for further processing. The server may parse the data records,

1 and extract all correlations, associations, and data from the data records, e.g., 3704.
2 The server may aggregate all the data values from all the records and the entity profile
3 record, e.g., 3705. Based on this, the server may return the aggregated data values as
4 search terms to trigger search processes (see e.g., FIG.20, 2001-2005), e.g., 3706.

5 MCB-Platform Wallet UIs

6 **[00374]** FIGURE 38 shows a user interface diagram illustrating an overview of
7 example features of virtual wallet applications in some embodiments of the MCB-
8 Platform. FIGURE 38 shows an illustration of various exemplary features of a virtual
9 wallet mobile application 3800. Some of the features displayed include a wallet 3801,
10 social integration via TWITTER, FACEBOOK, etc., offers and loyalty 3803, snap mobile
11 purchase 3804, alerts 3805 and security, setting and analytics 3896. These features are
12 explored in further detail below.

13 **[00375]** FIGURES 39A-G show user interface diagrams illustrating example
14 features of virtual wallet applications in a shopping mode, in some embodiments of the
15 MCB-Platform. With reference to FIGURE 39A, some embodiments of the virtual wallet
16 mobile app facilitate and greatly enhance the shopping experience of consumers. A
17 variety of shopping modes, as shown in FIGURE 39A, may be available for a consumer
18 to peruse. In one implementation, for example, a user may launch the shopping mode by
19 selecting the shop icon 3910 at the bottom of the user interface. A user may type in an
20 item in the search field 3912 to search and/or add an item to a cart 3911. A user may also
21 use a voice activated shopping mode by saying the name or description of an item to be
22 searched and/or added to the cart into a microphone 3913. In a further implementation,

1 a user may also select other shopping options 3914 such as current items 3915, bills
2 3916, address book 3917, merchants 3918 and local proximity 3919.

3 **[00376]** In one embodiment, for example, a user may select the option current
4 items 3915, as shown in the left most user interface of FIGURE 39A. When the current
5 items 3915 option is selected, the middle user interface may be displayed. As shown, the
6 middle user interface may provide a current list of items 3915a-h in a user's shopping
7 cart 3911. A user may select an item, for example item 3915a, to view product description
8 3915j of the selected item and/or other items from the same merchant. The price and
9 total payable information may also be displayed, along with a QR code 3915k that
10 captures the information necessary to effect a snap mobile purchase transaction.

11 **[00377]** With reference to FIGURE 39B, in another embodiment, a user may select
12 the bills 3916 option. Upon selecting the bills 3916 option, the user interface may display
13 a list of bills and/or receipts 3916a-h from one or more merchants. Next to each of the
14 bills, additional information such as date of visit, whether items from multiple stores are
15 present, last bill payment date, auto-payment, number of items, and/or the like may be
16 displayed. In one example, the wallet shop bill 3916a dated January 20, 2011 may be
17 selected. The wallet shop bill selection may display a user interface that provides a
18 variety of information regarding the selected bill. For example, the user interface may
19 display a list of items 3916k purchased, <<3916i>>, a total number of items and the
20 corresponding value. For example, 7 items worth \$102.54 were in the selected wallet
21 shop bill. A user may now select any of the items and select buy again to add purchase
22 the items. The user may also refresh offers 3916j to clear any invalid offers from last
23 time and/or search for new offers that may be applicable for the current purchase. As

1 shown in FIGURE 39B, a user may select two items for repeat purchase. Upon addition,
2 a message 3916l may be displayed to confirm the addition of the two items, which makes
3 the total number of items in the cart 14.

4 **[00378]** With reference to FIGURE 39C, in yet another embodiment, a user may
5 select the address book option 3917 to view the address book 3917a which includes a list
6 of contacts 3917b and make any money transfers or payments. In one embodiment, the
7 address book may identify each contact using their names and available and/or
8 preferred modes of payment. For example, a contact Amanda G. may be paid via social
9 pay (e.g., via FACEBOOK) as indicated by the icon 3917c. In another example, money
10 may be transferred to Brian S. via QR code as indicated by the QR code icon 3917d. In
11 yet another example, Charles B. may accept payment via near field communication
12 3917e, Bluetooth 3917f and email 3917g. Payment may also be made via USB 3917h (e.g.,
13 by physically connecting two mobile devices) as well as other social channels such as
14 TWITTER.

15 **[00379]** In one implementation, a user may select Joe P. for payment. Joe P., as
16 shown in the user interface, has an email icon 3917g next to his name indicating that Joe
17 P. accepts payment via email. When his name is selected, the user interface may display
18 his contact information such as email, phone, etc. If a user wishes to make a payment to
19 Joe P. by a method other than email, the user may add another transfer mode 3917j to
20 his contact information and make a payment transfer. With reference to FIGURE 39D,
21 the user may be provided with a screen 3917k where the user can enter an amount to
22 send Joe, as well as add other text to provide Joe with context for the payment
23 transaction 3917l. The user can choose modes (e.g., SMS, email, social networking) via

1 which Joe may be contacted via graphical user interface elements, 3917m. As the user
2 types, the text entered may be provided for review within a GUI element 3917n. When
3 the user has completed entering in the necessary information, the user can press the
4 send button 3917o to send the social message to Joe. If Joe also has a virtual wallet
5 application, Joe may be able to review 3917p social pay message within the app, or
6 directly at the website of the social network (e.g., for Twitter™, Facebook®, etc.).
7 Messages may be aggregated from the various social networks and other sources (e.g.,
8 SMS, email). The method of redemption appropriate for each messaging mode may be
9 indicated along with the social pay message. In the illustration in FIGURE 39D, the
10 SMS 3917q Joe received indicates that Joe can redeem the \$5 obtained via SMS by
11 replying to the SMS and entering the hash tag value ‘#1234’. In the same illustration,
12 Joe has also received a message 3917r via Facebook®, which includes a URL link that
13 Joe can activate to initiate redemption of the \$25 payment.

14 **[00380]** With reference to FIGURE 39E, in some other embodiments, a user may
15 select merchants 3918 from the list of options in the shopping mode to view a select list
16 of merchants 3918a-e. In one implementation, the merchants in the list may be affiliated
17 to the wallet, or have affinity relationship with the wallet. In another implementation,
18 the merchants may include a list of merchants meeting a user-defined or other criteria.
19 For example, the list may be one that is curated by the user, merchants where the user
20 most frequently shops or spends more than an x amount of sum or shopped for three
21 consecutive months, and/or the like. In one implementation, the user may further select
22 one of the merchants, Amazon 3918a for example. The user may then navigate through
23 the merchant’s listings to find items of interest such as 3918f-j. Directly through the
24 wallet and without visiting the merchant site from a separate page, the user may make a

1 selection of an item 3918j from the catalog of Amazon 3918a. As shown in the right most
2 user interface of FIGURE 39D, the selected item may then be added to cart. The
3 message 3918k indicates that the selected item has been added to the cart, and updated
4 number of items in the cart is now 13.

5 **[00381]** With reference to FIGURE 39F, in one embodiment, there may be a local
6 proximity option 3919 which may be selected by a user to view a list of merchants that
7 are geographically in close proximity to the user. For example, the list of merchants
8 3919a-e may be the merchants that are located close to the user. In one implementation,
9 the mobile application may further identify when the user is in a store based on the user's
10 location. For example, position icon 3919d may be displayed next to a store (e.g.,
11 Walgreens) when the user is in close proximity to the store. In one implementation, the
12 mobile application may refresh its location periodically in case the user moved away
13 from the store (e.g., Walgreens). In a further implementation, the user may navigate the
14 offerings of the selected Walgreens store through the mobile application. For example,
15 the user may navigate, using the mobile application, to items 3919f-j available on aisle 5
16 of Walgreens. In one implementation, the user may select corn 3919i from his or her
17 mobile application to add to cart 3919k.

18 **[00382]** With reference to FIGURE 39G, in another embodiment, the local
19 proximity option 3919 may include a store map and a real time map features among
20 others. For example, upon selecting the Walgreens store, the user may launch an aisle
21 map 3919l which displays a map 3919m showing the organization of the store and the
22 position of the user (indicated by a yellow circle). In one implementation, the user may
23 easily configure the map to add one or more other users (e.g., user's kids) to share each

1 other's location within the store. In another implementation, the user may have the
2 option to launch a "store view" similar to street views in maps. The store view 3919n
3 may display images/video of the user's surrounding. For example, if the user is about to
4 enter aisle 5, the store view map may show the view of aisle 5. Further the user may
5 manipulate the orientation of the map using the navigation tool 3919o to move the store
6 view forwards, backwards, right, left as well clockwise and counterclockwise rotation

7 **[00383]** FIGURES 40A-F show user interface diagrams illustrating example
8 features of virtual wallet applications in a payment mode, in some embodiments of the
9 MCB-Platform. With reference to FIGURE 40A, in one embodiment, the wallet mobile
10 application may provide a user with a number of options for paying for a transaction via
11 the wallet mode 4010. In one implementation, an example user interface 4011 for
12 making a payment is shown. The user interface may clearly identify the amount 4012
13 and the currency 4013 for the transaction. The amount may be the amount payable and
14 the currency may include real currencies such as dollars and euros, as well as virtual
15 currencies such as reward points. The amount of the transaction 4014 may also be
16 prominently displayed on the user interface. The user may select the funds tab 4016 to
17 select one or more forms of payment 4017, which may include various credit, debit, gift,
18 rewards and/or prepaid cards. The user may also have the option of paying, wholly or in
19 part, with reward points. For example, the graphical indicator 4018 on the user interface
20 shows the number of points available, the graphical indicator 4019 shows the number of
21 points to be used towards the amount due 234.56 and the equivalent 4020 of the
22 number of points in a selected currency (USD, for example).

1 **[00384]** In one implementation, the user may combine funds from multiple
2 sources to pay for the transaction. The amount 4015 displayed on the user interface may
3 provide an indication of the amount of total funds covered so far by the selected forms of
4 payment (e.g., Discover card and rewards points). The user may choose another form of
5 payment or adjust the amount to be debited from one or more forms of payment until
6 the amount 4015 matches the amount payable 4014. Once the amounts to be debited
7 from one or more forms of payment are finalized by the user, payment authorization
8 may begin.

9 **[00385]** In one implementation, the user may select a secure authorization of the
10 transaction by selecting the cloak button 4022 to effectively cloak or anonymize some
11 (e.g., pre-configured) or all identifying information such that when the user selects pay
12 button 4021, the transaction authorization is conducted in a secure and anonymous
13 manner. In another implementation, the user may select the pay button 4021 which may
14 use standard authorization techniques for transaction processing. In yet another
15 implementation, when the user selects the social button 4023, a message regarding the
16 transaction may be communicated to one of more social networks (set up by the user)
17 which may post or announce the purchase transaction in a social forum such as a wall
18 post or a tweet. In one implementation, the user may select a social payment processing
19 option 4023. The indicator 4024 may show the authorizing and sending social share
20 data in progress.

21 **[00386]** In another implementation, a restricted payment mode 4025 may be
22 activated for certain purchase activities such as prescription purchases. The mode may
23 be activated in accordance with rules defined by issuers, insurers, merchants, payment

1 processor and/or other entities to facilitate processing of specialized goods and services.
2 In this mode, the user may scroll down the list of forms of payments 4026 under the
3 funds tab to select specialized accounts such as a flexible spending account (FSA) 4027,
4 health savings account (HAS), and/or the like and amounts to be debited to the selected
5 accounts. In one implementation, such restricted payment mode 1925 processing may
6 disable social sharing of purchase information.

7 **[00387]** In one embodiment, the wallet mobile application may facilitate importing
8 of funds via the import funds user interface 4028. For example, a user who is
9 unemployed may obtain unemployment benefit fund 4029 via the wallet mobile
10 application. In one implementation, the entity providing the funds may also configure
11 rules for using the fund as shown by the processing indicator message 4030. The wallet
12 may read and apply the rules prior, and may reject any purchases with the
13 unemployment funds that fail to meet the criteria set by the rules. Example criteria may
14 include, for example, merchant category code (MCC), time of transaction, location of
15 transaction, and/or the like. As an example, a transaction with a grocery merchant
16 having MCC 5411 may be approved, while a transaction with a bar merchant having an
17 MCC 5813 may be refused.

18 **[00388]** With reference to FIGURE 40B, in one embodiment, the wallet mobile
19 application may facilitate dynamic payment optimization based on factors such as user
20 location, preferences and currency value preferences among others. For example, when
21 a user is in the United States, the country indicator 4031 may display a flag of the United
22 States and may set the currency 4033 to the United States. In a further implementation,
23 the wallet mobile application may automatically rearrange the order in which the forms

1 of payments 4035 are listed to reflect the popularity or acceptability of various forms of
2 payment. In one implementation, the arrangement may reflect the user's preference,
3 which may not be changed by the wallet mobile application.

4 **[00389]** Similarly, when a German user operates a wallet in Germany, the mobile
5 wallet application user interface may be dynamically updated to reflect the country of
6 operation 4032 and the currency 4034. In a further implementation, the wallet
7 application may rearrange the order in which different forms of payment 4036 are listed
8 based on their acceptance level in that country. Of course, the order of these forms of
9 payments may be modified by the user to suit his or her own preferences.

10 **[00390]** With reference to FIGURE 40C, in one embodiment, the payee tab 4037 in
11 the wallet mobile application user interface may facilitate user selection of one or more
12 payees receiving the funds selected in the funds tab. In one implementation, the user
13 interface may show a list of all payees 4038 with whom the user has previously
14 transacted or available to transact. The user may then select one or more payees. The
15 payees 4038 may include larger merchants such as Amazon.com Inc., and individuals
16 such as Jane P. Doe. Next to each payee name, a list of accepted payment modes for the
17 payee may be displayed. In one implementation, the user may select the payee Jane P.
18 Doe 4039 for receiving payment. Upon selection, the user interface may display
19 additional identifying information relating to the payee.

20 **[00391]** With reference to FIGURE 40D, in one embodiment, the mode tab 1940
21 may facilitate selection of a payment mode accepted by the payee. A number of payment
22 modes may be available for selection. Example modes include, blue tooth 4041, wireless
23 4042, snap mobile by user-obtained QR code 4043, secure chip 4044, TWITTER 4045,

1 near-field communication (NFC) 4046, cellular 4047, snap mobile by user-provided QR
2 code 4048, USB 4049 and FACEBOOK 4050, among others. In one implementation,
3 only the payment modes that are accepted by the payee may be selectable by the user.
4 Other non-accepted payment modes may be disabled.

5 **[00392]** With reference to FIGURE 40E, in one embodiment, the offers tab 4051
6 may provide real-time offers that are relevant to items in a user's cart for selection by
7 the user. The user may select one or more offers from the list of applicable offers 4052
8 for redemption. In one implementation, some offers may be combined, while others
9 may not. When the user selects an offer that may not be combined with another offer,
10 the unselected offers may be disabled. In a further implementation, offers that are
11 recommended by the wallet application's recommendation engine may be identified by
12 an indicator, such as the one shown by 4053. In a further implementation, the user may
13 read the details of the offer by expanding the offer row as shown by 4054 in the user
14 interface.

15 **[00393]** With reference to FIGURE 40F, in one embodiment, the social tab 4055
16 may facilitate integration of the wallet application with social channels 4056. In one
17 implementation, a user may select one or more social channels 4056 and may sign in to
18 the selected social channel from the wallet application by providing to the wallet
19 application the social channel user name and password 4057 and signing in 4058. The
20 user may then use the social button 4059 to send or receive money through the
21 integrated social channels. In a further implementation, the user may send social share
22 data such as purchase information or links through integrated social channels. In

1 another embodiment, the user supplied login credentials may allow MCB-Platform to
2 engage in interception parsing.

3 **[00394]** FIGURE 41 shows a user interface diagram illustrating example features of
4 virtual wallet applications, in a history mode, in some embodiments of the MCB-
5 Platform. In one embodiment, a user may select the history mode 4110 to view a history
6 of prior purchases and perform various actions on those prior purchases. For example, a
7 user may enter a merchant identifying information such as name, product, MCC, and/or
8 the like in the search bar 4111. In another implementation, the user may use voice
9 activated search feature by clicking on the microphone icon 4114. The wallet application
10 may query the storage areas in the mobile device or elsewhere (e.g., one or more
11 databases and/or tables remote from the mobile device) for transactions matching the
12 search keywords. The user interface may then display the results of the query such as
13 transaction 4115. The user interface may also identify the date 4112 of the transaction,
14 the merchants and items 4113 relating to the transaction, a barcode of the receipt
15 confirming that a transaction was made, the amount of the transaction and any other
16 relevant information.

17 **[00395]** In one implementation, the user may select a transaction, for example
18 transaction 4115, to view the details of the transaction. For example, the user may view
19 the details of the items associated with the transaction and the amounts 4116 of each
20 item. In a further implementation, the user may select the show option 4117 to view
21 actions 4118 that the user may take in regards to the transaction or the items in the
22 transaction. For example, the user may add a photo to the transaction (e.g., a picture of
23 the user and the iPad the user bought). In a further implementation, if the user

1 previously shared the purchase via social channels, a post including the photo may be
2 generated and sent to the social channels for publishing. In one implementation, any
3 sharing may be optional, and the user, who did not share the purchase via social
4 channels, may still share the photo through one or more social channels of his or her
5 choice directly from the history mode of the wallet application. In another
6 implementation, the user may add the transaction to a group such as company expense,
7 home expense, travel expense or other categories set up by the user. Such grouping may
8 facilitate year-end accounting of expenses, submission of work expense reports,
9 submission for value added tax (VAT) refunds, personal expenses, and/or the like. In yet
10 another implementation, the user may buy one or more items purchased in the
11 transaction. The user may then execute a transaction without going to the merchant
12 catalog or site to find the items. In a further implementation, the user may also cart one
13 or more items in the transaction for later purchase.

14 **[00396]** The history mode, in another embodiment, may offer facilities for
15 obtaining and displaying ratings 4119 of the items in the transaction. The source of the
16 ratings may be the user, the user's friends (e.g., from social channels, contacts, etc.),
17 reviews aggregated from the web, and/or the like. The user interface in some
18 implementations may also allow the user to post messages to other users of social
19 channels (e.g., TWITTER or FACEBOOK). For example, the display area 4120 shows
20 FACEBOOK message exchanges between two users. In one implementation, a user may
21 share a link via a message 4121. Selection of such a message having embedded link to a
22 product may allow the user to view a description of the product and/or purchase the
23 product directly from the history mode.

1 **[00397]** In one embodiment, the history mode may also include facilities for
2 exporting receipts. The export receipts pop up 4122 may provide a number of options for
3 exporting the receipts of transactions in the history. For example, a user may use one or
4 more of the options 4125, which include save (to local mobile memory, to server, to a
5 cloud account, and/or the like), print to a printer, fax, email, and/or the like. The user
6 may utilize his or her address book 4123 to look up email or fax number for exporting.
7 The user may also specify format options 4124 for exporting receipts. Example format
8 options may include, without limitation, text files (.doc, .txt, .rtf, iif, etc.), spreadsheet
9 (.csv, .xls, etc.), image files (.jpg, .tiff, .png, etc.), portable document format (.pdf),
10 postscript (.ps), and/or the like. The user may then click or tap the export button 4127 to
11 initiate export of receipts.

12 **[00398]** FIGURES 42A-E show user interface diagrams illustrating example
13 features of virtual wallet applications in a snap mode, in some embodiments of the
14 MCB-Platform. With reference to FIGURE 42A, in one embodiment, a user may select
15 the snap mode 2110 to access its snap features. The snap mode may handle any
16 machine-readable representation of data. Examples of such data may include linear and
17 2D bar codes such as UPC code and QR codes. These codes may be found on receipts,
18 product packaging, and/or the like. The snap mode may also process and handle
19 pictures of receipts, products, offers, credit cards or other payment devices, and/or the
20 like. An example user interface in snap mode is shown in FIGURE 42A. A user may use
21 his or her mobile phone to take a picture of a QR code 4215 and/or a barcode 4214. In
22 one implementation, the bar 4213 and snap frame 4215 may assist the user in snapping
23 codes properly. For example, the snap frame 4215, as shown, does not capture the
24 entirety of the code 4216. As such, the code captured in this view may not be resolvable

1 as information in the code may be incomplete. This is indicated by the message on the
2 bar 4213 that indicates that the snap mode is still seeking the code. When the code 4216
3 is completely framed by the snap frame 4215, the bar message may be updated to, for
4 example, “snap found.” Upon finding the code, in one implementation, the user may
5 initiate code capture using the mobile device camera. In another implementation, the
6 snap mode may automatically snap the code using the mobile device camera.

7 **[00399]** With reference to FIGURE 42B, in one embodiment, the snap mode may
8 facilitate payment reallocation post transaction. For example, a user may buy grocery
9 and prescription items from a retailer Acme Supermarket. The user may, inadvertently
10 or for ease of checkout for example, use his or her Visa card to pay for both grocery and
11 prescription items. However, the user may have an FSA account that could be used to
12 pay for prescription items, and which would provide the user tax benefits. In such a
13 situation, the user may use the snap mode to initiate transaction reallocation.

14 **[00400]** As shown, the user may enter a search term (e.g., bills) in the search bar
15 2121. The user may then identify in the tab 4222 the receipt 4223 the user wants to
16 reallocate. Alternatively, the user may directly snap a picture of a barcode on a receipt,
17 and the snap mode may generate and display a receipt 4223 using information from the
18 barcode. The user may now reallocate 4225. In some implementations, the user may
19 also dispute the transaction 4224 or archive the receipt 4226.

20 **[00401]** In one implementation, when the reallocate button 4225 is selected, the
21 wallet application may perform optical character recognition (OCR) of the receipt. Each
22 of the items in the receipt may then be examined to identify one or more items which
23 could be charged to which payment device or account for tax or other benefits such as

1 cash back, reward points, etc. In this example, there is a tax benefit if the prescription
2 medication charged to the user's Visa card is charged to the user's FSA. The wallet
3 application may then perform the reallocation as the back end. The reallocation process
4 may include the wallet contacting the payment processor to credit the amount of the
5 prescription medication to the Visa card and debit the same amount to the user's FSA
6 account. In an alternate implementation, the payment processor (e.g., Visa or
7 MasterCard) may obtain and OCR the receipt, identify items and payment accounts for
8 reallocation and perform the reallocation. In one implementation, the wallet application
9 may request the user to confirm reallocation of charges for the selected items to another
10 payment account. The receipt 4227 may be generated after the completion of the
11 reallocation process. As discussed, the receipt shows that some charges have been
12 moved from the Visa account to the FSA.

13 **[00402]** With reference to FIGURE 42C, in one embodiment, the snap mode may
14 facilitate payment via pay code such as barcodes or QR codes. For example, a user may
15 snap a QR code of a transaction that is not yet complete. The QR code may be displayed
16 at a merchant POS terminal, a web site, or a web application and may be encoded with
17 information identifying items for purchase, merchant details and other relevant
18 information. When the user snaps such as a QR code, the snap mode may decode the
19 information in the QR code and may use the decoded information to generate a receipt
20 4232. Once the QR code is identified, the navigation bar 4231 may indicate that the pay
21 code is identified. The user may now have an option to add to cart 4233, pay with a
22 default payment account 4234 or pay with wallet 4235.

1 **[00403]** In one implementation, the user may decide to pay with default 4234. The
2 wallet application may then use the user's default method of payment, in this example
3 the wallet, to complete the purchase transaction. Upon completion of the transaction, a
4 receipt may be automatically generated for proof of purchase. The user interface may
5 also be updated to provide other options for handling a completed transaction. Example
6 options include social 4237 to share purchase information with others, reallocate 4238
7 as discussed with regard to FIGURE 42B, and archive 4239 to store the receipt.

8 **[00404]** With reference to FIGURE 42D, in one embodiment, the snap mode may
9 also facilitate offer identification, application and storage for future use. For example, in
10 one implementation, a user may snap an offer code 4241 (e.g., a bar code, a QR code,
11 and/or the like). The wallet application may then generate an offer text 4242 from the
12 information encoded in the offer code. The user may perform a number of actions on the
13 offer code. For example, the user use the find button 4243 to find all merchants who
14 accept the offer code, merchants in the proximity who accept the offer code, products
15 from merchants that qualify for the offer code, and/or the like. The user may also apply
16 the offer code to items that are currently in the cart using the add to cart button 4244.
17 Furthermore, the user may also save the offer for future use by selecting the save button
18 4245.

19 **[00405]** In one implementation, after the offer or coupon 4246 is applied, the user
20 may have the option to find qualifying merchants and/or products using find, the user
21 may go to the wallet using 4248, and the user may also save the offer or coupon 4246 for
22 later use.

1 **[00406]** With reference to FIGURE 42E, in one embodiment, the snap mode may
2 also offer facilities for adding a funding source to the wallet application. In one
3 implementation, a pay card such as a credit card, debit card, pre-paid card, smart card
4 and other pay accounts may have an associated code such as a bar code or QR code.
5 Such a code may have encoded therein pay card information including, but not limited
6 to, name, address, pay card type, pay card account details, balance amount, spending
7 limit, rewards balance, and/or the like. In one implementation, the code may be found
8 on a face of the physical pay card. In another implementation, the code may be obtained
9 by accessing an associated online account or another secure location. In yet another
10 implementation, the code may be printed on a letter accompanying the pay card. A user,
11 in one implementation, may snap a picture of the code. The wallet application may
12 identify the pay card 4251 and may display the textual information 4252 encoded in the
13 pay card. The user may then perform verification of the information 4252 by selecting
14 the verify button 4253. In one implementation, the verification may include contacting
15 the issuer of the pay card for confirmation of the decoded information 4252 and any
16 other relevant information. In one implementation, the user may add the pay card to the
17 wallet by selecting the 'add to wallet' button 4254. The instruction to add the pay card to
18 the wallet may cause the pay card to appear as one of the forms of payment under the
19 funds tab 4016 discussed in FIGURE 40A. The user may also cancel importing of the pay
20 card as a funding source by selecting the cancel button 4255. When the pay card has
21 been added to the wallet, the user interface may be updated to indicate that the
22 importing is complete via the notification display 4256. The user may then access the
23 wallet 4257 to begin using the added pay card as a funding source.

1 **[00407]** FIGURE 43 shows a user interface diagram illustrating example features
2 of virtual wallet applications, in an offers mode, in some embodiments of the MCB-
3 Platform. In some implementations, the MCB-Platform may allow a user to search for
4 offers for products and/or services from within the virtual wallet mobile application.
5 For example, the user may enter text into a graphical user interface (“GUI”) element
6 4311, or issue voice commands by activating GUI element 4312 and speaking commands
7 into the device. In some implementations, the MCB-Platform may provide offers based
8 on the user’s prior behavior, demographics, current location, current cart selection or
9 purchase items, and/or the like. For example, if a user is in a brick-and-mortar store, or
10 an online shopping website, and leaves the (virtual) store, then the merchant associated
11 with the store may desire to provide a sweetener deal to entice the consumer back into
12 the (virtual) store. The merchant may provide such an offer 4313. For example, the
13 offer may provide a discount, and may include an expiry time. In some
14 implementations, other users may provide gifts (e.g., 4314) to the user, which the user
15 may redeem. In some implementations, the offers section may include alerts as to
16 payment of funds outstanding to other users (e.g., 4315). In some implementations, the
17 offers section may include alerts as to requesting receipt of funds from other users (e.g.,
18 4316). For example, such a feature may identify funds receivable from other
19 applications (e.g., mail, calendar, tasks, notes, reminder programs, alarm, etc.), or by a
20 manual entry by the user into the virtual wallet application. In some implementations,
21 the offers section may provide offers from participating merchants in the MCB-
22 Platform, e.g., 4317-4319, 4320. These offers may sometimes be assembled using a
23 combination of participating merchants, e.g., 4317. In some implementations, the MCB-

1 Platform itself may provide offers for users contingent on the user utilizing particular
2 payment forms from within the virtual wallet application, e.g., 4320.

3 **[00408]** FIGURES 44A-B show user interface diagrams illustrating example
4 features of virtual wallet applications, in a security and privacy mode, in some
5 embodiments of the MCB-Platform. With reference to FIGURE 44A, in some
6 implementations, the user may be able to view and/or modify the user profile and/or
7 settings of the user, e.g., by activating a user interface element. For example, the user
8 may be able to view/modify a user name (e.g., 4411a-b), account number (e.g., 4412a-b),
9 user security access code (e.g., 4413-b), user pin (e.g., 4414-b), user address (e.g., 4415-
10 b), social security number associated with the user (e.g., 4416-b), current device GPS
11 location (e.g., 4417-b), user account of the merchant in whose store the user currently is
12 (e.g., 4418-b), the user's rewards accounts (e.g., 4419-b), and/or the like. In some
13 implementations, the user may be able to select which of the data fields and their
14 associated values should be transmitted to facilitate the purchase transaction, thus
15 providing enhanced data security for the user. For example, in the example illustration
16 in FIGURE 44A, the user has selected the name 4411a, account number 4412a, security
17 code 4413a, merchant account ID 4418a and rewards account ID 4419a as the fields to
18 be sent as part of the notification to process the purchase transaction. In some
19 implementations, the user may toggle the fields and/or data values that are sent as part
20 of the notification to process the purchase transactions. In some implementations, the
21 app may provide multiple screens of data fields and/or associated values stored for the
22 user to select as part of the purchase order transmission. In some implementations, the
23 app may provide the MCB-Platform with the GPS location of the user. Based on the
24 GPS location of the user, the MCB-Platform may determine the context of the user (e.g.,

1 whether the user is in a store, doctor's office, hospital, postal service office, etc.). Based
2 on the context, the user app may present the appropriate fields to the user, from which
3 the user may select fields and/or field values to send as part of the purchase order
4 transmission.

5 **[00409]** For example, a user may go to doctor's office and desire to pay the co-pay
6 for doctor's appointment. In addition to basic transactional information such as
7 account number and name, the app may provide the user the ability to select to transfer
8 medical records, health information, which may be provided to the medical provider,
9 insurance company, as well as the transaction processor to reconcile payments between
10 the parties. In some implementations, the records may be sent in a Health Insurance
11 Portability and Accountability Act (HIPAA)-compliant data format and encrypted, and
12 only the recipients who are authorized to view such records may have appropriate
13 decryption keys to decrypt and view the private user information.

14 **[00410]** With reference to FIGURE 44B, in some implementations, the app
15 executing on the user's device may provide a "VerifyChat" feature for fraud prevention.
16 For example, the MCB-Platform may detect an unusual and/or suspicious transaction.
17 The MCB-Platform may utilize the VerifyChat feature to communicate with the user,
18 and verify the authenticity of the originator of the purchase transaction. In various
19 implementations, the MCB-Platform may send electronic mail message, text (SMS)
20 messages, Facebook® messages, Twitter™ tweets, text chat, voice chat, video chat (e.g.,
21 Apple FaceTime), and/or the like to communicate with the user. For example, the MCB-
22 Platform may initiate a video challenge for the user, e.g., 4421. For example, the user
23 may need to present him/her-self via a video chat, e.g., 4422. In some implementations,

1 a customer service representative, e.g., agent 4424, may manually determine the
2 authenticity of the user using the video of the user. In some implementations, the MCB-
3 Platform may utilize face, biometric and/or like recognition (e.g., using pattern
4 classification techniques) to determine the identity of the user. In some
5 implementations, the app may provide reference marker (e.g., cross-hairs, target box,
6 etc.), e.g., 4423, so that the user may the video to facilitate the MCB-Platform's
7 automated recognition of the user. In some implementations, the user may not have
8 initiated the transaction, e.g., the transaction is fraudulent. In such implementations,
9 the user may cancel the challenge. The MCB-Platform may then cancel the transaction,
10 and/or initiate fraud investigation procedures on behalf of the user.

11 **[00411]** In some implementations, the MCB-Platform may utilize a text challenge
12 procedure to verify the authenticity of the user, e.g., 4425. For example, the MCB-
13 Platform may communicate with the user via text chat, SMS messages, electronic mail,
14 Facebook® messages, Twitter™ tweets, and/or the like. The MCB-Platform may pose a
15 challenge question, e.g., 4426, for the user. The app may provide a user input interface
16 element(s) (e.g., virtual keyboard 4428) to answer the challenge question posed by the
17 MCB-Platform. In some implementations, the challenge question may be randomly
18 selected by the MCB-Platform automatically; in some implementations, a customer
19 service representative may manually communicate with the user. In some
20 implementations, the user may not have initiated the transaction, e.g., the transaction is
21 fraudulent. In such implementations, the user may cancel the text challenge. The MCB-
22 Platform may cancel the transaction, and/or initiate fraud investigation on behalf of the
23 user.

MCB-Platform Card Based Transactions

1

2 **[00412]** FIGURE 45 shows a data flow diagram illustrating an example user
3 purchase checkout procedure in some embodiments of the MCB-Platform. In some
4 embodiments, a user, e.g., 4501a, may desire to purchase a product, service, offering,
5 and/or the like (“product”), from a merchant via a merchant online site or in the
6 merchant’s store. The user may communicate with a merchant/acquirer (“merchant”)
7 server, e.g., 4503a, via a client such as, but not limited to: a personal computer, mobile
8 device, television, point-of-sale terminal, kiosk, ATM, and/or the like (e.g., 4502). For
9 example, the user may provide user input, e.g., checkout input 4511, into the client
10 indicating the user’s desire to purchase the product. In various embodiments, the user
11 input may include, but not be limited to: a single tap (e.g., a one-tap mobile app
12 purchasing embodiment) of a touchscreen interface, keyboard entry, card swipe,
13 activating a RFID/NFC enabled hardware device (e.g., electronic card having multiple
14 accounts, smartphone, tablet, etc.) within the user device, mouse clicks, depressing
15 buttons on a joystick/game console, voice commands, single/multi-touch gestures on a
16 touch-sensitive interface, touching user interface elements on a touch-sensitive display,
17 and/or the like. As an example, a user in a merchant store may scan a product barcode
18 of the product via a barcode scanner at a point-of-sale terminal. As another example,
19 the user may select a product from a webpage catalog on the merchant’s website, and
20 add the product to a virtual shopping cart on the merchant’s website. The user may
21 then indicate the user’s desire to checkout the items in the (virtual) shopping cart. For
22 example, the user may activate a user interface element provided by the client to
23 indicate the user’s desire to complete the user purchase checkout. The client may
24 generate a checkout request, e.g., 4512, and provide the checkout request, e.g., 4513, to

1 the merchant server. For example, the client may provide a (Secure) Hypertext Transfer
 2 Protocol ("HTTP(S)") POST message including the product details for the merchant
 3 server in the form of data formatted according to the eXtensible Markup Language
 4 ("XML"). An example listing of a checkout request 4512, substantially in the form of a
 5 HTTP(S) POST message including XML-formatted data, is provided below:

```

6  POST /checkoutrequest.php HTTP/1.1
7  Host: www.merchant.com
8  Content-Type: Application/XML
9  Content-Length: 667
10 <?XML version = "1.0" encoding = "UTF-8"?>
11 <checkout_request>
12     <checkout_ID>4NFU4RG94</checkout_ID>
13     <timestamp>2011-02-22 15:22:43</timestamp>
14     <purchase_detail>
15         <num_products>5</num_products>
16         <product_ID>AE95049324</product_ID>
17         <product_ID>MD09808755</product_ID>
18         <product_ID>OC12345764</product_ID>
19         <product_ID>KE76549043</product_ID>
20         <product_ID>SP27674509</product_ID>
21     </purchase_detail>
22 <!--optional parameters-->
23     <user_ID>john.q.public@gmail.com</user_ID>
24     <PoS_client_detail>
25         <client_IP>192.168.23.126</client_IP>
26         <client_type>smartphone</client_type>
27         <client_model>HTC Hero</client_model>
28         <OS>Android 2.2</OS>
29         <app_installed_flag>true</app_installed_flag>
30     </PoS_client_detail>
31 </checkout_request>
32

```

33 **[00413]** In some embodiments, the merchant server may obtain the checkout
 34 request from the client, and extract the checkout detail (e.g., XML data) from the
 35 checkout request. For example, the merchant server may utilize a parser such as the
 36 example parsers described below in the discussion with reference to FIGURE 51. Based
 37 on parsing the checkout request 4512, the merchant server may extract product data
 38 (e.g., product identifiers), as well as available PoS client data, from the checkout request.
 39 In some embodiments, using the product data, the merchant server may query, e.g.,
 40 4514, a merchant/acquirer ("merchant") database, e.g., 4503b, to obtain product data,

1 e.g., 4515, such as product information, product pricing, sales tax, offers, discounts,
 2 rewards, and/or other information to process the purchase transaction and/or provide
 3 value-added services for the user. For example, the merchant database may be a
 4 relational database responsive to Structured Query Language (“SQL”) commands. The
 5 merchant server may execute a hypertext preprocessor (“PHP”) script including SQL
 6 commands to query a database table (such as FIGURE 51, Products 5119r) for product
 7 data. An example product data query 4514, substantially in the form of PHP/SQL
 8 commands, is provided below:

```

9  <?PHP
10 header('Content-Type: text/plain');
11 mysql_connect("254.93.179.112",$DBserver,$password); // access database server
12 mysql_select_db("MCB-Platform_DB.SQL"); // select database table to search
13 //create query
14 $query = "SELECT product_title product_attributes_list product_price
15 tax_info_list related_products_list offers_list discounts_list rewards_list
16 merchants_list merchant_availability_list FROM ProductsTable WHERE product_ID
17 LIKE '%" $prodID";
18 $result = mysql_query($query); // perform the search query
19 mysql_close("MCB-Platform_DB.SQL"); // close database access
20 ?>
21

```

22 **[00414]** In some embodiments, in response to obtaining the product data, the
 23 merchant server may generate, e.g., 4516, checkout data to provide for the PoS client. In
 24 some embodiments, such checkout data, e.g., 4517, may be embodied, in part, in a
 25 HyperText Markup Language (“HTML”) page including data for display, such as
 26 product detail, product pricing, total pricing, tax information, shipping information,
 27 offers, discounts, rewards, value-added service information, etc., and input fields to
 28 provide payment information to process the purchase transaction, such as account
 29 holder name, account number, billing address, shipping address, tip amount, etc. In
 30 some embodiments, the checkout data may be embodied, in part, in a Quick Response
 31 (“QR”) code image that the PoS client can display, so that the user may capture the QR
 32 code using a user’s device to obtain merchant and/or product data for generating a

1 purchase transaction processing request. In some embodiments, a user alert
2 mechanism may be built into the checkout data. For example, the merchant server may
3 embed a URL specific to the transaction into the checkout data. In some embodiments,
4 the alerts URL may further be embedded into optional level 3 data in card authorization
5 requests, such as those discussed further below with reference to FIGURES 47-48. The
6 URL may point to a webpage, data file, executable script, etc., stored on the merchant's
7 server dedicated to the transaction that is the subject of the card authorization request.
8 For example, the object pointed to by the URL may include details on the purchase
9 transaction, e.g., products being purchased, purchase cost, time expiry, status of order
10 processing, and/or the like. Thus, the merchant server may provide to the payment
11 network the details of the transaction by passing the URL of the webpage to the
12 payment network. In some embodiments, the payment network may provide
13 notifications to the user, such as a payment receipt, transaction authorization
14 confirmation message, shipping notification and/or the like. In such messages, the
15 payment network may provide the URL to the user device. The user may navigate to the
16 URL on the user's device to obtain alerts regarding the user's purchase, as well as other
17 information such as offers, coupons, related products, rewards notifications, and/or the
18 like. An example listing of a checkout data 4517, substantially in the form of XML-
19 formatted data, is provided below:

```
20 <?XML version = "1.0" encoding = "UTF-8"?>
21 <checkout_data>
22     <session_ID>4NFU4RG94</session_ID>
23     <timestamp>2011-02-22 15:22:43</timestamp>
24     <expiry_lapse>00:00:30</expiry_lapse>
25     <transaction_cost>$34.78</transaction_cost>
26     <alerts_URL>www.merchant.com/shopcarts.php?sessionID=4NFU4RG94</alert
27 s_URL>
28     <!--optional data-->
29     <user_ID>john.q.public@gmail.com</user_ID>
30     <client_details>
31         <client_IP>192.168.23.126</client_IP>
```

```

1         <client_type>smartphone</client_type>
2         <client_model>HTC Hero</client_model>
3         <OS>Android 2.2</OS>
4         <app_installed_flag>true</app_installed_flag>
5     </client_details>
6     <purchase_details>
7         <num_products>1</num_products>
8         <product>
9             <product_type>book</product_type>
10            <product_params>
11                <product_title>XML for dummies</product_title>
12                <ISBN>938-2-14-168710-0</ISBN>
13                <edition>2nd ed.</edition>
14                <cover>hardbound</cover>
15                <seller>bestbuybooks</seller>
16            </product_params>
17            <quantity>1</quantity>
18        </product>
19    </purchase_details>
20    <offers_details>
21        <num_offers>1</num_offers>
22        <product>
23            <product_type>book</product_type>
24            <product_params>
25                <product_title>Here's more XML</product_title>
26                <ISBN>922-7-14-165720-1</ISBN>
27                <edition>1nd ed.</edition>
28                <cover>hardbound</cover>
29                <seller>digibooks</seller>
30            </product_params>
31            <quantity>1</quantity>
32        </product>
33    </offers_details>
34    <secure_element>www.merchant.com/securedyn/0394733/123.png</secure_el
35    ement>
36    <merchant_params>
37        <merchant_id>3FBCR4INC</merchant_id>
38        <merchant_name>Books & Things, Inc.</merchant_name>
39        <merchant_auth_key>1NNF484MCP59CHB27365</merchant_auth_key>
40    </merchant_params>
41    <checkout_data>
42

```

[00415] In alternate embodiments, the merchant server may invoke a component to generate checkout data, such as the example PoS terminal checkout discussed above with reference to FIGURE 3A. Upon obtaining the checkout data, e.g., 4517, the PoS client may render and display, e.g., 4518, the checkout data for the user.

[00416] FIGURE 46 shows a logic flow diagram illustrating example aspects of a user purchase checkout in some embodiments of the MCB-Platform, e.g., a User Purchase Checkout (“UPC”) component 4600. In some embodiments, a user may desire to purchase a product, service, offering, and/or the like (“product”), from a merchant via

1 a merchant online site or in the merchant's store. The user may communicate with a
2 merchant/acquirer ("merchant") server via a PoS client. For example, the user may
3 provide user input, e.g., 4601, into the client indicating the user's desire to purchase the
4 product. The client may generate a checkout request, e.g., 4602, and provide the
5 checkout request to the merchant server. In some embodiments, the merchant server
6 may obtain the checkout request from the client, and extract the checkout detail (e.g.,
7 XML data) from the checkout request. For example, the merchant server may utilize a
8 parser such as the example parsers described below in the discussion with reference to
9 FIGURE 51. Based on parsing the checkout request, the merchant server may extract
10 product data (e.g., product identifiers), as well as available PoS client data, from the
11 checkout request. In some embodiments, using the product data, the merchant server
12 may query, e.g., 4603, a merchant/acquirer ("merchant") database to obtain product
13 data, e.g., 4604, such as product information, product pricing, sales tax, offers,
14 discounts, rewards, and/or other information to process the purchase transaction
15 and/or provide value-added services for the user. In some embodiments, in response to
16 obtaining the product data, the merchant server may generate, e.g., 4605, checkout data
17 to provide, e.g., 4606, for the PoS client. In some embodiments, the merchant server
18 may invoke a component to generate checkout data, such as the example POS terminals
19 discussed above with reference to FIGURE 3A. Upon obtaining the checkout data, the
20 PoS client may render and display, e.g., 4607, the checkout data for the user.

21 **[00417]** FIGURES 47A-B show data flow diagrams illustrating an example
22 purchase transaction authorization procedure in some embodiments of the MCB-
23 Platform. With reference to FIGURE 47A, in some embodiments, a user, e.g., 4701a,
24 may wish to utilize a virtual wallet account to purchase a product, service, offering,

1 and/or the like (“product”), from a merchant via a merchant online site or in the
2 merchant’s store. The user may utilize a physical card, or a user wallet device, e.g.,
3 4701b, to access the user’s virtual wallet account. For example, the user wallet device may be
4 a personal/laptop computer, cellular telephone, smartphone, tablet, eBook reader,
5 netbook, gaming console, and/or the like. The user may provide a wallet access input,
6 e.g., 4711 into the user wallet device. In various embodiments, the user input may include,
7 but not be limited to: a single tap (e.g., a one-tap mobile app purchasing embodiment)
8 of a touchscreen interface, keyboard entry, card swipe, activating a RFID/NFC enabled
9 hardware device (e.g., electronic card having multiple accounts, smartphone, tablet,
10 etc.) within the user device, mouse clicks, depressing buttons on a joystick/game
11 console, voice commands, single/multi-touch gestures on a touch-sensitive interface,
12 touching user interface elements on a touch-sensitive display, and/or the like. In some
13 embodiments, the user wallet device may authenticate the user based on the user’s
14 wallet access input, and provide virtual wallet features for the user. In some
15 embodiments, the user wallet device may invoke a component to ensure the security of
16 the user’s wallet, such as the consumer wallet credentials discussed above with reference
17 to FIGURES 3C and 4B.

18 **[00418]** In some embodiments, upon authenticating the user for access to virtual
19 wallet features, the user wallet device may provide a transaction authorization input,
20 e.g., 4714, to a point-of-sale (“PoS”) client, e.g., 4702. For example, the user wallet device
21 may communicate with the PoS client via Bluetooth, Wi-Fi, cellular communication, one- or
22 two-way near-field communication (“NFC”), and/or the like. In embodiments where the user
23 utilizes a plastic card instead of the user wallet device, the user may swipe the plastic card at
24 the PoS client to transfer information from the plastic card into the PoS client. For example,

1 the PoS client may obtain, as transaction authorization input 4714, track 1 data from the
 2 user's plastic card (e.g., credit card, debit card, prepaid card, charge card, etc.), such as
 3 the example track 1 data provided below:

```
4 %B123456789012345^PUBLIC/J.Q.^99011200000000000000**901*****?*
5 (wherein '123456789012345' is the card number of 'J.Q. Public' and has a CVV
6 number of 901. '990112' is a service code, and *** represents decimal digits
7 which change randomly each time the card is used.)
8
```

9 **[00419]** In embodiments where the user utilizes a user wallet device, the user
 10 wallet device may provide payment information to the PoS client, formatted according
 11 to a data formatting protocol appropriate to the communication mechanism employed
 12 in the communication between the user wallet device and the PoS client. An example
 13 listing of transaction authorization input 4714, substantially in the form of XML-
 14 formatted data, is provided below:

```
15 <?XML version = "1.0" encoding = "UTF-8"?>
16 <transaction_authorization_input>
17   <payment_data>
18     <account>
19       <charge_priority>1</charge_priority>
20       <charge_ratio>40%</charge_ratio>
21       <account_number>123456789012345</account_number>
22       <account_name>John Q. Public</account_name>
23       <bill_add>987 Green St #456, Chicago, IL
24     94652</bill_add>
25       <ship_add>987 Green St #456, Chicago, IL
26     94652</ship_add>
27       <CVV>123</CVV>
28     </account>
29     <account>
30       <charge_priority>1</charge_priority>
31       <charge_ratio>60%</charge_ratio>
32       <account_number>234567890123456</account_number>
33       <account_name>John Q. Public</account_name>
34       <bill_add>987 Green St #456, Chicago, IL
35     94652</bill_add>
36       <ship_add>987 Green St #456, Chicago, IL
37     94652</ship_add>
38       <CVV>173</CVV>
39     </account>
40     <account>
41       <charge_priority>2</charge_priority>
42       <charge_ratio>100%</charge_ratio>
43       <account_number>345678901234567</account_number>
44       <account_name>John Q. Public</account_name>
45       <bill_add>987 Green St #456, Chicago, IL
46     94652</bill_add>
```

```

1           <ship_add>987 Green St #456, Chicago, IL
2 94652</ship_add>
3           <CVV>695</CVV>
4           </account>
5       </payment_data>
6       <!--optional data-->
7       <timestamp>2011-02-22 15:22:43</timestamp>
8       <expiry_lapse>00:00:30</expiry_lapse>
9       <secure_key>0445329070598623487956543322</secure_key>
10      <alerts_track_flag>TRUE</alerts_track_flag>
11      <wallet_device_details>
12          <device_IP>192.168.23.126</client_IP>
13          <device_type>smartphone</client_type>
14          <device_model>HTC Hero</client_model>
15          <OS>Android 2.2</OS>
16          <wallet_app_installed_flag>true</wallet_app_installed_flag>
17      </wallet_device_details>
18  </transaction_authorization_input>
19

```

[00420] In some embodiments, the PoS client may generate a card authorization request, e.g., 4715, using the obtained transaction authorization input from the user wallet device, and/or product/checkout data (see, e.g., FIGURE 45, 4515-4517). An example listing of a card authorization request 4715, substantially in the form of a HTTP(S) POST message including XML-formatted data, is provided below:

```

25 POST /authorizationrequests.php HTTP/1.1
26 Host: www.acquirer.com
27 Content-Type: Application/XML
28 Content-Length: 1306
29 <?XML version = "1.0" encoding = "UTF-8"?>
30 <card_authorization_request>
31     <session_ID>4NFU4RG94</order_ID>
32     <timestamp>2011-02-22 15:22:43</timestamp>
33     <expiry>00:00:30</expiry>
34     <alerts_URL>www.merchant.com/shopcars.php?sessionID=AEBB4356</alerts
35 _URL>
36     <!--optional data-->
37     <user_ID>john.q.public@gmail.com</user_ID>
38     <PoS_details>
39         <PoS_IP>192.168.23.126</client_IP>
40         <PoS_type>smartphone</client_type>
41         <PoS_model>HTC Hero</client_model>
42         <OS>Android 2.2</OS>
43         <app_installed_flag>true</app_installed_flag>
44     </PoS_details>
45     <purchase_details>
46         <num_products>1</num_products>
47         <product>
48             <product_type>book</product_type>
49             <product_params>
50                 <product_title>XML for dummies</product_title>
51                 <ISBN>938-2-14-168710-0</ISBN>
52                 <edition>2nd ed.</edition>
53                 <cover>hardbound</cover>
54                 <seller>bestbuybooks</seller>

```

```

1             </product_params>
2             <quantity>1</quantity>
3         </product>
4     </purchase_details>
5     <merchant_params>
6         <merchant_id>3FBCR4INC</merchant_id>
7         <merchant_name>Books & Things, Inc.</merchant_name>
8         <merchant_auth_key>1NNF484MCP59CHB27365</merchant_auth_key>
9     </merchant_params>
10    <account_params>
11        <account_name>John Q. Public</account_name>
12        <account_type>credit</account_type>
13        <account_num>123456789012345</account_num>
14        <billing_address>123 Green St., Norman, OK
15    98765</billing_address>
16        <phone>123-456-7809</phone>
17        <sign>/jq</sign>
18        <confirm_type>email</confirm_type>
19        <contact_info>john.q.public@gmail.com</contact_info>
20    </account_params>
21    <shipping_info>
22        <shipping_adress>same as billing</shipping_address>
23        <ship_type>expedited</ship_type>
24        <ship_carrier>FedEx</ship_carrier>
25        <ship_account>123-45-678</ship_account>
26        <tracking_flag>true</tracking_flag>
27        <sign_flag>false</sign_flag>
28    </shipping_info>
29    </card_authorization_request>
30

```

[00421] In some embodiments, the card authorization request generated by the user device may include a minimum of information required to process the purchase transaction. For example, this may improve the efficiency of communicating the purchase transaction request, and may also advantageously improve the privacy protections provided to the user and/or merchant. For example, in some embodiments, the card authorization request may include at least a session ID for the user's shopping session with the merchant. The session ID may be utilized by any component and/or entity having the appropriate access authority to access a secure site on the merchant server to obtain alerts, reminders, and/or other data about the transaction(s) within that shopping session between the user and the merchant. In some embodiments, the PoS client may provide the generated card authorization request to the merchant server, e.g., 4716. The merchant server may forward the card authorization request to a pay gateway server, e.g., 4704a, for routing the card authorization request to the appropriate

1 payment network for payment processing. For example, the pay gateway server may be
2 able to select from payment networks, such as Visa, Mastercard, American Express,
3 Paypal, etc., to process various types of transactions including, but not limited to: credit
4 card, debit card, prepaid card, B2B and/or like transactions. In some embodiments, the
5 merchant server may query a database, e.g., merchant/acquirer database 4703b, for a
6 network address of the payment gateway server, for example by using a portion of a user
7 payment card number, or a user ID (such as an email address) as a keyword for the database
8 query. For example, the merchant server may issue PHP/SQL commands to query a
9 database table (such as FIGURE 51, Pay Gateways 5119n) for a URL of the pay gateway
10 server. An example payment gateway address query 4717, substantially in the form of
11 PHP/SQL commands, is provided below:

```
12 <?PHP
13 header('Content-Type: text/plain');
14 mysql_connect("254.93.179.112",$DBserver,$password); // access database server
15 mysql_select_db("MCB-Platform_DB.SQL"); // select database table to search
16 //create query
17 $query = "SELECT paygate_id paygate_address paygate_URL paygate_name FROM
18 PayGatewayTable WHERE card_num LIKE '%" $cardnum";
19 $result = mysql_query($query); // perform the search query
20 mysql_close("MCB-Platform_DB.SQL"); // close database access
21 ?>
22
```

23 **[00422]** In response, the merchant/acquirer database may provide the requested
24 payment gateway address, e.g., 4718. The merchant server may forward the card
25 authorization request to the pay gateway server using the provided address, e.g., 4719. In
26 some embodiments, upon receiving the card authorization request from the merchant server,
27 the pay gateway server may invoke a component to provide one or more services
28 associated with purchase transaction authorization. For example, the pay gateway
29 server may invoke components for fraud prevention, loyalty and/or rewards, and/or
30 other services for which the user-merchant combination is authorized. In some

1 embodiments, the pay gateway server may invoke a component to provide point-of-sale
2 value-add services. The pay gateway server may forward the card authorization request
3 to a pay network server, e.g., 4705a, for payment processing. For example, the pay
4 gateway server may be able to select from payment networks, such as Visa, Mastercard,
5 American Express, Paypal, etc., to process various types of transactions including, but
6 not limited to: credit card, debit card, prepaid card, B2B and/or like transactions. In
7 some embodiments, the pay gateway server may query a database, e.g., pay gateway
8 database 4704b, for a network address of the payment network server, for example by using
9 a portion of a user payment card number, or a user ID (such as an email address) as a
10 keyword for the database query. For example, the pay gateway server may issue
11 PHP/SQL commands to query a database table (such as FIGURE 51, Pay Gateways
12 5119n) for a URL of the pay network server. An example payment network address
13 query 4721, substantially in the form of PHP/SQL commands, is provided below:

```
14 <?PHP
15 header('Content-Type: text/plain');
16 mysql_connect("254.93.179.112",$DBserver,$password); // access database server
17 mysql_select_db("MCB-Platform_DB.SQL"); // select database table to search
18 //create query
19 $query = "SELECT payNET_id payNET_address payNET_URL payNET_name FROM
20 PayGatewayTable WHERE card_num LIKE '%" $cardnum";
21 $result = mysql_query($query); // perform the search query
22 mysql_close("MCB-Platform_DB.SQL"); // close database access
23 ?>
24
```

25 **[00423]** In response, the payment gateway database may provide the requested
26 payment network address, e.g., 4722. The pay gateway server may forward the card
27 authorization request to the pay network server using the provided address, e.g., 4723.

28 **[00424]** With reference to FIGURE 47B, in some embodiments, the pay network
29 server may process the transaction so as to transfer funds for the purchase into an
30 account stored on an acquirer of the merchant. For example, the acquirer may be a

1 financial institution maintaining an account of the merchant. For example, the
2 proceeds of transactions processed by the merchant may be deposited into an account
3 maintained by at a server of the acquirer.

4 **[00425]** In some embodiments, the pay network server may generate a query, e.g.,
5 4724, for issuer server(s) corresponding to the user-selected payment options. For
6 example, the user's account may be linked to one or more issuer financial institutions
7 ("issuers"), such as banking institutions, which issued the account(s) for the user. For
8 example, such accounts may include, but not be limited to: credit card, debit card,
9 prepaid card, checking, savings, money market, certificates of deposit, stored (cash)
10 value accounts and/or the like. Issuer server(s), e.g., 4706a, of the issuer(s) may
11 maintain details of the user's account(s). In some embodiments, a database, e.g., pay
12 network database 4705b, may store details of the issuer server(s) associated with the
13 issuer(s). In some embodiments, the pay network server may query a database, e.g., pay
14 network database 4705b, for a network address of the issuer(s) server(s), for example by
15 using a portion of a user payment card number, or a user ID (such as an email address) as a
16 keyword for the database query. For example, the merchant server may issue PHP/SQL
17 commands to query a database table (such as FIGURE 51, Issuers 5119e) for network
18 address(es) of the issuer(s) server(s). An example issuer server address(es) query 4724,
19 substantially in the form of PHP/SQL commands, is provided below:

```
20 <?PHP
21 header('Content-Type: text/plain');
22 mysql_connect("254.93.179.112",$DBserver,$password); // access database server
23 mysql_select_db("MCB-Platform_DB.SQL"); // select database table to search
24 //create query
25 $query = "SELECT issuer_id issuer_address issuer_URL issuer_name FROM
26 IssuersTable WHERE card_num LIKE '%" $cardnum";
27 $result = mysql_query($query); // perform the search query
28 mysql_close("MCB-Platform_DB.SQL"); // close database access
29 ?>
30
```

1 **[00426]** In response to obtaining the issuer server query, e.g., 4724, the pay
 2 network database may provide, e.g., 4725, the requested issuer server data to the pay
 3 network server. In some embodiments, the pay network server may utilize the issuer
 4 server data to generate funds authorization request(s), e.g., 4726, for each of the issuer
 5 server(s) selected based on the pre-defined payment settings associated with the user's
 6 virtual wallet, and/or the user's payment options input, and provide the funds
 7 authorization request(s) to the issuer server(s). In some embodiments, the funds
 8 authorization request(s) may include details such as, but not limited to: the costs to the
 9 user involved in the transaction, card account details of the user, user billing and/or
 10 shipping information, and/or the like. An example listing of a funds authorization
 11 request 4726, substantially in the form of a HTTP(S) POST message including XML-
 12 formatted data, is provided below:

```

13 POST /fundsauthorizationrequest.php HTTP/1.1
14 Host: www.issuer.com
15 Content-Type: Application/XML
16 Content-Length: 624
17 <?XML version = "1.0" encoding = "UTF-8"?>
18 <funds_authorization_request>
19     <query_ID>VNEI39FK</query_ID>
20     <timestamp>2011-02-22 15:22:44</timestamp>
21     <transaction_cost>$22.61</transaction_cost>
22     <account_params>
23         <account_type>checking</account_type>
24         <account_num>1234567890123456</account_num>
25     </account_params>
26     <!--optional parameters-->
27     <purchase_summary>
28         <num_products>1</num_products>
29         <product>
30             <product_summary>Book - XML for
31 dummies</product_summary>
32             <product_quantity>1</product_quantity?
33         </product>
34     </purchase_summary>
35     <merchant_params>
36         <merchant_id>3FBCR4INC</merchant_id>
37         <merchant_name>Books & Things, Inc.</merchant_name>
38         <merchant_auth_key>1NNF484MCP59CHB27365</merchant_auth_key>
39     </merchant_params>
40 </funds_authorization_request>
41

```

1 **[00427]** In some embodiments, an issuer server may parse the authorization
2 request(s), and based on the request details may query a database, e.g., user profile
3 database 4706b, for data associated with an account linked to the user. For example, the
4 merchant server may issue PHP/SQL commands to query a database table (such as
5 FIGURE 51, Accounts 5119g) for user account(s) data. An example user account(s)
6 query 4727, substantially in the form of PHP/SQL commands, is provided below:

```
7  <?PHP
8  header('Content-Type: text/plain');
9  mysql_connect("254.93.179.112",$DBserver,$password); // access database server
10 mysql_select_db("MCB-Platform_DB.SQL"); // select database table to search
11 //create query
12 $query = "SELECT issuer user_id user_name user_balance account_type FROM
13 AccountsTable WHERE account_num LIKE '%" $accountnum";
14 $result = mysql_query($query); // perform the search query
15 mysql_close("MCB-Platform_DB.SQL"); // close database access
16 ?>
17
```

18 **[00428]** In some embodiments, on obtaining the user account(s) data, e.g., 4728,
19 the issuer server may determine whether the user can pay for the transaction using
20 funds available in the account, 4729. For example, the issuer server may determine
21 whether the user has a sufficient balance remaining in the account, sufficient credit
22 associated with the account, and/or the like. Based on the determination, the issuer
23 server(s) may provide a funds authorization response, e.g., 4730, to the pay network
24 server. For example, the issuer server(s) may provide a HTTP(S) POST message similar
25 to the examples above. In some embodiments, if at least one issuer server determines
26 that the user cannot pay for the transaction using the funds available in the account, the
27 pay network server may request payment options again from the user (e.g., by providing
28 an authorization fail message to the user device and requesting the user device to
29 provide new payment options), and re-attempt authorization for the purchase
30 transaction. In some embodiments, if the number of failed authorization attempts

1 exceeds a threshold, the pay network server may abort the authorization process, and
2 provide an “authorization fail” message to the merchant server, user device and/or
3 client.

4 **[00429]** In some embodiments, the pay network server may obtain the funds
5 authorization response including a notification of successful authorization, and parse
6 the message to extract authorization details. Upon determining that the user possesses
7 sufficient funds for the transaction, e.g., 4731, the pay network server may invoke a
8 component to provide value-add services for the user. In some embodiments, the pay
9 gateway server may invoke a component to provide point-of-sale value-add services. In
10 various embodiments, such value-add services may be provided at any point in the
11 purchase transaction process, including before the pay gateway server(s) and/or pay
12 network server(s) obtain verification from the issuer server(s) that the user has funds
13 sufficient for the transaction to be processed, or prior to obtaining such verification.

14 **[00430]** In some embodiments, the pay network server may generate a transaction
15 data record from the authorization request and/or authorization response, and store the
16 details of the transaction and authorization relating to the transaction in a transactions
17 database. For example, the pay network server may issue PHP/SQL commands to store
18 the data to a database table (such as FIGURE 51, Transactions 51190). An example
19 transaction store command, substantially in the form of PHP/SQL commands, is
20 provided below:

```
21 <?PHP
22 header('Content-Type: text/plain');
23 mysql_connect("254.92.185.103",$DBserver,$password); // access database server
24 mysql_select("MCB-Platform_DB.SQL"); // select database to append
25 mysql_query("INSERT INTO TransactionsTable (PurchasesTable (timestamp,
26 purchase_summary_list, num_products, product_summary, product_quantity,
27 transaction_cost, account_params_list, account_name, account_type, account_num,
```

```

1  billing_addres, zipcode, phone, sign, merchant_params_list, merchant_id,
2  merchant_name, merchant_auth_key)
3  VALUES (time(), $purchase_summary_list, $num_products, $product_summary,
4  $product_quantity, $transaction_cost, $account_params_list, $account_name,
5  $account_type, $account_num, $billing_addres, $zipcode, $phone, $sign,
6  $merchant_params_list, $merchant_id, $merchant_name, $merchant_auth_key)"); //
7  add data to table in database
8  mysql_close("MCB-Platform_DB.SQL"); // close connection to database
9  ?>
10

```

11 **[00431]** In some embodiments, the pay network server may forward a transaction
12 authorization response, e.g., 4732, to the user wallet device, PoS client, and/or merchant
13 server. The merchant may obtain the transaction authorization response, and
14 determine from it that the user possesses sufficient funds in the card account to conduct
15 the transaction. The merchant server may add a record of the transaction for the user to
16 a batch of transaction data relating to authorized transactions. For example, the
17 merchant may append the XML data pertaining to the user transaction to an XML data
18 file comprising XML data for transactions that have been authorized for various users,
19 e.g., 4733, and store the XML data file, e.g., 4734, in a database, e.g., merchant database
20 404. For example, a batch XML data file may be structured similar to the example XML
21 data structure template provided below:

```

22  <?XML version = "1.0" encoding = "UTF-8"?>
23  <merchant_data>
24      <merchant_id>3FBCR4INC</merchant_id>
25      <merchant_name>Books & Things, Inc.</merchant_name>
26      <merchant_auth_key>1NNF484MCP59CHB27365</merchant_auth_key>
27      <account_number>123456789</account_number>
28  </merchant_data>
29  <transaction_data>
30      <transaction 1>
31          ...
32      </transaction 1>
33      <transaction 2>
34          ...
35      </transaction 2>
36      .
37      .
38      .
39      <transaction n>
40          ...
41      </transaction n>
42  </transaction_data>
43

```

1 **[00432]** In some embodiments, the server may also generate a purchase receipt,
2 e.g., 4733, and provide the purchase receipt to the client, e.g., 4735. The client may
3 render and display, e.g., 4736, the purchase receipt for the user. In some embodiments,
4 the user's wallet device may also provide a notification of successful authorization to the
5 user. For example, the PoS client/user device may render a webpage, electronic
6 message, text / SMS message, buffer a voicemail, emit a ring tone, and/or play an audio
7 message, etc., and provide output including, but not limited to: sounds, music, audio,
8 video, images, tactile feedback, vibration alerts (e.g., on vibration-capable client devices
9 such as a smartphone etc.), and/or the like.

10 **[00433]** FIGURES 48A-B show logic flow diagrams illustrating example aspects of
11 purchase transaction authorization in some embodiments of the MCB-Platform, e.g., a
12 Purchase Transaction Authorization ("PTA") component 4800. With reference to
13 FIGURE 48A, in some embodiments, a user may wish to utilize a virtual wallet account
14 to purchase a product, service, offering, and/or the like ("product"), from a merchant via
15 a merchant online site or in the merchant's store. The user may utilize a physical card,
16 or a user wallet device to access the user's virtual wallet account. For example, the user
17 wallet device may be a personal/laptop computer, cellular telephone, smartphone, tablet,
18 eBook reader, netbook, gaming console, and/or the like. The user may provide a wallet
19 access input, e.g., 4801, into the user wallet device. In various embodiments, the user
20 input may include, but not be limited to: a single tap (e.g., a one-tap mobile app
21 purchasing embodiment) of a touchscreen interface, keyboard entry, card swipe,
22 activating a RFID/NFC enabled hardware device (e.g., electronic card having multiple
23 accounts, smartphone, tablet, etc.) within the user device, mouse clicks, depressing
24 buttons on a joystick/game console, voice commands, single/multi-touch gestures on a

1 touch-sensitive interface, touching user interface elements on a touch-sensitive display,
2 and/or the like. In some embodiments, the user wallet device may authenticate the user
3 based on the user's wallet access input, and provide virtual wallet features for the user,
4 e.g., 4802-4803. In some embodiments, the user wallet device may invoke a component
5 to ensure the security of the user's wallet, such as the consumer wallet credentials
6 discussed above with reference to FIGURES 3B and 4C.

7 **[00434]** In some embodiments, upon authenticating the user for access to virtual
8 wallet features, the user wallet device may provide a transaction authorization input,
9 e.g., 4804, to a point-of-sale ("PoS") client. For example, the user wallet device may
10 communicate with the PoS client via Bluetooth, Wi-Fi, cellular communication, one- or two-
11 way near-field communication ("NFC"), and/or the like. In embodiments where the user
12 utilizes a plastic card instead of the user wallet device, the user may swipe the plastic card at
13 the PoS client to transfer information from the plastic card into the PoS client. In
14 embodiments where the user utilizes a user wallet device, the user wallet device may
15 provide payment information to the PoS client, formatted according to a data formatting
16 protocol appropriate to the communication mechanism employed in the communication
17 between the user wallet device and the PoS client.

18 **[00435]** In some embodiments, the PoS client may obtain the transaction
19 authorization input, and parse the input to extract payment information from the
20 transaction authorization input, e.g., 4805. For example, the PoS client may utilize a
21 parser, such as the example parsers provided below in the discussion with reference to
22 FIGURE 51. The PoS client may generate a card authorization request, e.g., 4806, using

1 the obtained transaction authorization input from the user wallet device, and/or
2 product/checkout data (see, e.g., FIGURE 45, 4515-4517).

3 **[00436]** In some embodiments, the PoS client may provide the generated card
4 authorization request to the merchant server. The merchant server may forward the
5 card authorization request to a pay gateway server, for routing the card authorization
6 request to the appropriate payment network for payment processing. For example, the
7 pay gateway server may be able to select from payment networks, such as Visa,
8 Mastercard, American Express, Paypal, etc., to process various types of transactions
9 including, but not limited to: credit card, debit card, prepaid card, B2B and/or like
10 transactions. In some embodiments, the merchant server may query a database, e.g.,
11 4808, for a network address of the payment gateway server, for example by using a portion of
12 a user payment card number, or a user ID (such as an email address) as a keyword for the
13 database query. In response, the merchant/acquirer database may provide the requested
14 payment gateway address, e.g., 4810. The merchant server may forward the card
15 authorization request to the pay gateway server using the provided address. In some
16 embodiments, upon receiving the card authorization request from the merchant server, the
17 pay gateway server may invoke a component to provide one or more service associated
18 with purchase transaction authorization, e.g., 4811. For example, the pay gateway server
19 may invoke components for fraud prevention (see e.g., VerifyChat, FIGURE 3E), loyalty
20 and/or rewards, and/or other services for which the user-merchant combination is
21 authorized.

22 **[00437]** The pay gateway server may forward the card authorization request to a
23 pay network server for payment processing, e.g., 4814. For example, the pay gateway

1 server may be able to select from payment networks, such as Visa, Mastercard,
2 American Express, Paypal, etc., to process various types of transactions including, but
3 not limited to: credit card, debit card, prepaid card, B2B and/or like transactions. In
4 some embodiments, the pay gateway server may query a database, e.g., 4812, for a
5 network address of the payment network server, for example by using a portion of a user
6 payment card number, or a user ID (such as an email address) as a keyword for the database
7 query. In response, the payment gateway database may provide the requested payment
8 network address, e.g., 4813. The pay gateway server may forward the card authorization
9 request to the pay network server using the provided address, e.g., 4814.

10 **[00438]** With reference to FIGURE 48B, in some embodiments, the pay network
11 server may process the transaction so as to transfer funds for the purchase into an
12 account stored on an acquirer of the merchant. For example, the acquirer may be a
13 financial institution maintaining an account of the merchant. For example, the
14 proceeds of transactions processed by the merchant may be deposited into an account
15 maintained by at a server of the acquirer. In some embodiments, the pay network
16 server may generate a query, e.g., 4815, for issuer server(s) corresponding to the user-
17 selected payment options. For example, the user's account may be linked to one or
18 more issuer financial institutions ("issuers"), such as banking institutions, which issued
19 the account(s) for the user. For example, such accounts may include, but not be limited
20 to: credit card, debit card, prepaid card, checking, savings, money market, certificates of
21 deposit, stored (cash) value accounts and/or the like. Issuer server(s) of the issuer(s)
22 may maintain details of the user's account(s). In some embodiments, a database, e.g., a
23 pay network database, may store details of the issuer server(s) associated with the
24 issuer(s). In some embodiments, the pay network server may query a database, e.g.,

1 4815, for a network address of the issuer(s) server(s), for example by using a portion of a user
2 payment card number, or a user ID (such as an email address) as a keyword for the database
3 query.

4 **[00439]** In response to obtaining the issuer server query, the pay network database
5 may provide, e.g., 4816, the requested issuer server data to the pay network server. In
6 some embodiments, the pay network server may utilize the issuer server data to
7 generate funds authorization request(s), e.g., 4817, for each of the issuer server(s)
8 selected based on the pre-defined payment settings associated with the user's virtual
9 wallet, and/or the user's payment options input, and provide the funds authorization
10 request(s) to the issuer server(s). In some embodiments, the funds authorization
11 request(s) may include details such as, but not limited to: the costs to the user involved
12 in the transaction, card account details of the user, user billing and/or shipping
13 information, and/or the like. In some embodiments, an issuer server may parse the
14 authorization request(s), e.g., 4818, and based on the request details may query a
15 database, e.g., 4819, for data associated with an account linked to the user.

16 **[00440]** In some embodiments, on obtaining the user account(s) data, e.g., 4820,
17 the issuer server may determine whether the user can pay for the transaction using
18 funds available in the account, e.g., 4821. For example, the issuer server may determine
19 whether the user has a sufficient balance remaining in the account, sufficient credit
20 associated with the account, and/or the like. Based on the determination, the issuer
21 server(s) may provide a funds authorization response, e.g., 4822, to the pay network
22 server. In some embodiments, if at least one issuer server determines that the user
23 cannot pay for the transaction using the funds available in the account, the pay network

1 server may request payment options again from the user (e.g., by providing an
2 authorization fail message to the user device and requesting the user device to provide
3 new payment options), and re-attempt authorization for the purchase transaction. In
4 some embodiments, if the number of failed authorization attempts exceeds a threshold,
5 the pay network server may abort the authorization process, and provide an
6 “authorization fail” message to the merchant server, user device and/or client.

7 **[00441]** In some embodiments, the pay network server may obtain the funds
8 authorization response including a notification of successful authorization, and parse
9 the message to extract authorization details. Upon determining that the user possesses
10 sufficient funds for the transaction, e.g., 4823, the pay network server may invoke a
11 component to provide value-add services for the user, e.g., 4823.

12 **[00442]** In some embodiments, the pay network server may forward a transaction
13 authorization response to the user wallet device, PoS client, and/or merchant server.
14 The merchant may parse, e.g., 4824, the transaction authorization response, and
15 determine from it that the user possesses sufficient funds in the card account to conduct
16 the transaction, e.g., 4825, option “Yes.” The merchant server may add a record of the
17 transaction for the user to a batch of transaction data relating to authorized
18 transactions. For example, the merchant may append the XML data pertaining to the
19 user transaction to an XML data file comprising XML data for transactions that have
20 been authorized for various users, e.g., 4826, and store the XML data file, e.g., 4827, in a
21 database. In some embodiments, the server may also generate a purchase receipt, e.g.,
22 4828, and provide the purchase receipt to the client. The client may render and display,
23 e.g., 4829, the purchase receipt for the user. In some embodiments, the user’s wallet

1 device may also provide a notification of successful authorization to the user. For
2 example, the PoS client/user device may render a webpage, electronic message, text /
3 SMS message, buffer a voicemail, emit a ring tone, and/or play an audio message, etc.,
4 and provide output including, but not limited to: sounds, music, audio, video, images,
5 tactile feedback, vibration alerts (e.g., on vibration-capable client devices such as a
6 smartphone etc.), and/or the like.

7 **[0043]** FIGURES 49A-B show data flow diagrams illustrating an example
8 purchase transaction clearance procedure in some embodiments of the MCB-Platform.
9 With reference to FIGURE 49A, in some embodiments, a merchant server, e.g., 4903a,
10 may initiate clearance of a batch of authorized transactions. For example, the merchant
11 server may generate a batch data request, e.g., 4911, and provide the request, to a
12 merchant database, e.g., 4903b. For example, the merchant server may utilize
13 PHP/SQL commands similar to the examples provided above to query a relational
14 database. In response to the batch data request, the database may provide the
15 requested batch data, e.g., 4912. The server may generate a batch clearance request,
16 e.g., 4913, using the batch data obtained from the database, and provide, e.g., 4914, the
17 batch clearance request to an acquirer server, e.g., 4907a. For example, the merchant
18 server may provide a HTTP(S) POST message including XML-formatted batch data in
19 the message body for the acquirer server. The acquirer server may generate, e.g., 4915, a
20 batch payment request using the obtained batch clearance request, and provide, e.g.,
21 4918, the batch payment request to the pay network server, e.g., 4905a. The pay
22 network server may parse the batch payment request, and extract the transaction data
23 for each transaction stored in the batch payment request, e.g., 4919. The pay network
24 server may store the transaction data, e.g., 4920, for each transaction in a database, e.g.,

1 pay network database 4905b. In some embodiments, the pay network server may
2 invoke a component to provide value-add analytics services based on analysis of the
3 transactions of the merchant for whom the MCB-Platform is clearing purchase
4 transactions. For example, the pay network server may invoke a component such as the
5 example card transaction-based analytics component discussed above with reference to
6 FIGURES 27-28. Thus, in some embodiments, the pay network server may provide
7 analytics-based value-added services for the merchant and/or the merchant's users.

8 **[00444]** With reference to FIGURE 49B, in some embodiments, for each extracted
9 transaction, the pay network server may query, e.g., 4923, a database, e.g., pay network
10 database 4905b, for an address of an issuer server. For example, the pay network server
11 may utilize PHP/SQL commands similar to the examples provided above. The pay
12 network server may generate an individual payment request, e.g., 4925, for each
13 transaction for which it has extracted transaction data, and provide the individual
14 payment request, e.g., 4925, to the issuer server, e.g., 4906a. For example, the pay
15 network server may provide an individual payment request to the issuer server(s) as a
16 HTTP(S) POST message including XML-formatted data. An example listing of an
17 individual payment request 4925, substantially in the form of a HTTP(S) POST message
18 including XML-formatted data, is provided below:

```
19 POST /paymentrequest.php HTTP/1.1
20 Host: www.issuer.com
21 Content-Type: Application/XML
22 Content-Length: 788
23 <?XML version = "1.0" encoding = "UTF-8"?>
24 <pay_request>
25     <request_ID>CNI4ICNW2</request_ID>
26     <timestamp>2011-02-22 17:00:01</timestamp>
27     <pay_amount>$34.78</pay_amount>
28     <account_params>
29         <account_name>John Q. Public</account_name>
30         <account_type>credit</account_type>
31         <account_num>123456789012345</account_num>
```

```

1         <billing_address>123 Green St., Norman, OK
2 98765</billing_address>
3         <phone>123-456-7809</phone>
4         <sign>/jpg</sign>
5     </account_params>
6     <merchant_params>
7         <merchant_id>3FBCR4INC</merchant_id>
8         <merchant_name>Books & Things, Inc.</merchant_name>
9         <merchant_auth_key>1NNF484MCP59CHB27365</merchant_auth_key>
10    </merchant_params>
11    <purchase_summary>
12        <num_products>1</num_products>
13        <product>
14            <product_summary>Book - XML for
15 dummies</product_summary>
16            <product_quantity>1</product_quantity?
17        </product>
18    </purchase_summary>
19 </pay_request>
20

```

[00445] In some embodiments, the issuer server may generate a payment command, e.g., 4927. For example, the issuer server may issue a command to deduct funds from the user's account (or add a charge to the user's credit card account). The issuer server may issue a payment command, e.g., 4927, to a database storing the user's account information, e.g., user profile database 4906b. The issuer server may provide an individual payment confirmation, e.g., 4928, to the pay network server, which may forward, e.g., 4929, the funds transfer message to the acquirer server. An example listing of an individual payment confirmation 4928, substantially in the form of a HTTP(S) POST message including XML-formatted data, is provided below:

```

30 POST /clearance.php HTTP/1.1
31 Host: www.acquirer.com
32 Content-Type: Application/XML
33 Content-Length: 206
34 <?XML version = "1.0" encoding = "UTF-8"?>
35 <deposit_ack>
36     <request_ID>CNI4ICNW2</request_ID>
37     <clear_flag>true</clear_flag>
38     <timestamp>2011-02-22 17:00:02</timestamp>
39     <deposit_amount>$34.78</deposit_amount>
40 </deposit_ack>
41

```

[00446] In some embodiments, the acquirer server may parse the individual payment confirmation, and correlate the transaction (e.g., using the request_ID field in the example above) to the merchant. The acquirer server may then transfer the funds

1 specified in the funds transfer message to an account of the merchant. For example, the
2 acquirer server may query, e.g. 4930, an acquirer database 4907b for payment ledger
3 and/or merchant account data, e.g., 4931. The acquirer server may utilize payment
4 ledger and/or merchant account data from the acquirer database, along with the
5 individual payment confirmation, to generate updated payment ledger and/or merchant
6 account data, e.g., 4932. The acquirer server may then store, e.g., 4933, the updated
7 payment ledger and/or merchant account data to the acquire database.

8 **[00447]** FIGURES 50A-B show logic flow diagrams illustrating example aspects of
9 purchase transaction clearance in some embodiments of the MCB-Platform, e.g., a
10 Purchase Transaction Clearance (“PTC”) component 5000. With reference to FIGURE
11 50A, in some embodiments, a merchant server may initiate clearance of a batch of
12 authorized transactions. For example, the merchant server may generate a batch data
13 request, e.g., 5001, and provide the request to a merchant database. In response to the
14 batch data request, the database may provide the requested batch data, e.g., 5002. The
15 server may generate a batch clearance request, e.g., 5003, using the batch data obtained
16 from the database, and provide the batch clearance request to an acquirer server. The
17 acquirer server may parse, e.g., 5004, the obtained batch clearance request, and
18 generate, e.g., 5007, a batch payment request using the obtained batch clearance request
19 to provide, the batch payment request to a pay network server. For example, the
20 acquirer server may query, e.g., 5005, an acquirer database for an address of a payment
21 network server, and utilize the obtained address, e.g., 5006, to forward the generated
22 batch payment request to the pay network server.

1 **[00448]** The pay network server may parse the batch payment request obtained
2 from the acquirer server, and extract the transaction data for each transaction stored in
3 the batch payment request, e.g., 5008. The pay network server may store the
4 transaction data, e.g., 5009, for each transaction in a pay network database. In some
5 embodiments, the pay network server may invoke a component, e.g., 5010, to provide
6 analytics based on the transactions of the merchant for whom purchase transaction are
7 being cleared. For example, the pay network server may invoke a component such as
8 the example card transaction-based social data aggregation component discussed above
9 with reference to FIGURES 27-28.

10 **[00449]** With reference to FIGURE 50B, in some embodiments, for each extracted
11 transaction, the pay network server may query, e.g., 5011, a pay network database for an
12 address of an issuer server. The pay network server may generate an individual
13 payment request, e.g., 5013, for each transaction for which it has extracted transaction
14 data, and provide the individual payment request to the issuer server. In some
15 embodiments, the issuer server may parse the individual payment request, e.g., 5014,
16 and generate a payment command, e.g., 5015, based on the parsed individual payment
17 request. For example, the issuer server may issue a command to deduct funds from the
18 user's account (or add a charge to the user's credit card account). The issuer server may
19 issue a payment command, e.g., 5015, to a database storing the user's account
20 information, e.g., a user profile database. The issuer server may provide an individual
21 payment confirmation, e.g., 5017, to the pay network server, which may forward, e.g.,
22 5018, the individual payment confirmation to the acquirer server.

1 **[00450]** In some embodiments, the acquirer server may parse the individual
2 payment confirmation, and correlate the transaction (e.g., using the request_ID field in
3 the example above) to the merchant. The acquirer server may then transfer the funds
4 specified in the funds transfer message to an account of the merchant. For example, the
5 acquirer server may query, e.g. 5019, an acquirer database for payment ledger and/or
6 merchant account data, e.g., 5020. The acquirer server may utilize payment ledger
7 and/or merchant account data from the acquirer database, along with the individual
8 payment confirmation, to generate updated payment ledger and/or merchant account
9 data, e.g., 5021. The acquirer server may then store, e.g., 5022, the updated payment
10 ledger and/or merchant account data to the acquire database.

11 **MCB-Platform Controller**

12 **[00451]** FIGURE 51 shows a block diagram illustrating embodiments of a MCB-
13 Platform controller. In this embodiment, the MCB-Platform controller 5101 may serve
14 to aggregate, process, store, search, serve, identify, instruct, generate, match, and/or
15 facilitate interactions with a computer through various technologies, and/or other
16 related data.

17 **[00452]** Typically, users, which may be people and/or other systems, may engage
18 information technology systems (e.g., computers) to facilitate information processing.
19 In turn, computers employ processors to process information; such processors 5103
20 may be referred to as central processing units (CPU). One form of processor is referred
21 to as a microprocessor. CPUs use communicative circuits to pass binary encoded signals
22 acting as instructions to enable various operations. These instructions may be
23 operational and/or data instructions containing and/or referencing other instructions

1 and data in various processor accessible and operable areas of memory 5129 (e.g.,
2 registers, cache memory, random access memory, etc.). Such communicative
3 instructions may be stored and/or transmitted in batches (e.g., batches of instructions)
4 as programs and/or data components to facilitate desired operations. These stored
5 instruction codes, e.g., programs, may engage the CPU circuit components and other
6 motherboard and/or system components to perform desired operations. One type of
7 program is a computer operating system, which, may be executed by CPU on a
8 computer; the operating system enables and facilitates users to access and operate
9 computer information technology and resources. Some resources that may be employed
10 in information technology systems include: input and output mechanisms through
11 which data may pass into and out of a computer; memory storage into which data may
12 be saved; and processors by which information may be processed. These information
13 technology systems may be used to collect data for later retrieval, analysis, and
14 manipulation, which may be facilitated through a database program. These information
15 technology systems provide interfaces that allow users to access and operate various
16 system components.

17 **[00453]** In one embodiment, the MCB-Platform controller 5101 may be connected
18 to and/or communicate with entities such as, but not limited to: one or more users from
19 user input devices 5111; peripheral devices 5112; an optional cryptographic processor
20 device 5128; and/or a communications network 5113.

21 **[00454]** Networks are commonly thought to comprise the interconnection and
22 interoperation of clients, servers, and intermediary nodes in a graph topology. It should
23 be noted that the term “server” as used throughout this application refers generally to a

1 computer, other device, program, or combination thereof that processes and responds to
2 the requests of remote users across a communications network. Servers serve their
3 information to requesting “clients.” The term “client” as used herein refers generally to a
4 computer, program, other device, user and/or combination thereof that is capable of
5 processing and making requests and obtaining and processing any responses from
6 servers across a communications network. A computer, other device, program, or
7 combination thereof that facilitates, processes information and requests, and/or
8 furthers the passage of information from a source user to a destination user is
9 commonly referred to as a “node.” Networks are generally thought to facilitate the
10 transfer of information from source points to destinations. A node specifically tasked
11 with furthering the passage of information from a source to a destination is commonly
12 called a “router.” There are many forms of networks such as Local Area Networks
13 (LANs), Pico networks, Wide Area Networks (WANs), Wireless Networks (WLANs), etc.
14 For example, the Internet is generally accepted as being an interconnection of a
15 multitude of networks whereby remote clients and servers may access and interoperate
16 with one another.

17 **[00455]** The MCB-Platform controller 5101 may be based on computer systems
18 that may comprise, but are not limited to, components such as: a computer
19 systemization 5102 connected to memory 5129.

20

Computer Systemization

21 **[00456]** A computer systemization 5102 may comprise a clock 5130, central
22 processing unit (“CPU(s)” and/or “processor(s)” (these terms are used interchangeable
23 throughout the disclosure unless noted to the contrary)) 5103, a memory 5129 (e.g., a

1 read only memory (ROM) 5106, a random access memory (RAM) 5105, etc.), and/or an
2 interface bus 5107, and most frequently, although not necessarily, are all interconnected
3 and/or communicating through a system bus 5104 on one or more (mother)board(s)
4 5102 having conductive and/or otherwise transportive circuit pathways through which
5 instructions (e.g., binary encoded signals) may travel to effectuate communications,
6 operations, storage, etc. The computer systemization may be connected to a power
7 source 5186; e.g., optionally the power source may be internal. Optionally, a
8 cryptographic processor 5126 and/or transceivers (e.g., ICs) 5174 may be connected to
9 the system bus. In another embodiment, the cryptographic processor and/or
10 transceivers may be connected as either internal and/or external peripheral devices 5112
11 via the interface bus I/O. In turn, the transceivers may be connected to antenna(s) 5175,
12 thereby effectuating wireless transmission and reception of various communication
13 and/or sensor protocols; for example the antenna(s) may connect to: a Texas
14 Instruments WiLink WL1283 transceiver chip (e.g., providing 802.11n, Bluetooth 3.0,
15 FM, global positioning system (GPS) (thereby allowing MCB-Platform controller to
16 determine its location)); Broadcom BCM4329FKUBG transceiver chip (e.g., providing
17 802.11n, Bluetooth 2.1 + EDR, FM, etc.); a Broadcom BCM4750IUB8 receiver chip (e.g.,
18 GPS); an Infineon Technologies X-Gold 618-PMB9800 (e.g., providing 2G/3G
19 HSDPA/HSUPA communications); and/or the like. The system clock typically has a
20 crystal oscillator and generates a base signal through the computer systemization's
21 circuit pathways. The clock is typically coupled to the system bus and various clock
22 multipliers that will increase or decrease the base operating frequency for other
23 components interconnected in the computer systemization. The clock and various
24 components in a computer systemization drive signals embodying information

1 throughout the system. Such transmission and reception of instructions embodying
2 information throughout a computer systemization may be commonly referred to as
3 communications. These communicative instructions may further be transmitted,
4 received, and the cause of return and/or reply communications beyond the instant
5 computer systemization to: communications networks, input devices, other computer
6 systemizations, peripheral devices, and/or the like. It should be understood that in
7 alternative embodiments, any of the above components may be connected directly to
8 one another, connected to the CPU, and/or organized in numerous variations employed
9 as exemplified by various computer systems.

10 **[00457]** The CPU comprises at least one high-speed data processor adequate to
11 execute program components for executing user and/or system-generated requests.
12 Often, the processors themselves will incorporate various specialized processing units,
13 such as, but not limited to: integrated system (bus) controllers, memory management
14 control units, floating point units, and even specialized processing sub-units like
15 graphics processing units, digital signal processing units, and/or the like. Additionally,
16 processors may include internal fast access addressable memory, and be capable of
17 mapping and addressing memory 5129 beyond the processor itself; internal memory
18 may include, but is not limited to: fast registers, various levels of cache memory (e.g.,
19 level 1, 2, 3, etc.), RAM, etc. The processor may access this memory through the use of a
20 memory address space that is accessible via instruction address, which the processor
21 can construct and decode allowing it to access a circuit path to a specific memory
22 address space having a memory state. The CPU may be a microprocessor such as:
23 AMD's Athlon, Duron and/or Opteron; ARM's application, embedded and secure
24 processors; IBM and/or Motorola's DragonBall and PowerPC; IBM's and Sony's Cell

1 processor; Intel's Celeron, Core (2) Duo, Itanium, Pentium, Xeon, and/or XScale;
2 and/or the like processor(s). The CPU interacts with memory through instruction
3 passing through conductive and/or transportive conduits (e.g., (printed) electronic
4 and/or optic circuits) to execute stored instructions (i.e., program code) according to
5 conventional data processing techniques. Such instruction passing facilitates
6 communication within the MCB-Platform controller and beyond through various
7 interfaces. Should processing requirements dictate a greater amount speed and/or
8 capacity, distributed processors (e.g., Distributed MCB-Platform), mainframe, multi-
9 core, parallel, and/or super-computer architectures may similarly be
10 employed. Alternatively, should deployment requirements dictate greater portability,
11 smaller Personal Digital Assistants (PDAs) may be employed.

12 **[00458]** Depending on the particular implementation, features of the MCB-
13 Platform may be achieved by implementing a microcontroller such as CAST's R8051XC2
14 microcontroller; Intel's MCS 51 (i.e., 8051 microcontroller); and/or the like. Also, to
15 implement certain features of the MCB-Platform, some feature implementations may
16 rely on embedded components, such as: Application-Specific Integrated Circuit
17 ("ASIC"), Digital Signal Processing ("DSP"), Field Programmable Gate Array ("FPGA"),
18 and/or the like embedded technology. For example, any of the MCB-Platform
19 component collection (distributed or otherwise) and/or features may be implemented
20 via the microprocessor and/or via embedded components; e.g., via ASIC, coprocessor,
21 DSP, FPGA, and/or the like. Alternately, some implementations of the MCB-Platform
22 may be implemented with embedded components that are configured and used to
23 achieve a variety of features or signal processing.

1 **[00459]** Depending on the particular implementation, the embedded components
2 may include software solutions, hardware solutions, and/or some combination of both
3 hardware/software solutions. For example, MCB-Platform features discussed herein
4 may be achieved through implementing FPGAs, which are a semiconductor devices
5 containing programmable logic components called "logic blocks", and programmable
6 interconnects, such as the high performance FPGA Virtex series and/or the low cost
7 Spartan series manufactured by Xilinx. Logic blocks and interconnects can be
8 programmed by the customer or designer, after the FPGA is manufactured, to
9 implement any of the MCB-Platform features. A hierarchy of programmable
10 interconnects allow logic blocks to be interconnected as needed by the MCB-Platform
11 system designer/administrator, somewhat like a one-chip programmable breadboard.
12 An FPGA's logic blocks can be programmed to perform the operation of basic logic gates
13 such as AND, and XOR, or more complex combinational operators such as decoders or
14 mathematical operations. In most FPGAs, the logic blocks also include memory
15 elements, which may be circuit flip-flops or more complete blocks of memory. In some
16 circumstances, the MCB-Platform may be developed on regular FPGAs and then
17 migrated into a fixed version that more resembles ASIC implementations. Alternate or
18 coordinating implementations may migrate MCB-Platform controller features to a final
19 ASIC instead of or in addition to FPGAs. Depending on the implementation all of the
20 aforementioned embedded components and microprocessors may be considered the
21 "CPU" and/or "processor" for the MCB-Platform.

22

Power Source

23 **[00460]** The power source 5186 may be of any standard form for powering small

1 electronic circuit board devices such as the following power cells: alkaline, lithium
2 hydride, lithium ion, lithium polymer, nickel cadmium, solar cells, and/or the like.
3 Other types of AC or DC power sources may be used as well. In the case of solar cells, in
4 one embodiment, the case provides an aperture through which the solar cell may
5 capture photonic energy. The power cell 5186 is connected to at least one of the
6 interconnected subsequent components of the MCB-Platform thereby providing an
7 electric current to all subsequent components. In one example, the power source 5186 is
8 connected to the system bus component 5104. In an alternative embodiment, an outside
9 power source 5186 is provided through a connection across the I/O 5108 interface. For
10 example, a USB and/or IEEE 1394 connection carries both data and power across the
11 connection and is therefore a suitable source of power.

Interface Adapters

13 **[00461]** Interface bus(es) 5107 may accept, connect, and/or communicate to a
14 number of interface adapters, conventionally although not necessarily in the form of
15 adapter cards, such as but not limited to: input output interfaces (I/O) 5108, storage
16 interfaces 5109, network interfaces 5110, and/or the like. Optionally, cryptographic
17 processor interfaces 5127 similarly may be connected to the interface bus. The interface
18 bus provides for the communications of interface adapters with one another as well as
19 with other components of the computer systemization. Interface adapters are adapted
20 for a compatible interface bus. Interface adapters conventionally connect to the
21 interface bus via a slot architecture. Conventional slot architectures may be employed,
22 such as, but not limited to: Accelerated Graphics Port (AGP), Card Bus, (Extended)
23 Industry Standard Architecture ((E)ISA), Micro Channel Architecture (MCA), NuBus,

1 Peripheral Component Interconnect (Extended) (PCI(X)), PCI Express, Personal
2 Computer Memory Card International Association (PCMCIA), and/or the like.

3 **[00462]** Storage interfaces 5109 may accept, communicate, and/or connect to a
4 number of storage devices such as, but not limited to: storage devices 5114, removable
5 disc devices, and/or the like. Storage interfaces may employ connection protocols such
6 as, but not limited to: (Ultra) (Serial) Advanced Technology Attachment (Packet
7 Interface) ((Ultra) (Serial) ATA(PI)), (Enhanced) Integrated Drive Electronics ((E)IDE),
8 Institute of Electrical and Electronics Engineers (IEEE) 1394, fiber channel, Small
9 Computer Systems Interface (SCSI), Universal Serial Bus (USB), and/or the like.

10 **[00463]** Network interfaces 5110 may accept, communicate, and/or connect to a
11 communications network 5113. Through a communications network 5113, the MCB-
12 Platform controller is accessible through remote clients 5133b (e.g., computers with web
13 browsers) by users 5133a. Network interfaces may employ connection protocols such as,
14 but not limited to: direct connect, Ethernet (thick, thin, twisted pair 10/100/1000 Base
15 T, and/or the like), Token Ring, wireless connection such as IEEE 802.11a-x, and/or the
16 like. Should processing requirements dictate a greater amount speed and/or capacity,
17 distributed network controllers (e.g., Distributed MCB-Platform), architectures may
18 similarly be employed to pool, load balance, and/or otherwise increase the
19 communicative bandwidth required by the MCB-Platform controller. A communications
20 network may be any one and/or the combination of the following: a direct
21 interconnection; the Internet; a Local Area Network (LAN); a Metropolitan Area
22 Network (MAN); an Operating Missions as Nodes on the Internet (OMNI); a secured
23 custom connection; a Wide Area Network (WAN); a wireless network (e.g., employing

1 protocols such as, but not limited to a Wireless Application Protocol (WAP), I-mode,
2 and/or the like); and/or the like. A network interface may be regarded as a specialized
3 form of an input output interface. Further, multiple network interfaces 5110 may be
4 used to engage with various communications network types 5113. For example, multiple
5 network interfaces may be employed to allow for the communication over broadcast,
6 multicast, and/or unicast networks.

7 **[00464]** Input Output interfaces (I/O) 5108 may accept, communicate, and/or
8 connect to user input devices 5111, peripheral devices 5112, cryptographic processor
9 devices 5128, and/or the like. I/O may employ connection protocols such as, but not
10 limited to: audio: analog, digital, monaural, RCA, stereo, and/or the like; data: Apple
11 Desktop Bus (ADB), IEEE 1394a-b, serial, universal serial bus (USB); infrared; joystick;
12 keyboard; midi; optical; PC AT; PS/2; parallel; radio; video interface: Apple Desktop
13 Connector (ADC), BNC, coaxial, component, composite, digital, Digital Visual Interface
14 (DVI), high-definition multimedia interface (HDMI), RCA, RF antennae, S-Video, VGA,
15 and/or the like; wireless transceivers: 802.11a/b/g/n/x; Bluetooth; cellular (e.g., code
16 division multiple access (CDMA), high speed packet access (HSPA(+)), high-speed
17 downlink packet access (HSDPA), global system for mobile communications (GSM),
18 long term evolution (LTE), WiMax, etc.); and/or the like. One typical output device may
19 include a video display, which typically comprises a Cathode Ray Tube (CRT) or Liquid
20 Crystal Display (LCD) based monitor with an interface (e.g., DVI circuitry and cable)
21 that accepts signals from a video interface, may be used. The video interface composites
22 information generated by a computer systemization and generates video signals based
23 on the composited information in a video memory frame. Another output device is a
24 television set, which accepts signals from a video interface. Typically, the video interface

1 provides the composited video information through a video connection interface that
2 accepts a video display interface (e.g., an RCA composite video connector accepting an
3 RCA composite video cable; a DVI connector accepting a DVI display cable, etc.).

4 **[00465]** User input devices 5111 often are a type of peripheral device 512 (see
5 below) and may include: card readers, dongles, finger print readers, gloves, graphics
6 tablets, joysticks, keyboards, microphones, mouse (mice), remote controls, retina
7 readers, touch screens (e.g., capacitive, resistive, etc.), trackballs, trackpads, sensors
8 (e.g., accelerometers, ambient light, GPS, gyroscopes, proximity, etc.), styluses, and/or
9 the like.

10 **[00466]** Peripheral devices 5112 may be connected and/or communicate to I/O
11 and/or other facilities of the like such as network interfaces, storage interfaces, directly
12 to the interface bus, system bus, the CPU, and/or the like. Peripheral devices may be
13 external, internal and/or part of the MCB-Platform controller. Peripheral devices may
14 include: antenna, audio devices (e.g., line-in, line-out, microphone input, speakers, etc.),
15 cameras (e.g., still, video, webcam, etc.), dongles (e.g., for copy protection, ensuring
16 secure transactions with a digital signature, and/or the like), external processors (for
17 added capabilities; e.g., crypto devices 528), force-feedback devices (e.g., vibrating
18 motors), network interfaces, printers, scanners, storage devices, transceivers (e.g.,
19 cellular, GPS, etc.), video devices (e.g., goggles, monitors, etc.), video sources, visors,
20 and/or the like. Peripheral devices often include types of input devices (e.g., cameras).

21 **[00467]** It should be noted that although user input devices and peripheral devices
22 may be employed, the MCB-Platform controller may be embodied as an embedded,
23 dedicated, and/or monitor-less (i.e., headless) device, wherein access would be provided

1 over a network interface connection.

2 **[00468]** Cryptographic units such as, but not limited to, microcontrollers,
3 processors 5126, interfaces 5127, and/or devices 5128 may be attached, and/or
4 communicate with the MCB-Platform controller. A MC68HC16 microcontroller,
5 manufactured by Motorola Inc., may be used for and/or within cryptographic units. The
6 MC68HC16 microcontroller utilizes a 16-bit multiply-and-accumulate instruction in the
7 16 MHz configuration and requires less than one second to perform a 512-bit RSA
8 private key operation. Cryptographic units support the authentication of
9 communications from interacting agents, as well as allowing for anonymous
10 transactions. Cryptographic units may also be configured as part of the CPU. Equivalent
11 microcontrollers and/or processors may also be used. Other commercially available
12 specialized cryptographic processors include: Broadcom's CryptoNetX and other
13 Security Processors; nCipher's nShield; SafeNet's Luna PCI (e.g., 7100) series;
14 Semaphore Communications' 40 MHz Roadrunner 184; Sun's Cryptographic
15 Accelerators (e.g., Accelerator 6000 PCIe Board, Accelerator 500 Daughtercard); Via
16 Nano Processor (e.g., L2100, L2200, U2400) line, which is capable of performing 500+
17 MB/s of cryptographic instructions; VLSI Technology's 33 MHz 6868; and/or the like.

18

Memory

19 **[00469]** Generally, any mechanization and/or embodiment allowing a processor to
20 affect the storage and/or retrieval of information is regarded as memory 5129. However,
21 memory is a fungible technology and resource, thus, any number of memory
22 embodiments may be employed in lieu of or in concert with one another. It is to be
23 understood that the MCB-Platform controller and/or a computer systemization may

1 employ various forms of memory 5129. For example, a computer systemization may be
2 configured wherein the operation of on-chip CPU memory (e.g., registers), RAM, ROM,
3 and any other storage devices are provided by a paper punch tape or paper punch card
4 mechanism; however, such an embodiment would result in an extremely slow rate of
5 operation. In a typical configuration, memory 5129 will include ROM 5106, RAM 5105,
6 and a storage device 5114. A storage device 5114 may be any conventional computer
7 system storage. Storage devices may include a drum; a (fixed and/or removable)
8 magnetic disk drive; a magneto-optical drive; an optical drive (i.e., Blu-ray, CD
9 ROM/RAM/Recordable (R)/ReWritable (RW), DVD R/RW, HD DVD R/RW etc.); an
10 array of devices (e.g., Redundant Array of Independent Disks (RAID)); solid state
11 memory devices (USB memory, solid state drives (SSD), etc.); other processor-readable
12 storage mediums; and/or other devices of the like. Thus, a computer systemization
13 generally requires and makes use of memory.

14 **Component Collection**

15 **[00470]** The memory 5129 may contain a collection of program and/or database
16 components and/or data such as, but not limited to: operating system component(s)
17 5115 (operating system); information server component(s) 5116 (information server);
18 user interface component(s) 5117 (user interface); Web browser component(s) 5118
19 (Web browser); database(s) 5119; mail server component(s) 5121; mail client
20 component(s) 5122; cryptographic server component(s) 5120 (cryptographic server); the
21 MCB-Platform component(s) 5135; and/or the like (i.e., collectively a component
22 collection). These components may be stored and accessed from the storage devices
23 and/or from storage devices accessible through an interface bus. Although non-

1 conventional program components such as those in the component collection, typically,
2 are stored in a local storage device 5114, they may also be loaded and/or stored in
3 memory such as: peripheral devices, RAM, remote storage facilities through a
4 communications network, ROM, various forms of memory, and/or the like.

5 **Operating System**

6 **[00471]** The operating system component 5115 is an executable program
7 component facilitating the operation of the MCB-Platform controller. Typically, the
8 operating system facilitates access of I/O, network interfaces, peripheral devices,
9 storage devices, and/or the like. The operating system may be a highly fault tolerant,
10 scalable, and secure system such as: Apple Macintosh OS X (Server); AT&T Plan 9; Be
11 OS; Unix and Unix-like system distributions (such as AT&T's UNIX; Berkley Software
12 Distribution (BSD) variations such as FreeBSD, NetBSD, OpenBSD, and/or the like;
13 Linux distributions such as Red Hat, Ubuntu, and/or the like); and/or the like operating
14 systems. However, more limited and/or less secure operating systems also may be
15 employed such as Apple Macintosh OS, IBM OS/2, Microsoft DOS, Microsoft Windows
16 2000/2003/3.1/95/98/CE/Millennium/NT/Vista/XP (Server), Palm OS, and/or the like.
17 An operating system may communicate to and/or with other components in a
18 component collection, including itself, and/or the like. Most frequently, the operating
19 system communicates with other program components, user interfaces, and/or the like.
20 For example, the operating system may contain, communicate, generate, obtain, and/or
21 provide program component, system, user, and/or data communications, requests,
22 and/or responses. The operating system, once executed by the CPU, may enable the
23 interaction with communications networks, data, I/O, peripheral devices, program

1 components, memory, user input devices, and/or the like. The operating system may
2 provide communications protocols that allow the MCB-Platform controller to
3 communicate with other entities through a communications network 5113. Various
4 communication protocols may be used by the MCB-Platform controller as a subcarrier
5 transport mechanism for interaction, such as, but not limited to: multicast, TCP/IP,
6 UDP, unicast, and/or the like.

7 **Information Server**

8 **[00472]** An information server component 5116 is a stored program component
9 that is executed by a CPU. The information server may be a conventional Internet
10 information server such as, but not limited to Apache Software Foundation's Apache,
11 Microsoft's Internet Information Server, and/or the like. The information server may
12 allow for the execution of program components through facilities such as Active Server
13 Page (ASP), ActiveX, (ANSI) (Objective-) C (++), C# and/or .NET, Common Gateway
14 Interface (CGI) scripts, dynamic (D) hypertext markup language (HTML), FLASH, Java,
15 JavaScript, Practical Extraction Report Language (PERL), Hypertext Pre-Processor
16 (PHP), pipes, Python, wireless application protocol (WAP), WebObjects, and/or the like.
17 The information server may support secure communications protocols such as, but not
18 limited to, File Transfer Protocol (FTP); HyperText Transfer Protocol (HTTP); Secure
19 Hypertext Transfer Protocol (HTTPS), Secure Socket Layer (SSL), messaging protocols
20 (e.g., America Online (AOL) Instant Messenger (AIM), Application Exchange (APEX),
21 ICQ, Internet Relay Chat (IRC), Microsoft Network (MSN) Messenger Service, Presence
22 and Instant Messaging Protocol (PRIM), Internet Engineering Task Force's (IETF's)
23 Session Initiation Protocol (SIP), SIP for Instant Messaging and Presence Leveraging

1 Extensions (SIMPLE), open XML-based Extensible Messaging and Presence Protocol
2 (XMPP) (i.e., Jabber or Open Mobile Alliance's (OMA's) Instant Messaging and
3 Presence Service (IMPS)), Yahoo! Instant Messenger Service, and/or the like. The
4 information server provides results in the form of Web pages to Web browsers, and
5 allows for the manipulated generation of the Web pages through interaction with other
6 program components. After a Domain Name System (DNS) resolution portion of an
7 HTTP request is resolved to a particular information server, the information server
8 resolves requests for information at specified locations on the MCB-Platform controller
9 based on the remainder of the HTTP request. For example, a request such as
10 `http://123.124.125.126/myInformation.html` might have the IP portion of the request
11 "123.124.125.126" resolved by a DNS server to an information server at that IP address;
12 that information server might in turn further parse the http request for the
13 `"/myInformation.html"` portion of the request and resolve it to a location in memory
14 containing the information "myInformation.html." Additionally, other information
15 serving protocols may be employed across various ports, e.g., FTP communications
16 across port 21, and/or the like. An information server may communicate to and/or with
17 other components in a component collection, including itself, and/or facilities of the
18 like. Most frequently, the information server communicates with the MCB-Platform
19 database 5119, operating systems, other program components, user interfaces, Web
20 browsers, and/or the like.

21 **[00473]** Access to the MCB-Platform database may be achieved through a number
22 of database bridge mechanisms such as through scripting languages as enumerated
23 below (e.g., CGI) and through inter-application communication channels as enumerated
24 below (e.g., CORBA, WebObjects, etc.). Any data requests through a Web browser are

1 parsed through the bridge mechanism into appropriate grammars as required by the
2 MCB-Platform. In one embodiment, the information server would provide a Web form
3 accessible by a Web browser. Entries made into supplied fields in the Web form are
4 tagged as having been entered into the particular fields, and parsed as such. The entered
5 terms are then passed along with the field tags, which act to instruct the parser to
6 generate queries directed to appropriate tables and/or fields. In one embodiment, the
7 parser may generate queries in standard SQL by instantiating a search string with the
8 proper join/select commands based on the tagged text entries, wherein the resulting
9 command is provided over the bridge mechanism to the MCB-Platform as a query. Upon
10 generating query results from the query, the results are passed over the bridge
11 mechanism, and may be parsed for formatting and generation of a new results Web page
12 by the bridge mechanism. Such a new results Web page is then provided to the
13 information server, which may supply it to the requesting Web browser.

14 **[00474]** Also, an information server may contain, communicate, generate, obtain,
15 and/or provide program component, system, user, and/or data communications,
16 requests, and/or responses.

17 **User Interface**

18 **[00475]** Computer interfaces in some respects are similar to automobile operation
19 interfaces. Automobile operation interface elements such as steering wheels, gearshifts,
20 and speedometers facilitate the access, operation, and display of automobile resources,
21 and status. Computer interaction interface elements such as check boxes, cursors,
22 menus, scrollers, and windows (collectively and commonly referred to as widgets)
23 similarly facilitate the access, capabilities, operation, and display of data and computer

1 hardware and operating system resources, and status. Operation interfaces are
2 commonly called user interfaces. Graphical user interfaces (GUIs) such as the Apple
3 Macintosh Operating System's Aqua, IBM's OS/2, Microsoft's Windows
4 2000/2003/3.1/95/98/CE/Millennium/NT/XP/Vista/7 (i.e., Aero), Unix's X-Windows
5 (e.g., which may include additional Unix graphic interface libraries and layers such as K
6 Desktop Environment (KDE), mythTV and GNU Network Object Model Environment
7 (GNOME)), web interface libraries (e.g., ActiveX, AJAX, (D)HTML, FLASH, Java,
8 JavaScript, etc. interface libraries such as, but not limited to, Dojo, jQuery(UI),
9 MooTools, Prototype, script.aculo.us, SWFObject, Yahoo! User Interface, any of which
10 may be used and) provide a baseline and means of accessing and displaying information
11 graphically to users.

12 **[00476]** A user interface component 5117 is a stored program component that is
13 executed by a CPU. The user interface may be a conventional graphic user interface as
14 provided by, with, and/or atop operating systems and/or operating environments such
15 as already discussed. The user interface may allow for the display, execution,
16 interaction, manipulation, and/or operation of program components and/or system
17 facilities through textual and/or graphical facilities. The user interface provides a facility
18 through which users may affect, interact, and/or operate a computer system. A user
19 interface may communicate to and/or with other components in a component
20 collection, including itself, and/or facilities of the like. Most frequently, the user
21 interface communicates with operating systems, other program components, and/or the
22 like. The user interface may contain, communicate, generate, obtain, and/or provide
23 program component, system, user, and/or data communications, requests, and/or
24 responses.

Web Browser

1
2 **[00477]** A Web browser component 5118 is a stored program component that is
3 executed by a CPU. The Web browser may be a conventional hypertext viewing
4 application such as Microsoft Internet Explorer or Netscape Navigator. Secure Web
5 browsing may be supplied with 128bit (or greater) encryption by way of HTTPS, SSL,
6 and/or the like. Web browsers allowing for the execution of program components
7 through facilities such as ActiveX, AJAX, (D)HTML, FLASH, Java, JavaScript, web
8 browser plug-in APIs (e.g., FireFox, Safari Plug-in, and/or the like APIs), and/or the
9 like. Web browsers and like information access tools may be integrated into PDAs,
10 cellular telephones, and/or other mobile devices. A Web browser may communicate to
11 and/or with other components in a component collection, including itself, and/or
12 facilities of the like. Most frequently, the Web browser communicates with information
13 servers, operating systems, integrated program components (e.g., plug-ins), and/or the
14 like; e.g., it may contain, communicate, generate, obtain, and/or provide program
15 component, system, user, and/or data communications, requests, and/or responses.
16 Also, in place of a Web browser and information server, a combined application may be
17 developed to perform similar operations of both. The combined application would
18 similarly affect the obtaining and the provision of information to users, user agents,
19 and/or the like from the MCB-Platform enabled nodes. The combined application may
20 be nugatory on systems employing standard Web browsers.

Mail Server

21
22 **[00478]** A mail server component 5121 is a stored program component that is
23 executed by a CPU 5103. The mail server may be a conventional Internet mail server

1 such as, but not limited to sendmail, Microsoft Exchange, and/or the like. The mail
2 server may allow for the execution of program components through facilities such as
3 ASP, ActiveX, (ANSI) (Objective-) C (++), C# and/or .NET, CGI scripts, Java,
4 JavaScript, PERL, PHP, pipes, Python, WebObjects, and/or the like. The mail server
5 may support communications protocols such as, but not limited to: Internet message
6 access protocol (IMAP), Messaging Application Programming Interface
7 (MAPI)/Microsoft Exchange, post office protocol (POP3), simple mail transfer protocol
8 (SMTP), and/or the like. The mail server can route, forward, and process incoming and
9 outgoing mail messages that have been sent, relayed and/or otherwise traversing
10 through and/or to the MCB-Platform.

11 **[00479]** Access to the MCB-Platform mail may be achieved through a number of
12 APIs offered by the individual Web server components and/or the operating system.

13 **[00480]** Also, a mail server may contain, communicate, generate, obtain, and/or
14 provide program component, system, user, and/or data communications, requests,
15 information, and/or responses.

16

Mail Client

17 **[00481]** A mail client component 5122 is a stored program component that is
18 executed by a CPU 5103. The mail client may be a conventional mail viewing application
19 such as Apple Mail, Microsoft Entourage, Microsoft Outlook, Microsoft Outlook
20 Express, Mozilla, Thunderbird, and/or the like. Mail clients may support a number of
21 transfer protocols, such as: IMAP, Microsoft Exchange, POP3, SMTP, and/or the like. A
22 mail client may communicate to and/or with other components in a component
23 collection, including itself, and/or facilities of the like. Most frequently, the mail client

1 communicates with mail servers, operating systems, other mail clients, and/or the like;
2 e.g., it may contain, communicate, generate, obtain, and/or provide program
3 component, system, user, and/or data communications, requests, information, and/or
4 responses. Generally, the mail client provides a facility to compose and transmit
5 electronic mail messages.

6 **Cryptographic Server**

7 **[00482]** A cryptographic server component 5120 is a stored program component
8 that is executed by a CPU 5103, cryptographic processor 5126, cryptographic processor
9 interface 5127, cryptographic processor device 5128, and/or the like. Cryptographic
10 processor interfaces will allow for expedition of encryption and/or decryption requests
11 by the cryptographic component; however, the cryptographic component, alternatively,
12 may run on a conventional CPU. The cryptographic component allows for the
13 encryption and/or decryption of provided data. The cryptographic component allows for
14 both symmetric and asymmetric (e.g., Pretty Good Protection (PGP)) encryption and/or
15 decryption. The cryptographic component may employ cryptographic techniques such
16 as, but not limited to: digital certificates (e.g., X.509 authentication framework), digital
17 signatures, dual signatures, enveloping, password access protection, public key
18 management, and/or the like. The cryptographic component will facilitate numerous
19 (encryption and/or decryption) security protocols such as, but not limited to: checksum,
20 Data Encryption Standard (DES), Elliptical Curve Encryption (ECC), International Data
21 Encryption Algorithm (IDEA), Message Digest 5 (MD5, which is a one way hash
22 operation), passwords, Rivest Cipher (RC5), Rijndael, RSA (which is an Internet
23 encryption and authentication system that uses an algorithm developed in 1977 by Ron

1 Rivest, Adi Shamir, and Leonard Adleman), Secure Hash Algorithm (SHA), Secure
2 Socket Layer (SSL), Secure Hypertext Transfer Protocol (HTTPS), and/or the like.
3 Employing such encryption security protocols, the MCB-Platform may encrypt all
4 incoming and/or outgoing communications and may serve as node within a virtual
5 private network (VPN) with a wider communications network. The cryptographic
6 component facilitates the process of “security authorization” whereby access to a
7 resource is inhibited by a security protocol wherein the cryptographic component effects
8 authorized access to the secured resource. In addition, the cryptographic component
9 may provide unique identifiers of content, e.g., employing and MD5 hash to obtain a
10 unique signature for an digital audio file. A cryptographic component may communicate
11 to and/or with other components in a component collection, including itself, and/or
12 facilities of the like. The cryptographic component supports encryption schemes
13 allowing for the secure transmission of information across a communications network
14 to enable the MCB-Platform component to engage in secure transactions if so desired.
15 The cryptographic component facilitates the secure accessing of resources on the MCB-
16 Platform and facilitates the access of secured resources on remote systems; i.e., it may
17 act as a client and/or server of secured resources. Most frequently, the cryptographic
18 component communicates with information servers, operating systems, other program
19 components, and/or the like. The cryptographic component may contain, communicate,
20 generate, obtain, and/or provide program component, system, user, and/or data
21 communications, requests, and/or responses.

22

The MCB-Platform Database

23 **[00483]** The MCB-Platform database component 5119 may be embodied in a

1 database and its stored data. The database is a stored program component, which is
2 executed by the CPU; the stored program component portion configuring the CPU to
3 process the stored data. The database may be a conventional, fault tolerant, relational,
4 scalable, secure database such as Oracle or Sybase. Relational databases are an
5 extension of a flat file. Relational databases consist of a series of related tables. The
6 tables are interconnected via a key field. Use of the key field allows the combination of
7 the tables by indexing against the key field; i.e., the key fields act as dimensional pivot
8 points for combining information from various tables. Relationships generally identify
9 links maintained between tables by matching primary keys. Primary keys represent
10 fields that uniquely identify the rows of a table in a relational database. More precisely,
11 they uniquely identify rows of a table on the “one” side of a one-to-many relationship.

12 **[00484]** Alternatively, the MCB-Platform database may be implemented using
13 various standard data-structures, such as an array, hash, (linked) list, struct, structured
14 text file (e.g., XML), table, and/or the like. Such data-structures may be stored in
15 memory and/or in (structured) files. In another alternative, an object-oriented database
16 may be used, such as Frontier, ObjectStore, Poet, Zope, and/or the like. Object
17 databases can include a number of object collections that are grouped and/or linked
18 together by common attributes; they may be related to other object collections by some
19 common attributes. Object-oriented databases perform similarly to relational databases
20 with the exception that objects are not just pieces of data but may have other types of
21 capabilities encapsulated within a given object. If the MCB-Platform database is
22 implemented as a data-structure, the use of the MCB-Platform database 5119 may be
23 integrated into another component such as the MCB-Platform component 5135. Also,
24 the database may be implemented as a mix of data structures, objects, and relational

1 structures. Databases may be consolidated and/or distributed in countless variations
2 through standard data processing techniques. Portions of databases, e.g., tables, may be
3 exported and/or imported and thus decentralized and/or integrated.

4 **[00485]** In one embodiment, the database component 5119 includes several tables
5 5119a-u. A Users table 5119a may include fields such as, but not limited to: user_id, ssn,
6 dob, first_name, last_name, address, age, state, address_firstline, address_secondline,
7 zipcode, application_id, application_type, exchange_preferences_list, exchange_
8 preferences_values, devices_list, user_accounts_list, user_passwords_list, security_
9 level, and/or the like. The Users table may support and/or track multiple entity
10 accounts on a MCB-Platform. A confidence score table 5119b includes fields such as, but
11 not limited to: merchant_id, data_source, source_ip, merchant_info, confidence_score,
12 score_timestamp, score_status, activity_status, and/or the like. An Exchanges table
13 5119c may include fields such as, but not limited to: user_id, currency_type,
14 currency_id, currency_name, currency_float_flag, currency_exchange_restrictions,
15 unit_currency_value, exchange_rate, exchange_refresh_rate, baseline_rate,
16 market_symbol, market_name, exchange_rate_startdate, exchange_rate_enddate,
17 base_currency, and/or the like. A Merchants/provider table 5119d may include fields
18 such as, but not limited to: provider_id, program_name, address_firstline,
19 address_secondline, zipcode, application_id, application_type,
20 exchange_preferences_list, exchange_preferences_values, devices_list, registered_
21 users_list, currency_type and/or the like. A Banks/issuer table 5119e includes fields
22 such as, but not limited to: bank_id, bank_bame, aba_number, routing_number, micr,
23 branch_name, branch_code, address_first_line, address_secondline, zipcode,
24 issuer_address, ip_address, mac_address, auth_key, port_num, security_settings_list,

1 and/or the like. A rules and restrictions table 5119f includes fields such as, but not
2 limited to: rules_ID, rulesrestriction_list, and/or the like. An accounts table 5119g
3 includes fields such as, but not limited to: user_ID, program_ID, enrolled_status,
4 points_balance, last_update_date, account_number, account_security_code,
5 account_name, issuer_acquirer_flag, issuer_name, acquirer_name, account_address,
6 routing_number, access_API_call, linked_wallets_list, and/or the like. An exchange
7 rates table 5119h includes fields such as, but not limited to: program_ID,
8 base_currency, exchangerate, date, and/or the like. A payment devices/cards table 5119i
9 includes fields such as, but not limited to: user_ID, payment_device_type,
10 payment_device_identifier, payment_device_securitycode, billing_address,
11 bank_account_number, and/or the like. An analytics table 5119j includes fields such as,
12 but not limited to: program_ID, user_ID, transaction_volume, time_period, and/or the
13 like. A programs table 5119k includes fields such as, but not limited to program_ID,
14 rules_ID, notallowedprogram_IDs, preferred_program_IDs, normal_exchnage_rate,
15 preferred_exchange_rate, and/or the like. A Web Data table 5119l includes fields such
16 as, but not limited to: web_data_feed_ID, web_data_source, web_URL, web_image,
17 web_content, web_content_URL, web_viewing_status, and/or the like. An Acquirers
18 table 5119m may include fields such as, but not limited to: merchant_ID,
19 account_firstname, account_lastname, account_type, account_num, account_
20 balance_list, billingaddress_line1, billingaddress_line2, billing_zipcode, billing_state,
21 shipping_preferences, shippingaddress_line1, shippingaddress_line2, shipping_
22 zipcode, shipping_state, and/or the like. A Pay Gateways table 5119n may include fields
23 such as, but not limited to: gateway_ID, gateway_IP, gateway_MAC,
24 gateway_secure_key, gateway_access_list, gateway_API_call_list,

1 gateway_services_list, and/or the like. A Transactions table 5119o may include fields
2 such as, but not limited to: order_id, user_id, timestamp, transaction_cost,
3 purchase_details_list, num_products, products_list, product_type,
4 product_params_list, product_title, product_summary, quantity, user_id, client_id,
5 client_ip, client_type, client_model, operating_system, os_version, app_installed_flag,
6 user_id, account_firstname, account_lastname, account_type, account_num,
7 account_priority_account_ratio, billingaddress_line1, billingaddress_line2,
8 billing_zipcode, billing_state, shipping_preferences, shippingaddress_line1,
9 shippingaddress_line2, shipping_zipcode, shipping_state, merchant_id,
10 merchant_name, merchant_auth_key, and/or the like. A Batches table 5119p may
11 include fields such as, but not limited to: batch_id, transaction_id_list, timestamp_list,
12 cleared_flag_list, clearance_trigger_settings, and/or the like. A Ledgers table 5119q
13 may include fields such as, but not limited to: request_id, timestamp, deposit_amount,
14 batch_id, transaction_id, clear_flag, deposit_account, transaction_summary, payor_
15 name, payor_account, and/or the like. A Products table 5119r may include fields such
16 as, but not limited to: product_ID, product_title, product_attributes_list,
17 product_price, tax_info_list, related_products_list, offers_list, discounts_list,
18 rewards_list, merchants_list, merchant_availability_list, and/or the like. An Offers
19 table 5119s may include fields such as, but not limited to: offer_ID, offer_title,
20 offer_attributes_list, offer_price, offer_expiry, related_products_list, discounts_list,
21 rewards_list, merchants_list, merchant_availability_list, and/or the like. An Apps table
22 5119t may include fields such as, but not limited to: app_ID, app_name, app_type,
23 app_dependencies, and/or the like. A value card table 5119u may include fields such as,
24 but not limited to: value_card_ID, value_amount, tracking_equivalent_amount,

1 source_user_ID, destination_user_ID, current_user_ID, and/or the like.

2 **[00486]** In one embodiment, the MCB-Platform database may interact with other
3 database systems. For example, employing a distributed database system, queries and
4 data access by search MCB-Platform component may treat the combination of the MCB-
5 Platform database, an integrated data security layer database as a single database entity.

6 **[00487]** In one embodiment, user programs may contain various user interface
7 primitives, which may serve to update the MCB-Platform. Also, various accounts may
8 require custom database tables depending upon the environments and the types of
9 clients the MCB-Platform may need to serve. It should be noted that any unique fields
10 may be designated as a key field throughout. In an alternative embodiment, these tables
11 have been decentralized into their own databases and their respective database
12 controllers (i.e., individual database controllers for each of the above tables). Employing
13 standard data processing techniques, one may further distribute the databases over
14 several computer systemizations and/or storage devices. Similarly, configurations of the
15 decentralized database controllers may be varied by consolidating and/or distributing
16 the various database components 5119a-u. The MCB-Platform may be configured to
17 keep track of various settings, inputs, and parameters via database controllers.

18 **[00488]** The MCB-Platform database may communicate to and/or with other
19 components in a component collection, including itself, and/or facilities of the like.
20 Most frequently, the MCB-Platform database communicates with the MCB-Platform
21 component, other program components, and/or the like. The database may contain,
22 retain, and provide information regarding other nodes and data.

The MCB-Platforms

1
2 **[00489]** The MCB-Platform component 5135 is a stored program component that is
3 executed by a CPU. In one embodiment, the MCB-Platform component incorporates any
4 and/or all combinations of the aspects of the MCB-Platform that was discussed in the
5 previous figures. As such, the MCB-Platform affects accessing, obtaining and the
6 provision of information, services, transactions, and/or the like across various
7 communications networks.

8 **[00490]** The MCB-Platform transforms various activity requests (e.g., transaction
9 request, merchant information update request, offer issuance request, etc.) via MCB-
10 Platform components, such as but not limited to web claw component 5149, offer
11 bridging component 5150, scoring component 5151, DB update component 5152, and
12 centralized personal information platform component 5153 into transaction records and
13 merchant database updates outputs. In other embodiments, the MCB-Platform
14 transforms value exchange requests via CLGC-MCB-Platform 5144, GC-MCB-Platform
15 5145, SD-MCB-Platform 5146, EVD 5147, CE-MCB-Platform 5148 into currency
16 exchanges outputs. In one embodiment, the MCB-Platform component 5135 takes
17 inputs (e.g., checkout request 4511; product data 4515; wallet access input 4711;
18 transaction authorization input 4714; payment gateway address 4718; payment network
19 address 4722; issuer server address(es) 4725; funds authorization request(s) 4726; user(s)
20 account(s) data 4728; batch data 4912; payment network address 4916; issuer server
21 address(es) 4924; individual payment request 4925; payment ledger, merchant account data
22 4931; and/or the like) etc., and transforms the inputs via various components (e.g., UPC
23 5141; PTA 5142; PTC 5143; and/or the like), into outputs (e.g., checkout request
24 message 4513; checkout data 4517; card authorization request 4716, 4723; funds

1 authorization response(s) 4730; transaction authorization response 4732; batch append data
2 4734; purchase receipt 4735; batch clearance request 4914; batch payment request 4918;
3 transaction data 4920; individual payment confirmation 4928, 4929; updated payment
4 ledger, merchant account data 4933; and/or the like).

5 **[00491]** The MCB-Platform component enabling access of information between
6 nodes may be developed by employing standard development tools and languages such
7 as, but not limited to: Apache components, Assembly, ActiveX, binary executables,
8 (ANSI) (Objective-) C (++), C# and/or .NET, database adapters, CGI scripts, Java,
9 JavaScript, mapping tools, procedural and object oriented development tools, PERL,
10 PHP, Python, shell scripts, SQL commands, web application server extensions, web
11 development environments and libraries (e.g., Microsoft's ActiveX; Adobe AIR, FLEX &
12 FLASH; AJAX; (D)HTML; Dojo, Java; JavaScript; jQuery(UI); MooTools; Prototype;
13 script.aculo.us; Simple Object Access Protocol (SOAP); SWFObject; Yahoo! User
14 Interface; and/or the like), WebObjects, and/or the like. In one embodiment, the MCB-
15 Platform server employs a cryptographic server to encrypt and decrypt communications.
16 The MCB-Platform component may communicate to and/or with other components in a
17 component collection, including itself, and/or facilities of the like. Most frequently, the
18 MCB-Platform component communicates with the MCB-Platform database, operating
19 systems, other program components, and/or the like. The MCB-Platform may contain,
20 communicate, generate, obtain, and/or provide program component, system, user,
21 and/or data communications, requests, and/or responses.

22

Distributed MCB-Platforms

23 **[00492]** The structure and/or operation of any of the MCB-Platform node

1 controller components may be combined, consolidated, and/or distributed in any
2 number of ways to facilitate development and/or deployment. Similarly, the component
3 collection may be combined in any number of ways to facilitate deployment and/or
4 development. To accomplish this, one may integrate the components into a common
5 code base or in a facility that can dynamically load the components on demand in an
6 integrated fashion.

7 **[00493]** The component collection may be consolidated and/or distributed in
8 countless variations through standard data processing and/or development techniques.
9 Multiple instances of any one of the program components in the program component
10 collection may be instantiated on a single node, and/or across numerous nodes to
11 improve performance through load-balancing and/or data-processing techniques.
12 Furthermore, single instances may also be distributed across multiple controllers
13 and/or storage devices; e.g., databases. All program component instances and
14 controllers working in concert may do so through standard data processing
15 communication techniques.

16 **[00494]** The configuration of the MCB-Platform controller will depend on the
17 context of system deployment. Factors such as, but not limited to, the budget, capacity,
18 location, and/or use of the underlying hardware resources may affect deployment
19 requirements and configuration. Regardless of if the configuration results in more
20 consolidated and/or integrated program components, results in a more distributed
21 series of program components, and/or results in some combination between a
22 consolidated and distributed configuration, data may be communicated, obtained,
23 and/or provided. Instances of components consolidated into a common code base from

1 the program component collection may communicate, obtain, and/or provide data. This
2 may be accomplished through intra-application data processing communication
3 techniques such as, but not limited to: data referencing (e.g., pointers), internal
4 messaging, object instance variable communication, shared memory space, variable
5 passing, and/or the like.

6 **[00495]** If component collection components are discrete, separate, and/or
7 external to one another, then communicating, obtaining, and/or providing data with
8 and/or to other component components may be accomplished through inter-application
9 data processing communication techniques such as, but not limited to: Application
10 Program Interfaces (API) information passage; (distributed) Component Object Model
11 ((D)COM), (Distributed) Object Linking and Embedding ((D)OLE), and/or the like),
12 Common Object Request Broker Architecture (CORBA), Jini local and remote
13 application program interfaces, JavaScript Object Notation (JSON), Remote Method
14 Invocation (RMI), SOAP, process pipes, shared files, and/or the like. Messages sent
15 between discrete component components for inter-application communication or within
16 memory spaces of a singular component for intra-application communication may be
17 facilitated through the creation and parsing of a grammar. A grammar may be
18 developed by using development tools such as lex, yacc, XML, and/or the like, which
19 allow for grammar generation and parsing capabilities, which in turn may form the basis
20 of communication messages within and between components.

²¹ **[00496]** For example, a grammar may be arranged to recognize the tokens of an
²² HTTP post command, e.g.:

```
23 w3c -post http://... Value1
24
```

1 **[00497]** where Value1 is discerned as being a parameter because “http://” is part of
2 the grammar syntax, and what follows is considered part of the post value. Similarly,
3 with such a grammar, a variable “Value1” may be inserted into an “http://” post
4 command and then sent. The grammar syntax itself may be presented as structured data
5 that is interpreted and/or otherwise used to generate the parsing mechanism (e.g., a
6 syntax description text file as processed by lex, yacc, etc.). Also, once the parsing
7 mechanism is generated and/or instantiated, it itself may process and/or parse
8 structured data such as, but not limited to: character (e.g., tab) delineated text, HTML,
9 structured text streams, XML, and/or the like structured data. In another embodiment,
10 inter-application data processing protocols themselves may have integrated and/or
11 readily available parsers (e.g., JSON, SOAP, and/or like parsers) that may be employed
12 to parse (e.g., communications) data. Further, the parsing grammar may be used
13 beyond message parsing, but may also be used to parse: databases, data collections, data
14 stores, structured data, and/or the like. Again, the desired configuration will depend
15 upon the context, environment, and requirements of system deployment.

16 **[00498]** For example, in some implementations, the MCB-Platform controller may
17 be executing a PHP script implementing a Secure Sockets Layer (“SSL”) socket server
18 via the information sherver, which listens to incoming communications on a server port
19 to which a client may send data, e.g., data encoded in JSON format. Upon identifying
20 an incoming communication, the PHP script may read the incoming message from the
21 client device, parse the received JSON-encoded text data to extract information from the
22 JSON-encoded text data into PHP script variables, and store the data (e.g., client
23 identifying information, etc.) and/or extracted information in a relational database
24 accessible using the Structured Query Language (“SQL”). An exemplary listing, written

1 substantially in the form of PHP/SQL commands, to accept JSON-encoded input data
 2 from a client device via a SSL connection, parse the data to extract variables, and store
 3 the data to a database, is provided below:

```

4  <?PHP
5  header('Content-Type: text/plain');
6
7  // set ip address and port to listen to for incoming data
8  $address = '192.168.0.100';
9  $port = 255;
10
11 // create a server-side SSL socket, listen for/accept incoming communication
12 $sock = socket_create(AF_INET, SOCK_STREAM, 0);
13 socket_bind($sock, $address, $port) or die('Could not bind to address');
14 socket_listen($sock);
15 $client = socket_accept($sock);
16
17 // read input data from client device in 1024 byte blocks until end of message
18 do {
19     $input = "";
20     $input = socket_read($client, 1024);
21     $data .= $input;
22 } while($input != "");
23
24 // parse data to extract variables
25 $obj = json_decode($data, true);
26
27 // store input data in a database
28 mysql_connect("201.408.185.132", $DBserver, $password); // access database server
29 mysql_select("CLIENT_DB.SQL"); // select database to append
30 mysql_query("INSERT INTO UserTable (transmission)
31 VALUES ($data)"); // add data to UserTable table in a CLIENT database
32 mysql_close("CLIENT_DB.SQL"); // close connection to database
33 ?>
34

```

35 **[00499]** Also, the following resources may be used to provide example
 36 embodiments regarding SOAP parser implementation:

```

37 http://www.xav.com/perl/site/lib/SOAP/Parser.html
38 http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?topic=/com.ibm
39 .IBMDI.doc/referenceguide295.htm
40

```

41 **[00500]** and other parser implementations:

```

42 http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?topic=/com.ibm
43 .IBMDI.doc/referenceguide259.htm
44

```

45 **[00501]** all of which are hereby expressly incorporated by reference.

46 **[00502]** Additional embodiments of MCB-Platform universal value equivalents
 47 may further include:

- 1 1. A universal value exchange processor-implemented method embodiment to
2 transform value equivalent exchange instructions into cross-ecosystem currency
3 exchanges, comprising:
- 4 obtaining a cross-ecosystem currency exchange instruction;
5 determining one or more sources based on parsing the cross-ecosystem
6 currency exchange instruction;
7 identifying currency types associated with the sources;
8 determining exchange rates of the currency types relative to a standard
9 currency;
10 obtaining currency exchange restrictions and conditions associated with
11 the sources;
12 generating, based on the currency exchange restrictions and conditions,
13 currency exchange flow paths for currency transfer from the sources to destinations;
14 issuing currency transfer requests to the sources and one of the
15 destinations;
16 determining that the cross-ecosystem currency exchange has been
17 completed; and
18 providing a notification of completion of the cross-ecosystem currency
19 exchange.
- 20 2. The embodiment of claim 1, wherein the currency exchange flow paths are
21 generated based on the currency exchange restrictions and conditions associated with
22 the sources.
- 23 3. The embodiment of claim 1, further comprising automatically selecting a
24 currency exchange flow path from the generated currency exchange flow paths based on

1 exchange rates between the sources and the destinations.

2 4. The embodiment of claim 3, wherein said one of the destinations
3 corresponds to the selected currency exchange flow path.

4 5. The embodiment of claim 1, further comprising selecting a currency
5 exchange flow path from the generated currency exchange flow paths based on user
6 preference.

7 6. The embodiment of claim 1, further comprising selecting a currency
8 exchange flow path from the generated currency exchange flow paths based on affiliate
9 relationship between the sources and the destinations.

10 7. The embodiment of claim 1, wherein determining the exchange rates of the
11 currency types includes determining liquidity of the currency types.

12 8. The embodiment of claim 1, wherein the cross-ecosystem currency
13 exchange instruction further comprises source currency amounts in accounts of the one
14 or more sources.

15 9. The embodiment of claim 8, wherein the issued currency transfer requests
16 to the sources include a request to deallocate the source currency amounts from the
17 accounts of the one or more sources.

18 10. The embodiment of claim 8, wherein the issued currency transfer requests
19 include a request to allocate an equivalent amount to an account corresponding to said
20 one of the destinations.

21 11. The embodiment of claim 10, wherein the equivalent amount is
22 determined by the exchange rates.

23 12. A universal value exchange system, comprising means to:
24 obtain a cross-ecosystem currency exchange instruction;

1 determine one or more sources based on parsing the cross-ecosystem
2 currency exchange instruction;
3 identify currency types associated with the sources;
4 determine exchange rates of the currency types relative to a standard
5 currency;
6 obtain currency exchange restrictions and conditions associated with the
7 sources;
8 generate, based on the currency exchange restrictions and conditions,
9 currency exchange flow paths for currency transfer from the sources to destinations;
10 issue currency transfer requests to the sources and one of the destinations;
11 determine that the cross-ecosystem currency exchange has been
12 completed; and
13 provide a notification of completion of the cross-ecosystem currency
14 exchange.

15 13. A processor-readable medium storing processor-issuable instructions to:
16 obtain a cross-ecosystem currency exchange instruction;
17 determine one or more sources based on parsing the cross-ecosystem
18 currency exchange instruction;
19 identify currency types associated with the sources;
20 determine exchange rates of the currency types relative to a standard
21 currency;
22 obtain currency exchange restrictions and conditions associated with the
23 sources;
24 generate, based on the currency exchange restrictions and conditions,

1 currency exchange flow paths for currency transfer from the sources to destinations;
2 issue currency transfer requests to the sources and one of the destinations;
3 determine that the cross-ecosystem currency exchange has been
4 completed; and
5 provide a notification of completion of the cross-ecosystem currency
6 exchange.

7 14. A universal value exchange apparatus, comprising:
8 a memory;
9 a processor disposed in communication with said memory, and configured to
10 issue a plurality of processing instructions stored in the memory, wherein the processor
11 issues instructions to:

12 obtain a cross-ecosystem currency exchange instruction;
13 determine one or more sources based on parsing the cross-ecosystem
14 currency exchange instruction;
15 identify currency types associated with the sources;
16 determine exchange rates of the currency types relative to a standard
17 currency;
18 obtain currency exchange restrictions and conditions associated with the
19 sources;
20 generate, based on the currency exchange restrictions and conditions,
21 currency exchange flow paths for currency transfer from the sources to destinations;
22 issue currency transfer requests to the sources and one of the destinations;
23 determine that the cross-ecosystem currency exchange has been
24 completed; and

1 provide a notification of completion of the cross-ecosystem currency
2 exchange.

3 15. A processor-implemented cross-ecosystem brokerage embodiment,
4 comprising:

5 receiving an offer to carry out an exchange of a first ecosystem currency at
6 a first exchange rate relative to a second ecosystem currency;

7 receiving user instructions to exchange the first ecosystem currency for a
8 third ecosystem currency at a second exchange rate;

9 obtaining authorization to execute a cross-ecosystem exchange;

10 executing the cross-ecosystem exchange, wherein said executing includes:

11 reducing the user's first ecosystem currency balance by a first
12 amount; and

13 incrementing the user's third ecosystem currency balance by a
14 second amount determined based on the second exchange rate; and

15 charging the offer provider a third amount in the second ecosystem
16 currency for carrying out the exchange of the first ecosystem currency.

17 16. The embodiment of claim 15, wherein offer provider is an issuer of the first
18 ecosystem currency.

19 17. The embodiment of claim 15, wherein the first, second and third
20 ecosystem currencies are selected from a group consisting of: (i) loyalty program
21 currency; (ii) virtual currency; and (iii) financial currency.

22 18. The embodiment of claim 15, wherein obtaining the authorization
23 includes:

24 providing the user the second amount of the third ecosystem currency

1 obtainable upon executing the cross-ecosystem exchange; and

2 soliciting from the user a response to execute the cross-ecosystem
3 exchange.

4 19. The embodiment of claim 15, wherein reducing the user's first ecosystem
5 currency balance includes:

6 notifying the offer provider the exchange of the first amount of the first
7 ecosystem currency by the user, wherein the offer provider reduces the user's first
8 ecosystem currency balance by the first amount.

9 20. The embodiment of claim 15, wherein the cross-ecosystem exchange is
10 executed on a third ecosystem having the third ecosystem currency.

11 21. The embodiment of claim 15, wherein the third amount charged to the
12 offer provider is determined based on the first amount of the first ecosystem currency
13 and the first exchange rate relative to the second ecosystem currency.

14 22. The embodiment of claim 15, wherein the received offer includes one or
15 more rules and restrictions.

16 23. The embodiment of claim 22, further comprising:

17 determining that the user instructions to exchange the first ecosystem
18 currency for the third ecosystem currency at the second exchange rate meet the one or
19 more rules and restrictions.

20 24. The embodiment of claim 23, further comprising:

21 requesting the user to modify the user instructions when the user
22 instructions fail to meet the one or more rules and restrictions.

23 25. The embodiment of claim 15, wherein the offer provider is associated with
24 a first ecosystem having the first ecosystem currency.

1 26. The embodiment of claim 25, wherein the first ecosystem has a bilateral
2 relationship with a third ecosystem having the third ecosystem currency, allowing cross-
3 ecosystem exchange between the first and third ecosystems.

4 27. The embodiment of claim 26, wherein the user is program member in
5 third ecosystem.

6 28. A cross-ecosystem brokerage system, comprising means to:

7 receive an offer to carry out an exchange of a first ecosystem currency at a
8 first exchange rate relative to a second ecosystem currency;

9 receive user instructions to exchange the first ecosystem currency for a
10 third ecosystem currency at a second exchange rate;

11 obtain authorization to execute a cross-ecosystem exchange;

12 execute the cross-ecosystem exchange, wherein said executing includes:

13 reduce the user's first ecosystem currency balance by a first
14 amount; and

15 increment the user's third ecosystem currency balance by a second
16 amount determined based on the second exchange rate; and

17 charge the offer provider a third amount in the second ecosystem currency
18 for carrying out the exchange of the first ecosystem currency.

19 29. A processor-readable medium storing processor-issuable instructions to:

20 receive an offer to carry out an exchange of a first ecosystem currency at a
21 first exchange rate relative to a second ecosystem currency;

22 receive user instructions to exchange the first ecosystem currency for a
23 third ecosystem currency at a second exchange rate;

24 obtain authorization to execute a cross-ecosystem exchange;

1 execute the cross-ecosystem exchange, wherein said executing includes:

2 reduce the user's first ecosystem currency balance by a first

3 amount; and

4 increment the user's third ecosystem currency balance by a second

5 amount determined based on the second exchange rate; and

6 charge the offer provider a third amount in the second ecosystem currency

7 for carrying out the exchange of the first ecosystem currency.

8 30. A cross-ecosystem brokerage apparatus, comprising:

9 a memory;

10 a processor disposed in communication with said memory, and configured to

11 issue a plurality of processing instructions stored in the memory, wherein the processor

12 issues instructions to:

13 receive an offer to carry out an exchange of a first ecosystem

14 currency at a first exchange rate relative to a second ecosystem currency;

15 receive user instructions to exchange the first ecosystem currency

16 for a third ecosystem currency at a second exchange rate;

17 obtain authorization to execute a cross-ecosystem exchange;

18 execute the cross-ecosystem exchange, wherein said executing

19 includes:

20 reduce the user's first ecosystem currency balance by a first

21 amount; and

22 increment the user's third ecosystem currency balance by a

23 second amount determined based on the second exchange rate; and

24 charge the offer provider a third amount in the second ecosystem

1 currency for carrying out the exchange of the first ecosystem currency.

2 31. A processor-implemented gift card exchange and transaction facilitation
3 embodiment, comprising:

4 receiving a request from a user to transfer funds from a first gift card to a
5 second gift card, each of said gift cards being associated with different issuers;

6 determining an amount of funds associated the first gift card;

7 determining an equivalent amount of funds transferable to the second gift
8 card;

9 querying a gift card database to determine a target gift card identifier
10 having at least the equivalent amount of funds, said target gift card being associated
11 with the issuer of the second gift card; deallocating the amount of funds from the first
12 gift card and the equivalent amount of funds from the target gift card; allocating the
13 equivalent amount of funds to the second gift card;

14 receiving a payment request corresponding to the user's purchase using
15 the second gift card; and forwarding the payment to the issuer of the target gift card,
16 wherein the issuer of the target gift card deducts the equivalent amount of funds from
17 the target gift card; and

18 deallocating the equivalent amount of funds from the second gift card.

19 32. The embodiment of claim 31, wherein the target gift card is associated with
20 a second user associated with a complimentary request to transfer funds from a gift card
21 issued by the second gift card issuer to a gift card issued by the first gift card issuer.

22 33. A processor-implemented gift card exchange and transaction facilitation
23 embodiment, comprising:

24 receiving a request from a user to transfer funds from a source gift card

1 issued by a first issuer to a destination gift card issued by a second issuer;
2 determining an amount of transfer funds associated the source gift card;
3 determining an equivalent amount of funds transferable to the destination
4 gift card; querying a gift card database to obtain a target gift card issued by the first
5 issuer and a target gift card issued by the second issuer;
6 facilitating transfer of the amount of transfer funds from the source gift
7 card to the target gift card issued by the first issuer; and facilitating transfer of the
8 equivalent amount of funds from the target gift card issued by the second issuer to the
9 destination gift card.

10 34. A processor-implemented low-latency value equivalent exchange
11 embodiment, comprising:

12 receiving a specified consumer value equivalent swap request to exchange a value
13 source for a value substitute as part of a purchase transaction;

14 determining a conversion exchange rate and amount of a denomination amount
15 specified in the swap request from the value source to a denomination amount specified
16 in the swap request for the value substitute;

17 identifying, within a low-latency purchase transaction time-frame, a target source
18 for the value substitute having sufficient value to supply the swap request based on the
19 determined conversion exchange rate and amount;

20 associating the value source with a holder of the target source by modifying a
21 value source record;

22 associating the target source with the holder of the value source by modifying the
23 value source record;

24 deallocating the determined conversion rate and amount specified in the swap

1 request for the value source holder and allocating it to the target source holder by
2 modifying the value source record to reduce network transactions and decrease network
3 and transaction latency; and

4 charging the determined conversion rate and amount specified in the swap
5 request to the target source to assist the payment of the purchase transaction for the
6 value source holder by using target source account information from a target source
7 record to decrease network and transaction latency.

8 **[00503]** Further embodiments of the MCB-Platform illustrating a centralized
9 information scoring may comprise:

10 **[00504]** A centralized personal information platform processor-implemented
11 method, comprising: aggregating data records including search results, purchase
12 transaction data, service usage data, service enrollment data, and social data; identifying
13 data field types within the data records and their associated data values; identifying an
14 entity from the data field types and their associated data values; generating, via a
15 processor, correlations of the entity to other entities identifiable from the data field
16 types and their associated data values; associating, via the processor, attributes to the
17 entity by drawing inferences related to the entity from the data field types and their
18 associated data values; generating an updated profile and social graph of the entity
19 using the generated correlations and associated attributes; and providing the updated
20 profile and social graph for an automated web form filling request.

21 **[00505]** In order to address various issues and advance the art, the entirety of this
22 application for MERCHANT-CONSUMER BRIDGING PLATFORM APPARATUSES,
23 METHODS AND SYSTEMS (including the Cover Page, Title, Headings, Field,

1 Background, Summary, Brief Description of the Drawings, Detailed Description, Claims,
2 Abstract, Figures, Appendices, and otherwise) shows, by way of illustration, various
3 embodiments in which the claimed innovations may be practiced. The advantages and
4 features of the application are of a representative sample of embodiments only, and are
5 not exhaustive and/or exclusive. They are presented only to assist in understanding and
6 teach the claimed principles. It should be understood that they are not representative of
7 all claimed innovations. As such, certain aspects of the disclosure have not been
8 discussed herein. That alternate embodiments may not have been presented for a
9 specific portion of the innovations or that further undescribed alternate embodiments
10 may be available for a portion is not to be considered a disclaimer of those alternate
11 embodiments. It will be appreciated that many of those undescribed embodiments
12 incorporate the same principles of the innovations and others are equivalent. Thus, it is
13 to be understood that other embodiments may be utilized and functional, logical,
14 operational, organizational, structural and/or topological modifications may be made
15 without departing from the scope and/or spirit of the disclosure. As such, all examples
16 and/or embodiments are deemed to be non-limiting throughout this disclosure. Also, no
17 inference should be drawn regarding those embodiments discussed herein relative to
18 those not discussed herein other than it is as such for purposes of reducing space and
19 repetition. For instance, it is to be understood that the logical and/or topological
20 structure of any combination of any program components (a component collection),
21 other components and/or any present feature sets as described in the figures and/or
22 throughout are not limited to a fixed operating order and/or arrangement, but rather,
23 any disclosed order is exemplary and all equivalents, regardless of order, are
24 contemplated by the disclosure. Furthermore, it is to be understood that such features

1 are not limited to serial execution, but rather, any number of threads, processes,
2 services, servers, and/or the like that may execute asynchronously, concurrently, in
3 parallel, simultaneously, synchronously, and/or the like are contemplated by the
4 disclosure. As such, some of these features may be mutually contradictory, in that they
5 cannot be simultaneously present in a single embodiment. Similarly, some features are
6 applicable to one aspect of the innovations, and inapplicable to others. In addition, the
7 disclosure includes other innovations not presently claimed. Applicant reserves all
8 rights in those presently unclaimed innovations including the right to claim such
9 innovations, file additional applications, continuations, continuations in part, divisions,
10 and/or the like thereof. As such, it should be understood that advantages, embodiments,
11 examples, functional, features, logical, operational, organizational, structural,
12 topological, and/or other aspects of the disclosure are not to be considered limitations
13 on the disclosure as defined by the claims or limitations on equivalents to the claims. It
14 is to be understood that, depending on the particular needs and/or characteristics of a
15 MCB-Platform individual and/or enterprise user, database configuration and/or
16 relational model, data type, data transmission and/or network framework, syntax
17 structure, and/or the like, various embodiments of the MCB-Platform, may be
18 implemented that enable a great deal of flexibility and customization. For example,
19 aspects of the MCB-Platform may be adapted for exchanging securities, rights,
20 obligations, debt, and/or the like. While various embodiments and discussions of the
21 MCB-Platform have been directed to currency exchange, however, it is to be understood
22 that the embodiments described herein may be readily configured and/or customized
23 for a wide variety of other applications and/or implementations.

CLAIMS

What is claimed is:

1. A merchant consumer bridging low-latency processor-implemented method, comprising:

receiving an activity request including merchant information associated with a merchant;

retrieving a previously stored merchant record from a database;

determining merchant information update indicia based on a comparison of the merchant information and the previously stored merchant record;

determining a confidence metric for the merchant information update;

retrieving a confidence requirement based on the activity request;

determining, within a low-latency processing time-frame, whether the determined confidence metric satisfies the retrieved confidence requirement;

performing the requested activity and updating the previously stored merchant record with the merchant information update indicia when the determined confidence metric satisfies the retrieved confidence requirement; and

declining the activity request when the determined confidence metric satisfies the retrieved confidence requirement.

2. The method of claim 1, wherein the activity request comprises any of a transaction payment request, a merchant database update request, and an offer issuance request.

1 3. The method of claim 1, wherein the activity request is received from a
2 merchant point of sale terminal.

3

4 4. The method of claim 2, wherein the transaction payment request further
5 comprises consumer wallet information.

6

7 5. The method of claim 1, wherein the merchant record comprises any of a
8 merchant ID, merchant address, merchant inventory, merchant alias, and merchant
9 name.

10

11 6. The method of claim 1, wherein the merchant record comprises any of a
12 merchant URL, merchant web server ID, and merchant web server address.

13

14 7. The method of claim 1, further comprising:
15 forming a query on the merchant database based on a merchant ID.

16

17 8. The method of claim 1, wherein merchant information update indicia
18 comprises an inconsistent data field between the received merchant information and the
19 previously stored merchant record.

20

21 9. The method of claim 1, wherein the confidence metric comprise a numeric
22 value.

23

24 10. The method of claim 1, wherein the confidence metric is determined by a

1 scoring component.

2

3 11. The method of claim 10, wherein the scoring component receives a variety
4 of merchant related information.

5

6 12. The method of claim 11, wherein the variety of merchant related
7 information comprises web claws from merchant websites.

8

9 13. The method of claim 11, wherein the variety of merchant related
10 information comprises merchant statistics.

11

12 14. The method of claim 11, wherein the variety of merchant related
13 information comprises prior transaction history with the merchant.

14

15 15. The method of claim 11, wherein the scoring component generate the
16 confidence metric based on the received variety of merchant related information,
17 wherein the confidence metric is indicative of a confidence level of accuracy of the
18 merchant information update indicia.

19

20 16. The method of claim 1, wherein the confidence requirement comprises a
21 numeric confidence metric threshold.

22

23 17. The method of claim 1, wherein the confidence requirement is dependent
24 on the activity type.

1

2 18. The method of claim 1, wherein the confidence requirement comprises a
3 threshold value of 0.8 to allow a transaction request.

4

5 19. The method of claim 1, wherein the confidence requirement comprises a
6 threshold value of 0.5 to allow an offer issuance request.

7

8 20. The method of claim 1, wherein the confidence requirement comprises a
9 threshold value of 0.5 to allow a merchant record update request.

10

11 21. A merchant consumer bridging system, comprising:

12 a memory;

13 a processor disposed in communication with said memory, and configured to
14 issue a plurality of processing instructions stored in the memory, wherein the processor
15 issues instructions to:

16 receive an activity request including merchant information associated with a
17 merchant;

18 retrieve a previously stored merchant record from a database;

19 determine merchant information update indicia based on a comparison of the
20 merchant information and the previously stored merchant record;

21 determine a confidence metric for the merchant information update;

22 retrieve a confidence requirement based on the activity request;

23 determine, within a low-latency processing time-frame, whether the determined
24 confidence metric satisfies the retrieved confidence requirement;

1 perform the requested activity and updating the previously stored merchant
2 record with the merchant information update indicia when the determined confidence
3 metric satisfies the retrieved confidence requirement; and
4 decline the activity request when the determined confidence metric satisfies the
5 retrieved confidence requirement.

6

7 22. The system of claim 21, wherein the activity request comprises any of a
8 transaction payment request, a merchant database update request, and an offer
9 issuance request.

10

11 23. The system of claim 21, wherein the activity request is received from a
12 merchant point of sale terminal.

13

14 24. The system of claim 22, wherein the transaction payment request further
15 comprises consumer wallet information.

16

17 25. The system of claim 21, wherein the merchant record comprises any of a
18 merchant ID, merchant address, merchant inventory, merchant alias, and merchant
19 name.

20

21 26. The system of claim 21, wherein the merchant record comprises any of a
22 merchant URL, merchant web server ID, and merchant web server address.

23

24 27. The system of claim 21, wherein the processor further issues instructions

1 to:

2 form a query on the merchant database based on a merchant ID.

3

4 28. The system of claim 21, wherein merchant information update indicia
5 comprises an inconsistent data field between the received merchant information and the
6 previously stored merchant record.

7

8 29. The system of claim 21, wherein the confidence metric comprise a numeric
9 value.

10

11 30. The system of claim 21, wherein the confidence metric is determined by a
12 scoring component.

13

14 31. The system of claim 30, wherein the scoring component receives a variety
15 of merchant related information.

16

17 32. The system of claim 31, wherein the variety of merchant related
18 information comprises web claws from merchant websites.

19

20 33. The system of claim 31, wherein the variety of merchant related
21 information comprises merchant statistics.

22

23 34. The system of claim 31, wherein the variety of merchant related
24 information comprises prior transaction history with the merchant.

1

2 35. The system of claim 31, wherein the scoring component generate the
3 confidence metric based on the received variety of merchant related information,
4 wherein the confidence metric is indicative of a confidence level of accuracy of the
5 merchant information update indicia.

6

7 36. The system of claim 21, wherein the confidence requirement comprises a
8 numeric confidence metric threshold.

9

10 37. The system of claim 21, wherein the confidence requirement is dependent
11 on the activity type.

12

13 38. The system of claim 21, wherein the confidence requirement comprises a
14 threshold value of 0.8 to allow a transaction request.

15

16 39. The system of claim 21, wherein the confidence requirement comprises a
17 threshold value of 0.5 to allow an offer issuance request.

18

19 40. The system of claim 21, wherein the confidence requirement comprises a
20 threshold value of 0.5 to allow a merchant record update request.

21

22 41. A merchant consumer bridging processor-readable storage medium
23 storing processor-executable instructions executable by a processor to:

24 receive an activity request including merchant information associated with a

1 merchant;

2 retrieve a previously stored merchant record from a database;

3 determine merchant information update indicia based on a comparison of the

4 merchant information and the previously stored merchant record;

5 determine a confidence metric for the merchant information update;

6 retrieve a confidence requirement based on the activity request;

7 determine, within a low-latency processing time-frame, whether the determined

8 confidence metric satisfies the retrieved confidence requirement;

9 perform the requested activity and updating the previously stored merchant

10 record with the merchant information update indicia when the determined confidence

11 metric satisfies the retrieved confidence requirement; and

12 decline the activity request when the determined confidence metric satisfies the

13 retrieved confidence requirement.

14

15 42. The medium of claim 41, wherein the activity request comprises any of a

16 transaction payment request, a merchant database update request, and an offer

17 issuance request.

18

19 43. The medium of claim 41, wherein the activity request is received from a

20 merchant point of sale terminal.

21

22 44. The medium of claim 42, wherein the transaction payment request further

23 comprises consumer wallet information.

24

1 45. The medium of claim 41, wherein the merchant record comprises any of a
2 merchant ID, merchant address, merchant inventory, merchant alias, and merchant
3 name.

4

5 46. The medium of claim 41, wherein the merchant record comprises any of a
6 merchant URL, merchant web server ID, and merchant web server address.

7

8 47. The medium of claim 41, wherein the processor further issues instructions
9 to:

10 form a query on the merchant database based on a merchant ID.

11

12 48. The medium of claim 41, wherein merchant information update indicia
13 comprises an inconsistent data field between the received merchant information and the
14 previously stored merchant record.

15

16 49. The medium of claim 41, wherein the confidence metric comprise a
17 numeric value.

18

19 50. The medium of claim 41, wherein the confidence metric is determined by a
20 scoring component.

21

22 51. The system of claim 50, wherein the scoring component receives a variety
23 of merchant related information.

24

1 52. The system of claim 51, wherein the variety of merchant related
2 information comprises web claws from merchant websites.

3

4 53. The system of claim 51, wherein the variety of merchant related
5 information comprises merchant statistics.

6

7 54. The system of claim 51, wherein the variety of merchant related
8 information comprises prior transaction history with the merchant.

9

10 55. The system of claim 51, wherein the scoring component generate the
11 confidence metric based on the received variety of merchant related information,
12 wherein the confidence metric is indicative of a confidence level of accuracy of the
13 merchant information update indicia.

14

15 56. The medium of claim 41, wherein the confidence requirement comprises a
16 numeric confidence metric threshold.

17

18 57. The medium of claim 41, wherein the confidence requirement is
19 dependent on the activity type.

20

21 58. The medium of claim 41, wherein the confidence requirement comprises a
22 threshold value of 0.8 to allow a transaction request.

23

24 59. The medium of claim 41, wherein the confidence requirement comprises a

1 threshold value of 0.5 to allow an offer issuance request.

2

3 60. The medium of claim 41, wherein the confidence requirement comprises a
4 threshold value of 0.5 to allow a merchant record update request.

5

6 61. A merchant-consumer bridging processor-implemented method to
7 transform consumer and merchant registration information to product purchase
8 transaction, comprising:

9 receiving consumer indentifying information from a consumer electronic wallet
10 vehicle;

11 receiving merchant information and a proposed transaction from a merchant
12 terminal;

13 verifying consumer and merchant qualification to engage in the proposed
14 transaction;

15 initiating consumer payment by sending a payment approval to an electronic
16 wallet account; and

17 completing the proposed transaction by sending an approval to the merchant
18 terminal.

19

20 62. A merchant-consumer bridging processor-implemented method to
21 transform consumer and merchant registration information to product purchase
22 transaction, comprising:

23 receiving information related to consumer opt-in activities of a consumer from a
24 merchant;

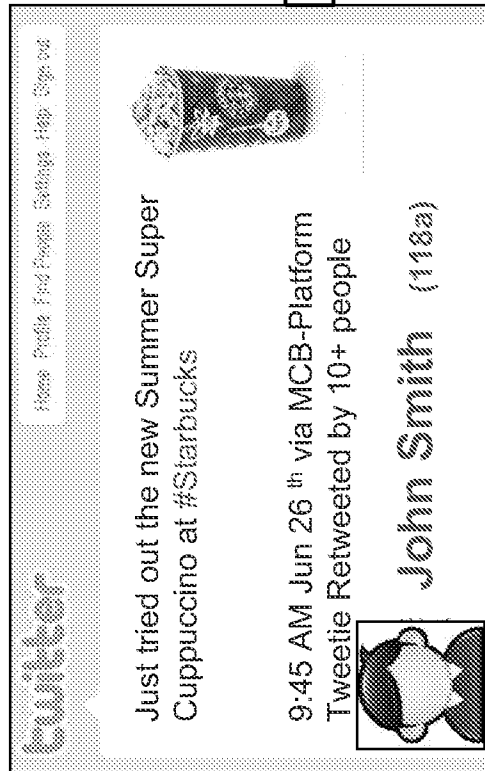
- 1 retrieve a record of merchant offers;
- 2 generating a query based on the received information on the record of merchant
- 3 offers;
- 4 obtaining a list of matched offers from the query; and
- 5 sending the list of matched offers to the consumer.

6

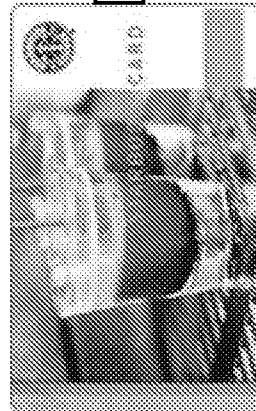
Prompting Offers 1 07

Determining Matching Merchant 1 06

Consumer Opt-in Activities 1 05

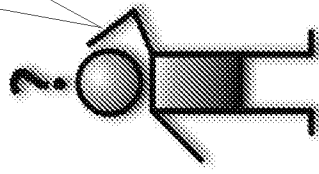
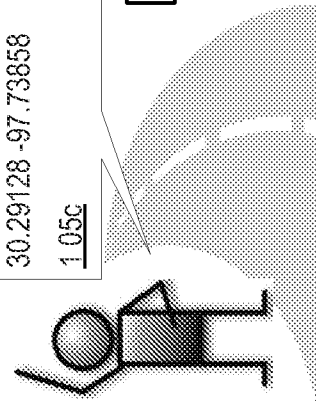


Social Media Activities 1 05a



Consumption Habit 1 05b

GPS Coordinates:
30.29128 -97.73858
1 05c

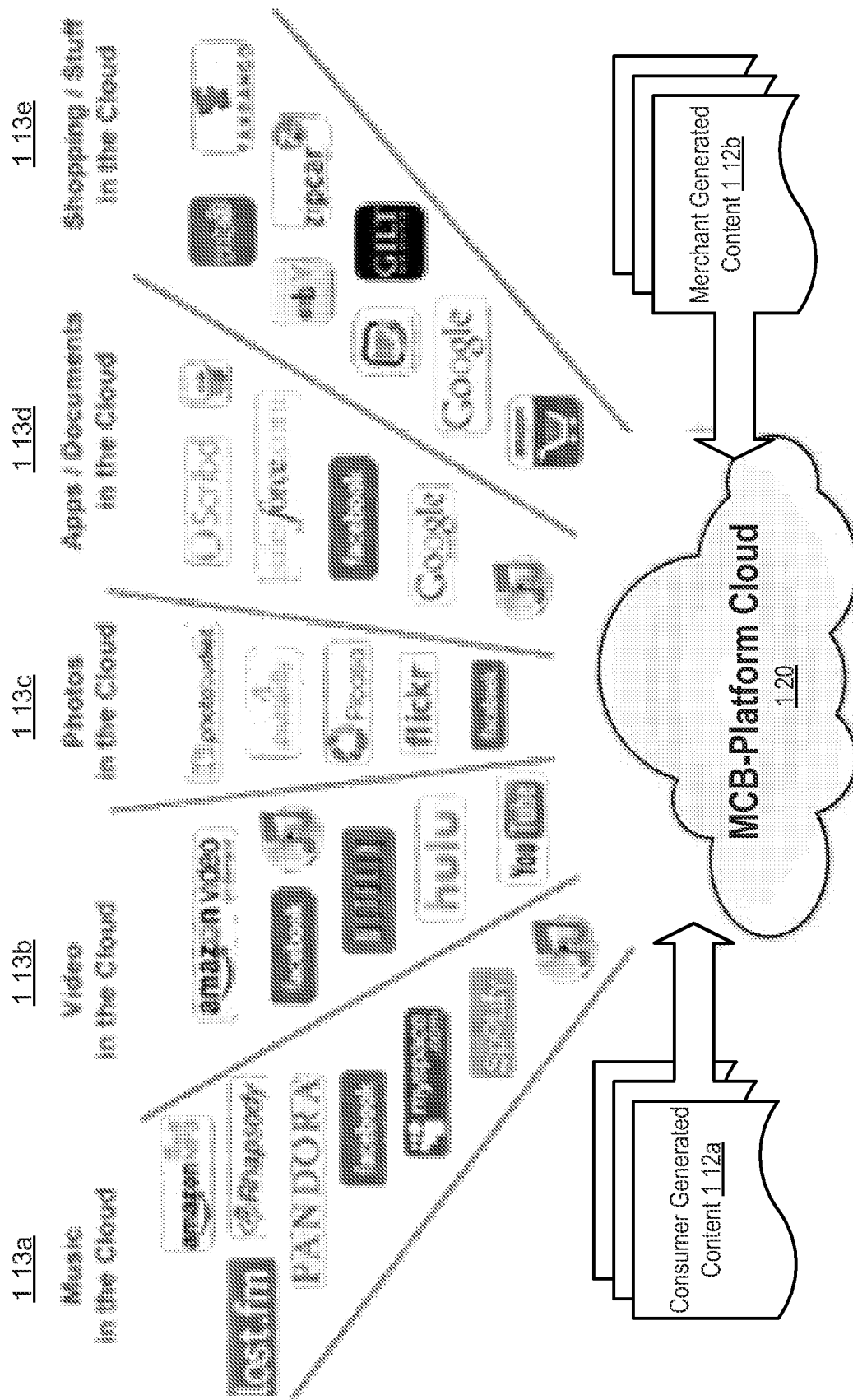


Consumer(s)
1 02



Figure 1A

MCB-Platform Example: Matching Merchant Offers to Consumers 1 00a



2015

MCB-Platform Example: Merchant-Consumer Cloud 100b

Figure 1C.(a)

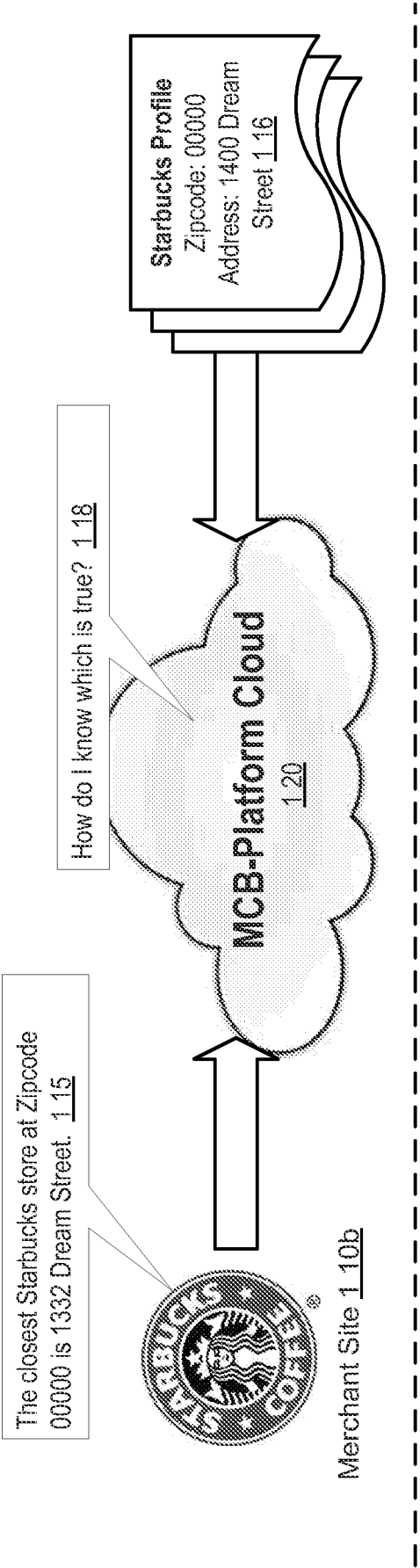


Figure 1C.(b)

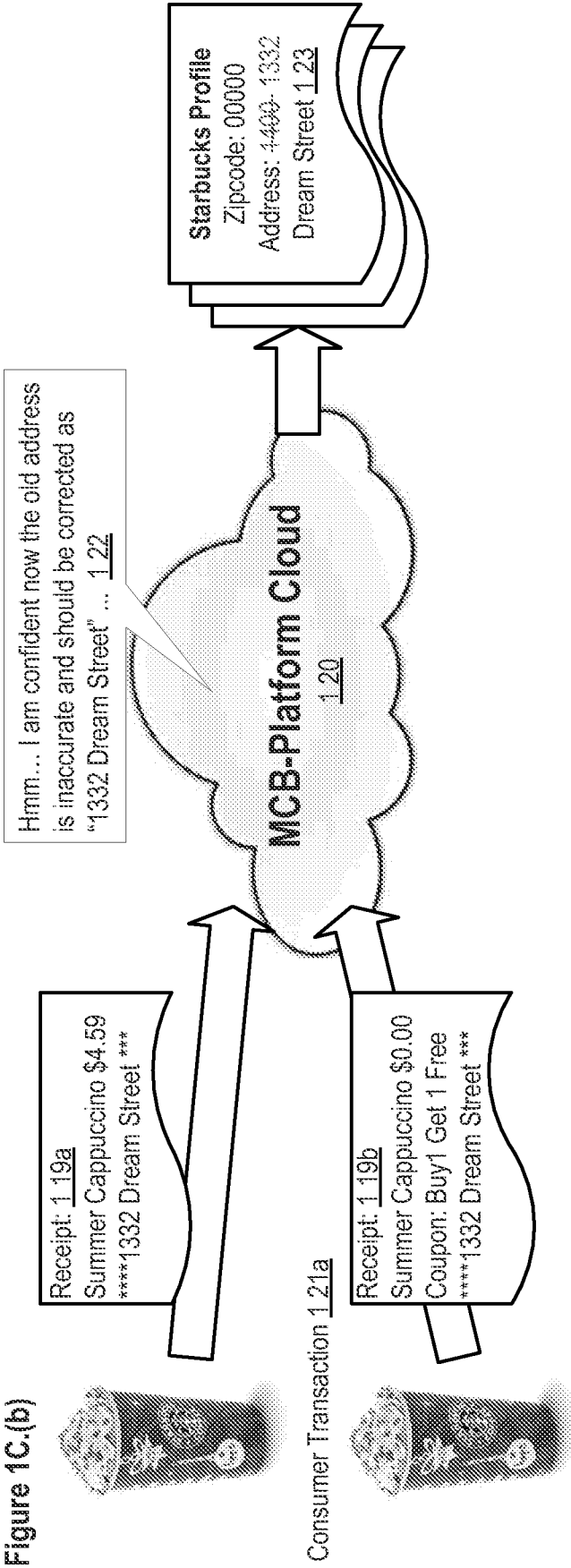


Figure 1C

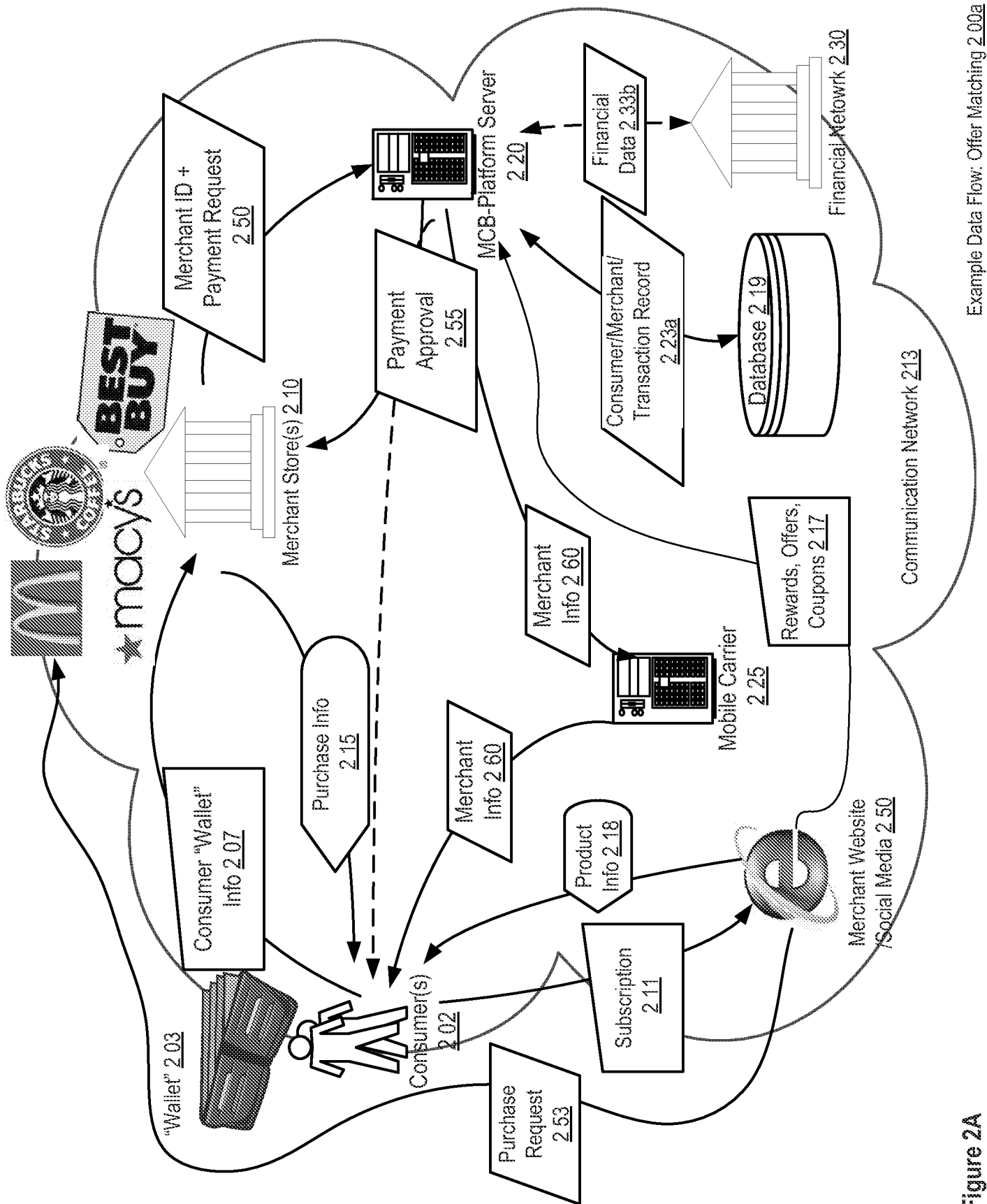


Figure 2A

Example Data Flow: Offer Matching 2.00a

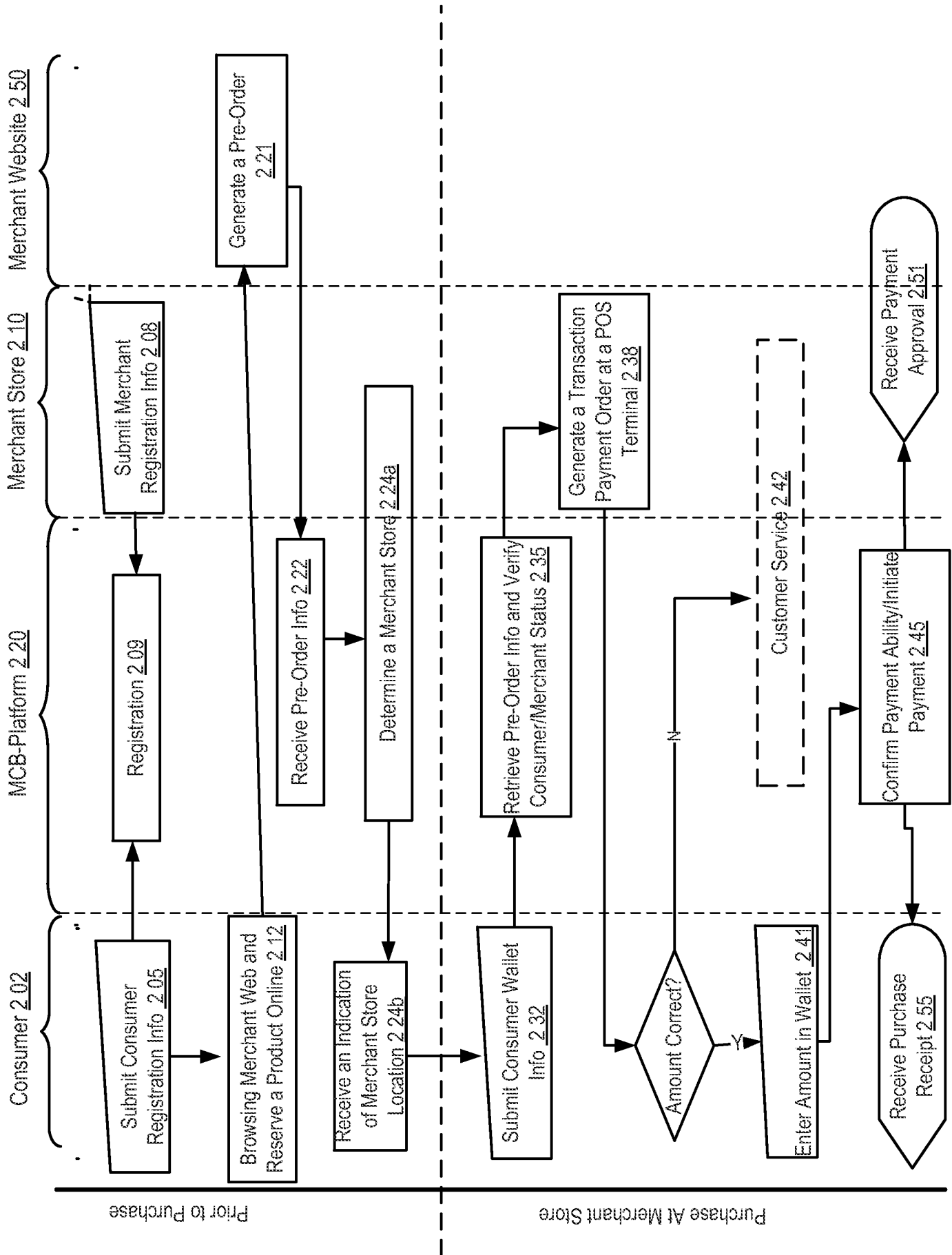
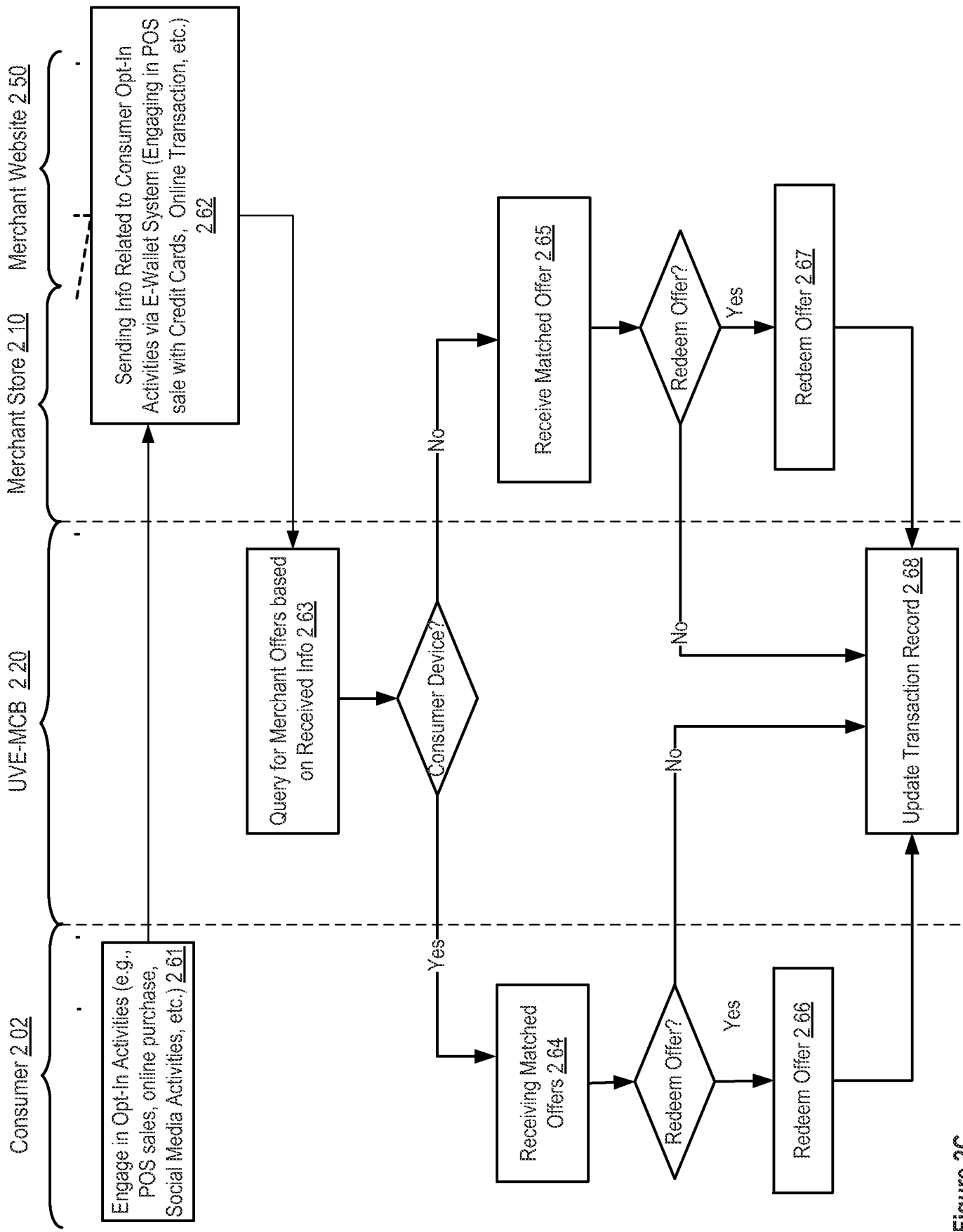


Figure 2B

Example Logic Flow: Offer Matching 2 00b



Example Logic Flow: Offer Matching 200b

Figure 2C

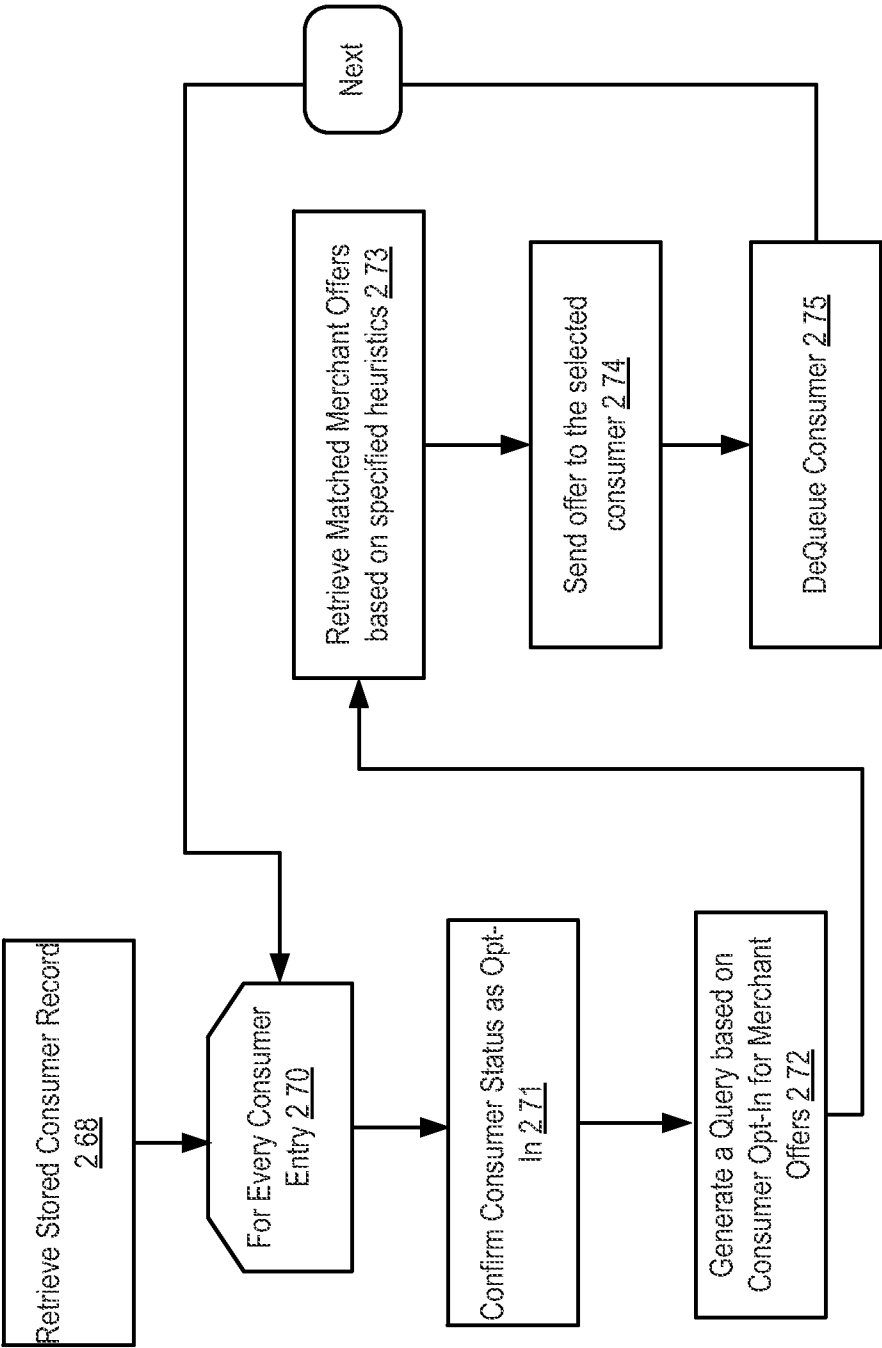


Figure 2D

Example Logic Flow: Consumer-Merchant Matching 2 00d

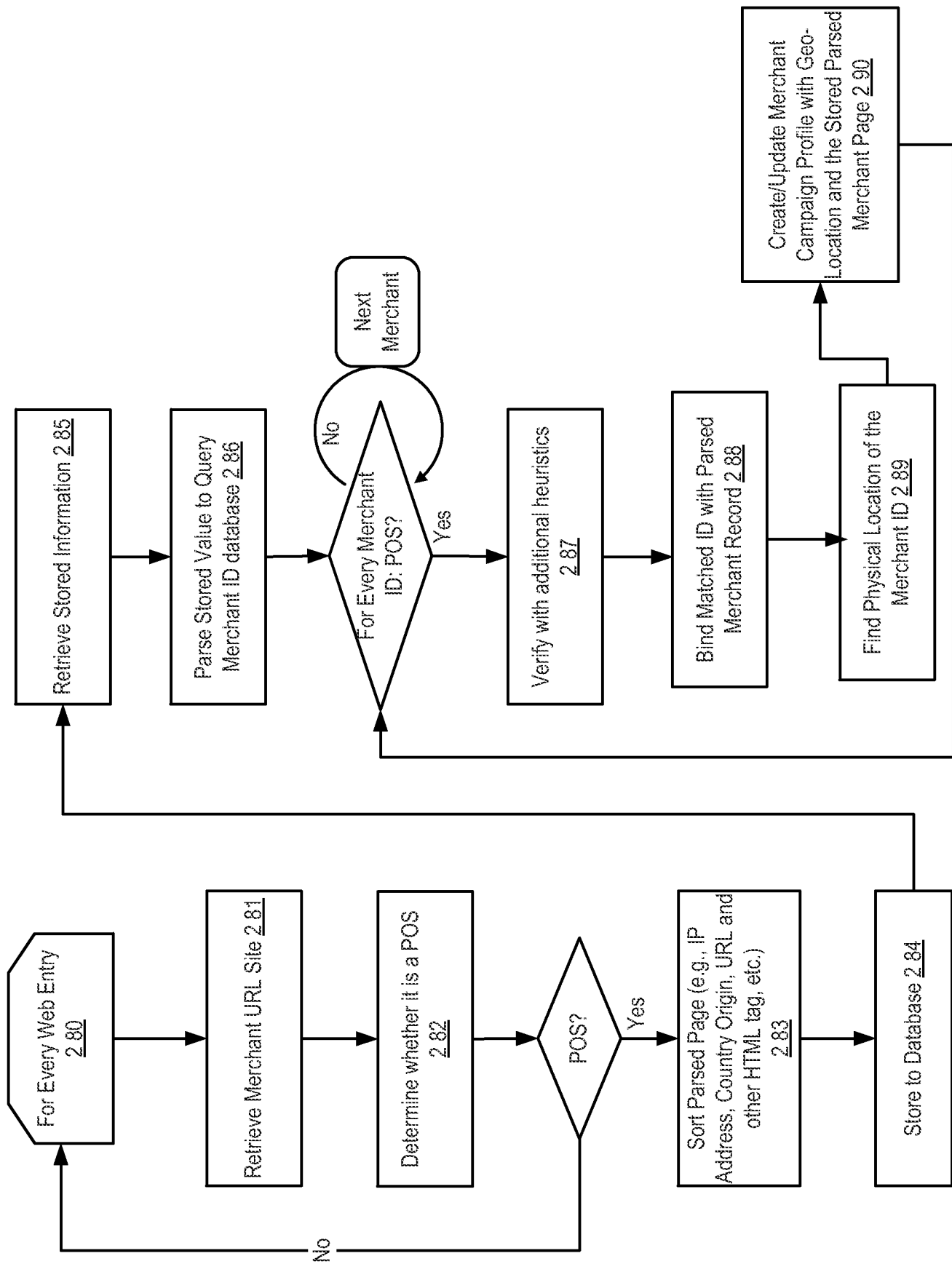
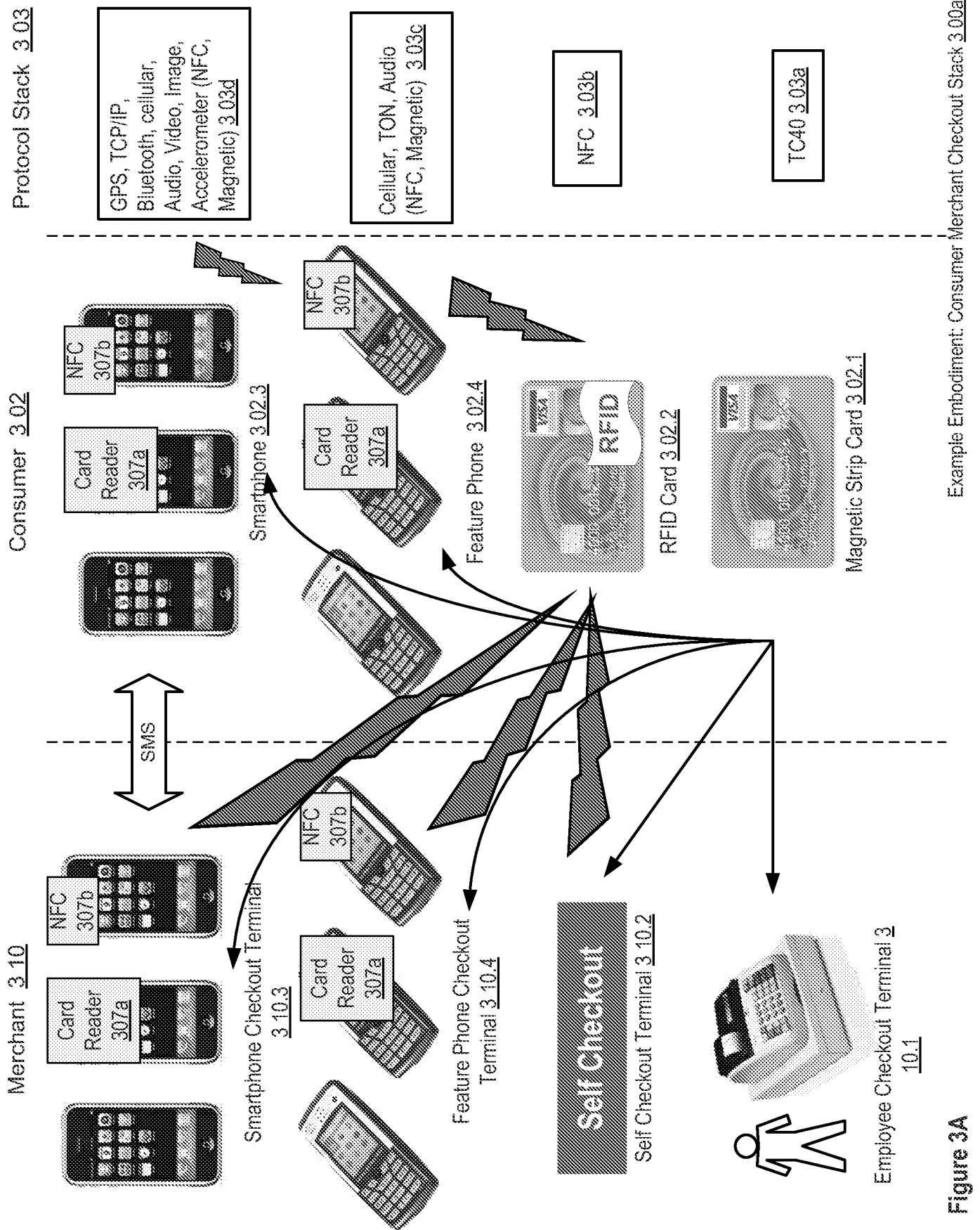
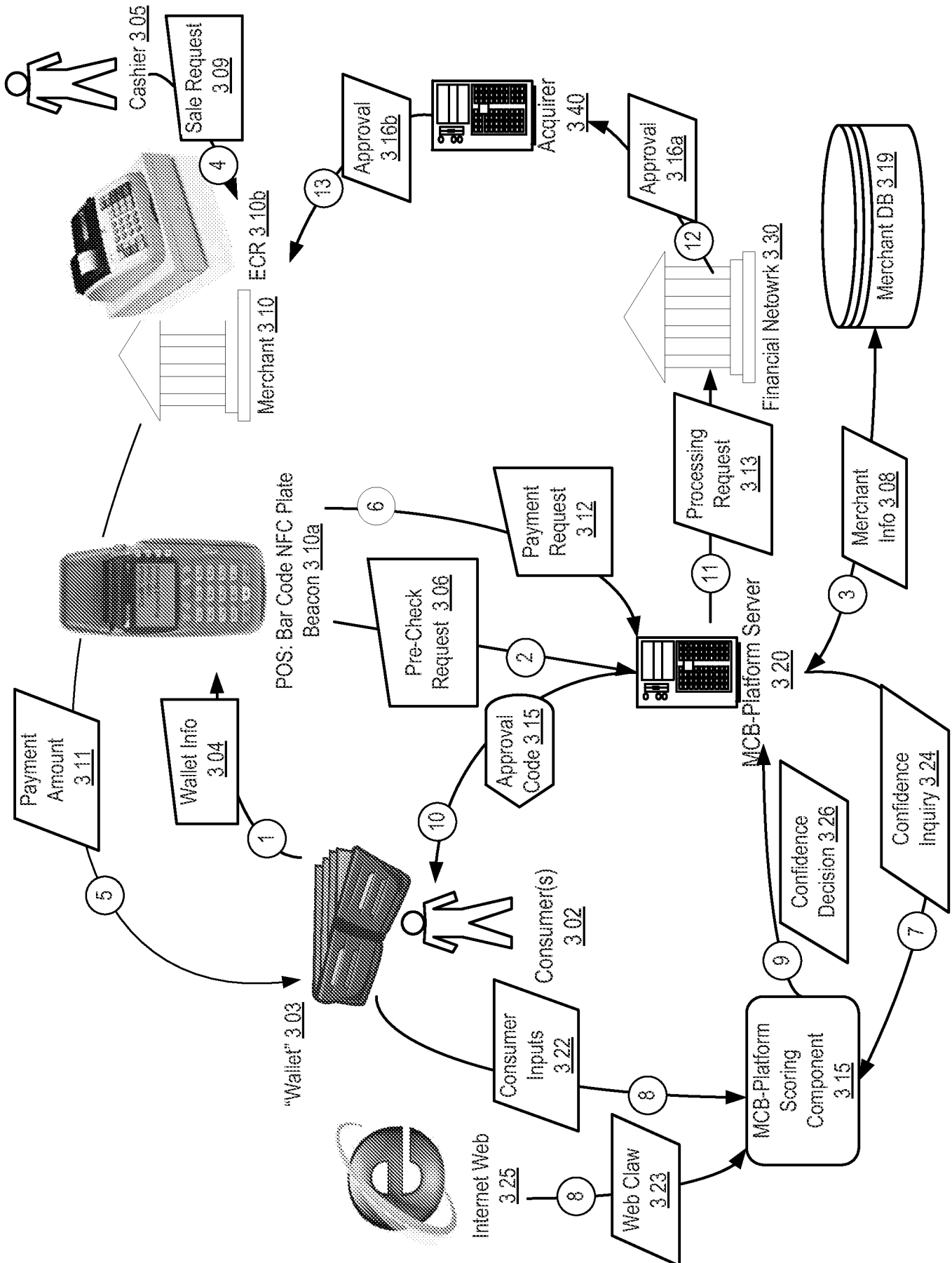


Figure 2E

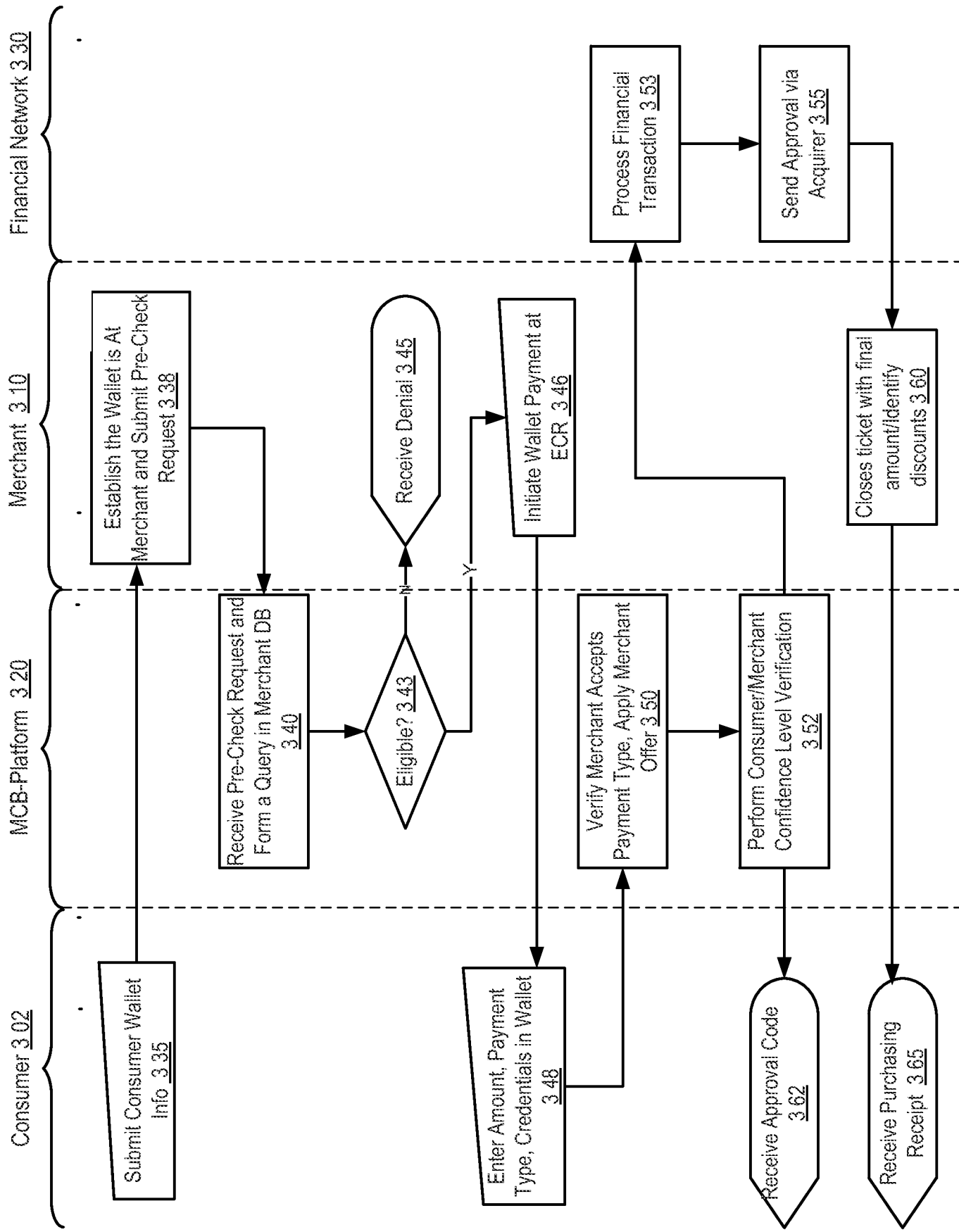
Example Logic Flow: Merchant Information 2.00e





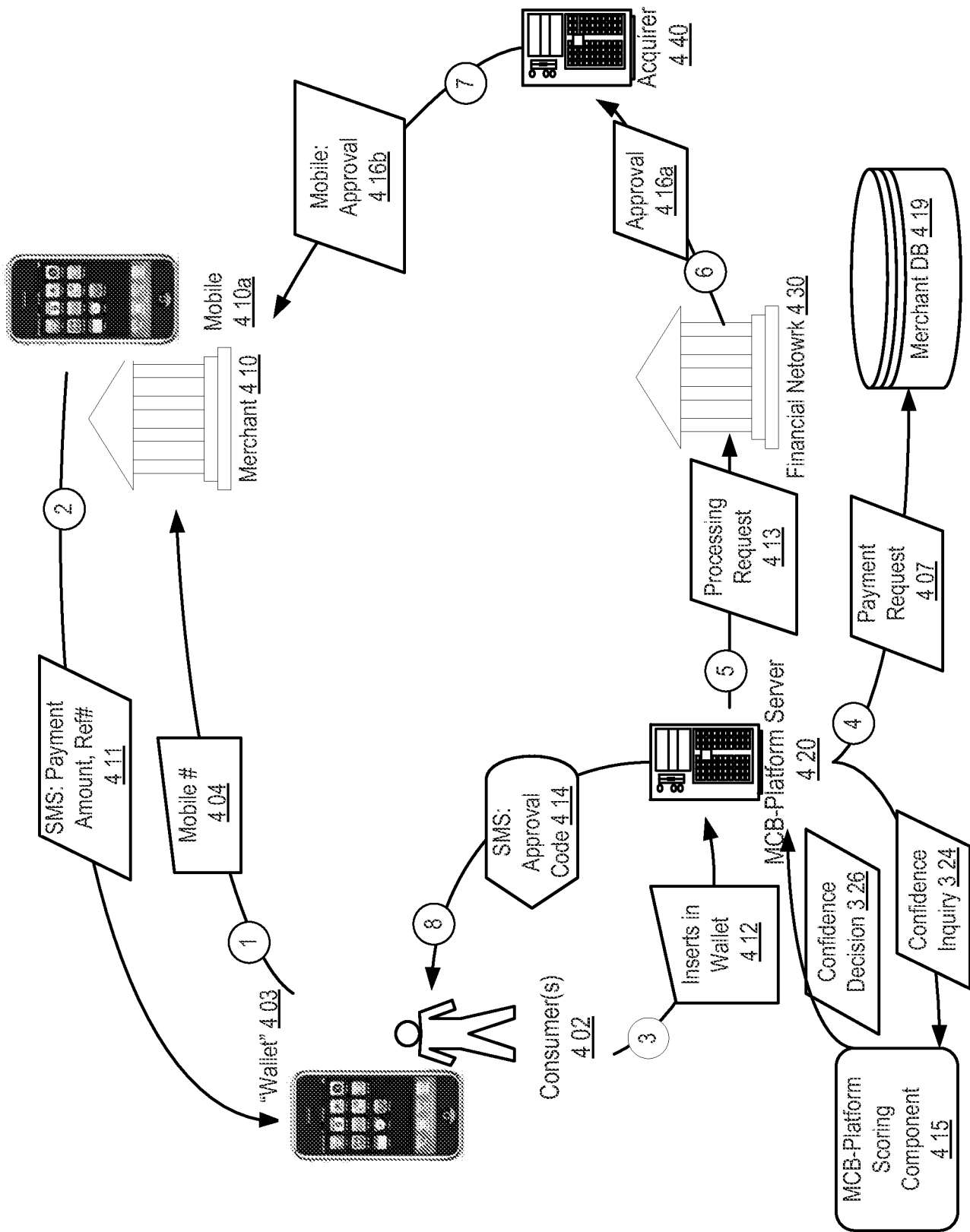
2025

Example Data Flow: Transaction at Merchant POS



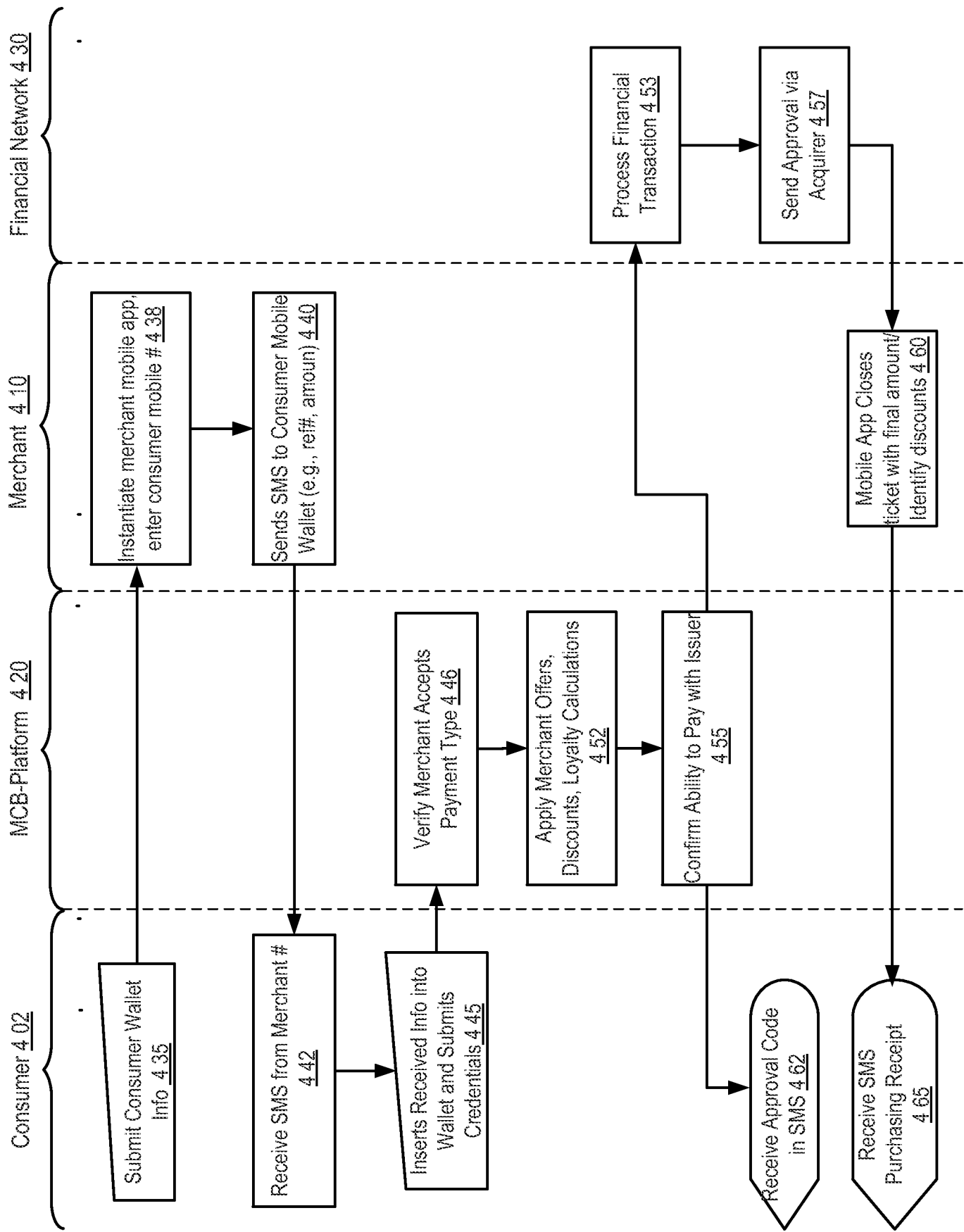
Example Logic Flow: Transaction at Merchant POS

Figure 3C



Example Data Flow: Mobile to Mobile POS Redemption

Figure 4A



Example Logic Flow: Mobile to Mobile POS Redemption

Figure 4B

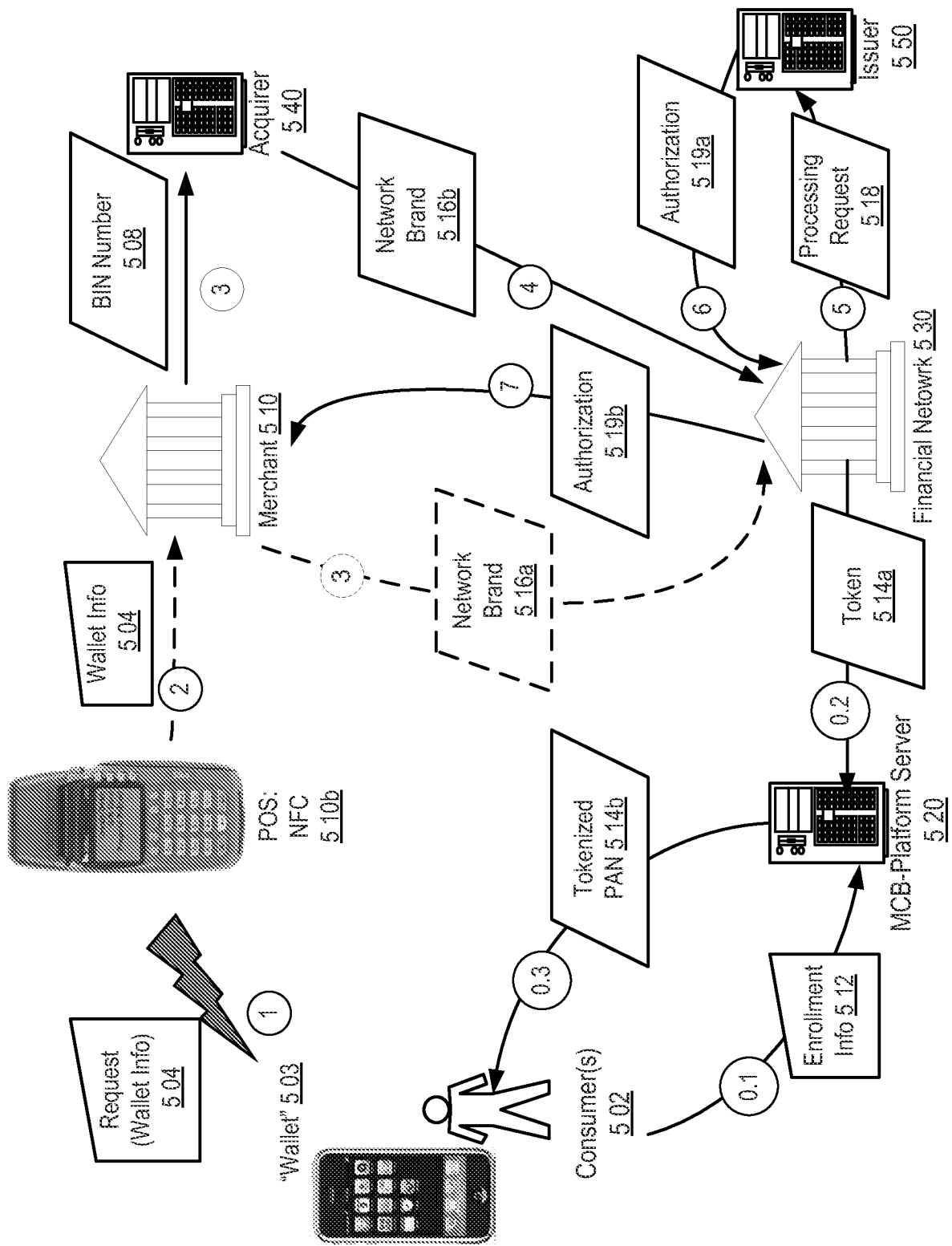


Figure 5A

Example Data Flow: Enrollment Tokenization

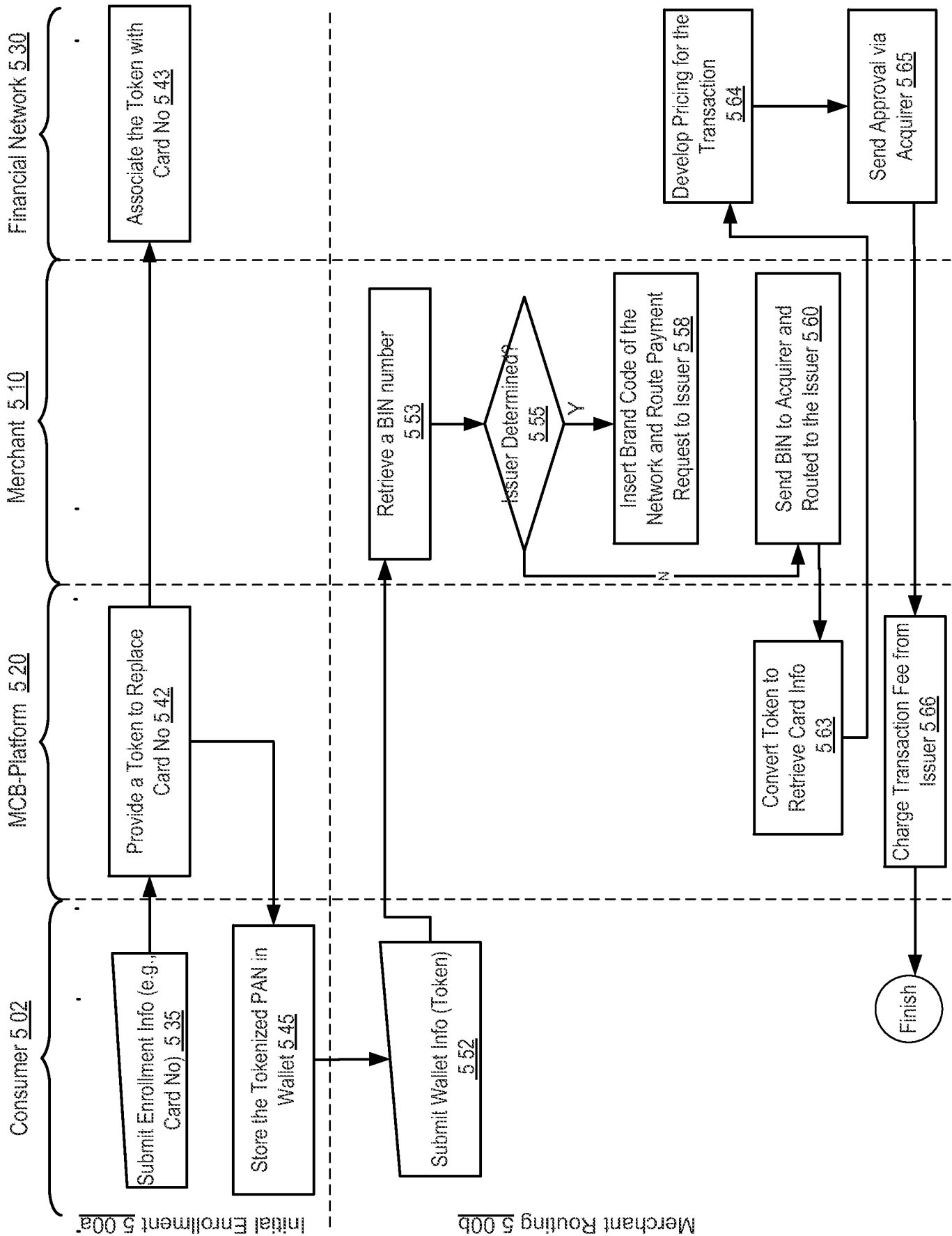
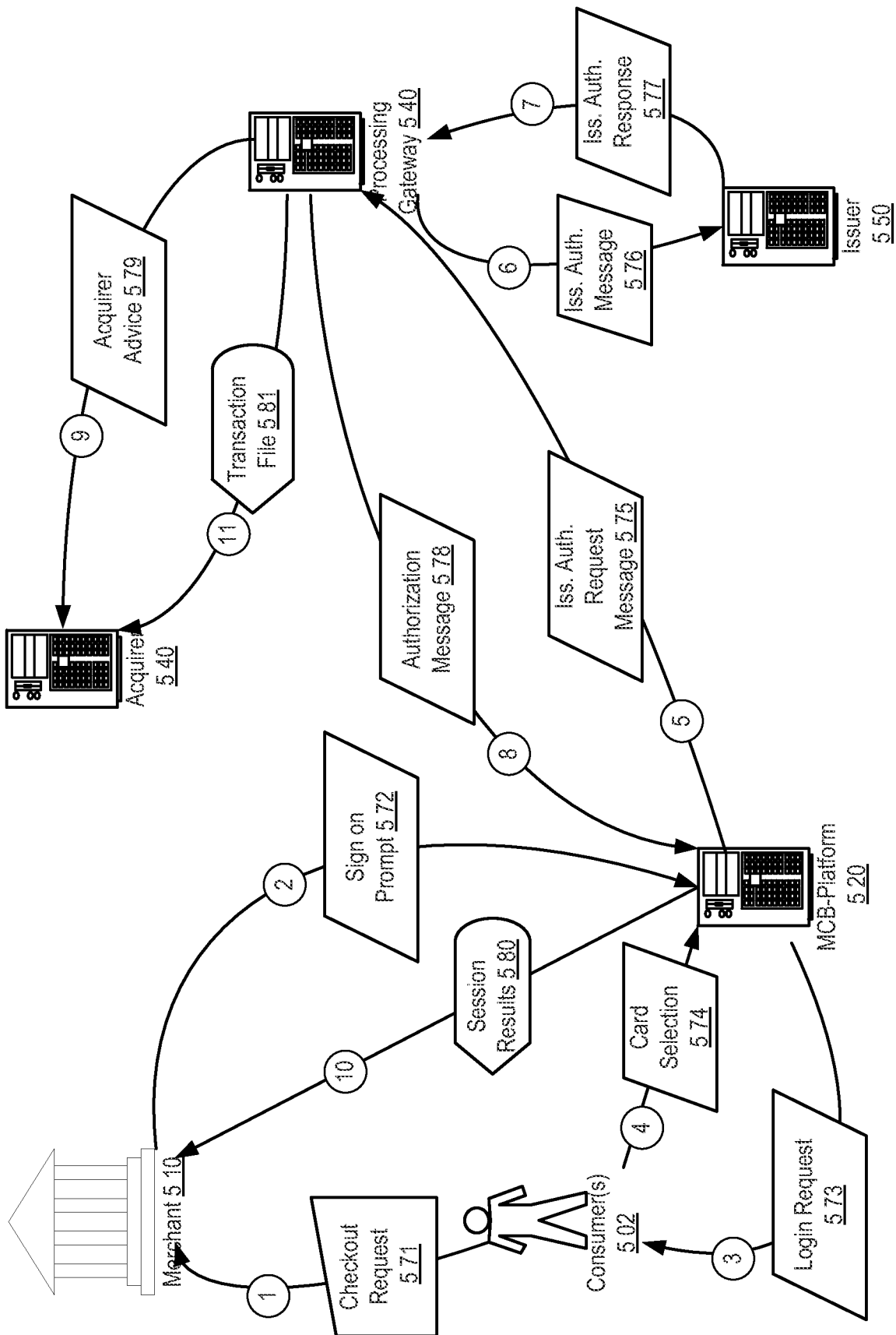


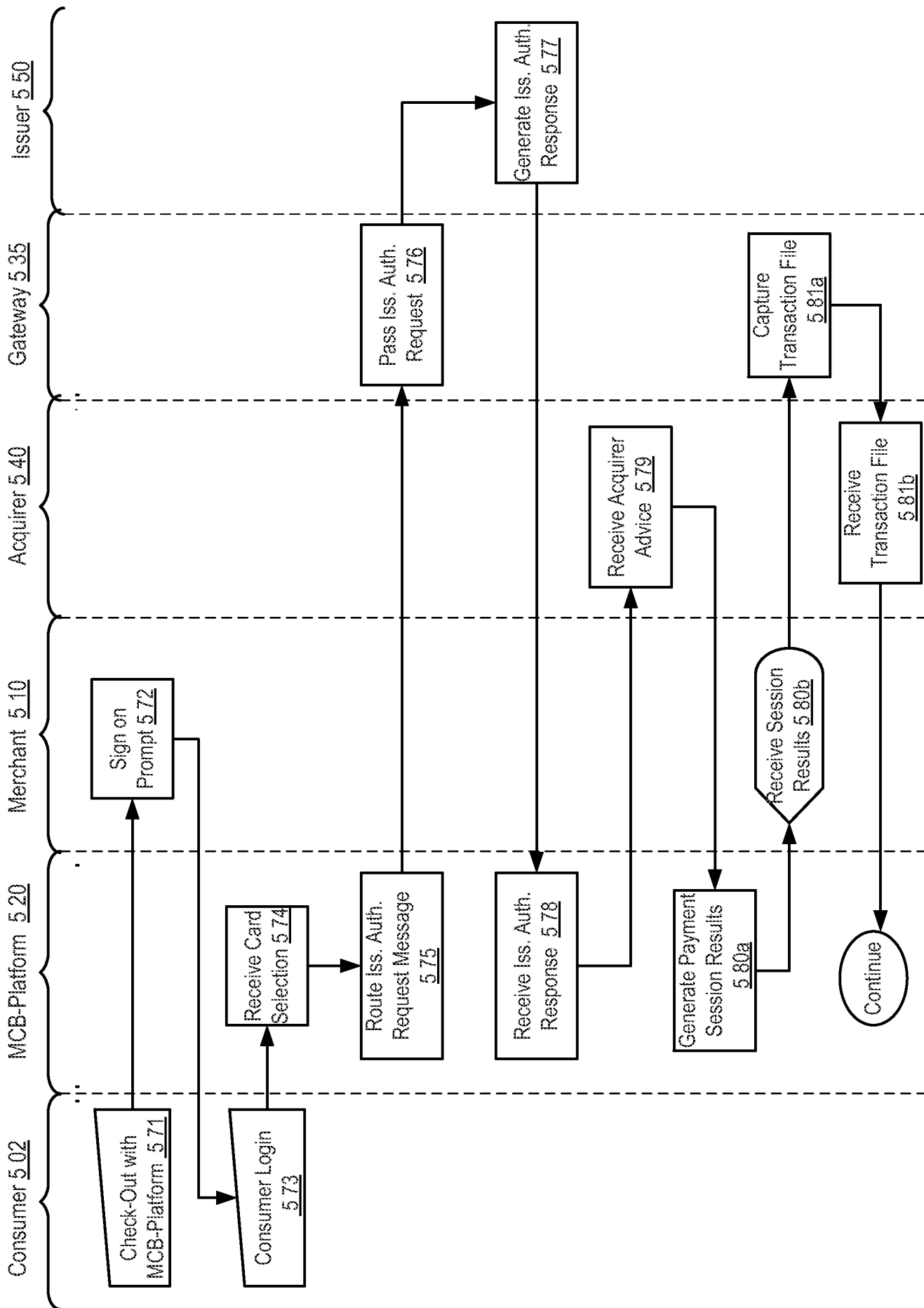
Figure 5B

Example Logic Flow: Enrollment Tokenization



Example Data Flow: Acquirer Routing 560

Figure 5C



Example Logic Flow: Acquirer Routing 560

Figure 5D

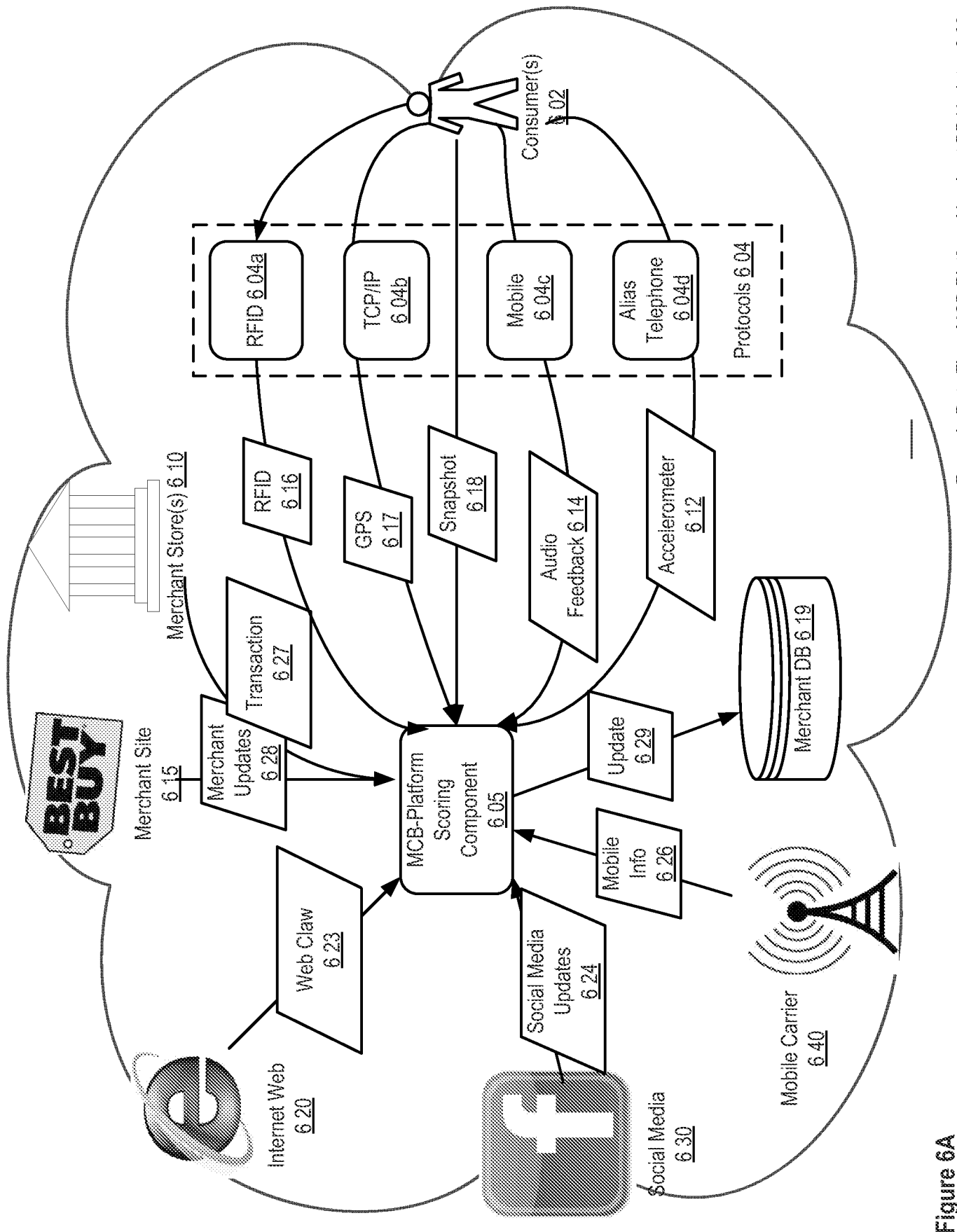
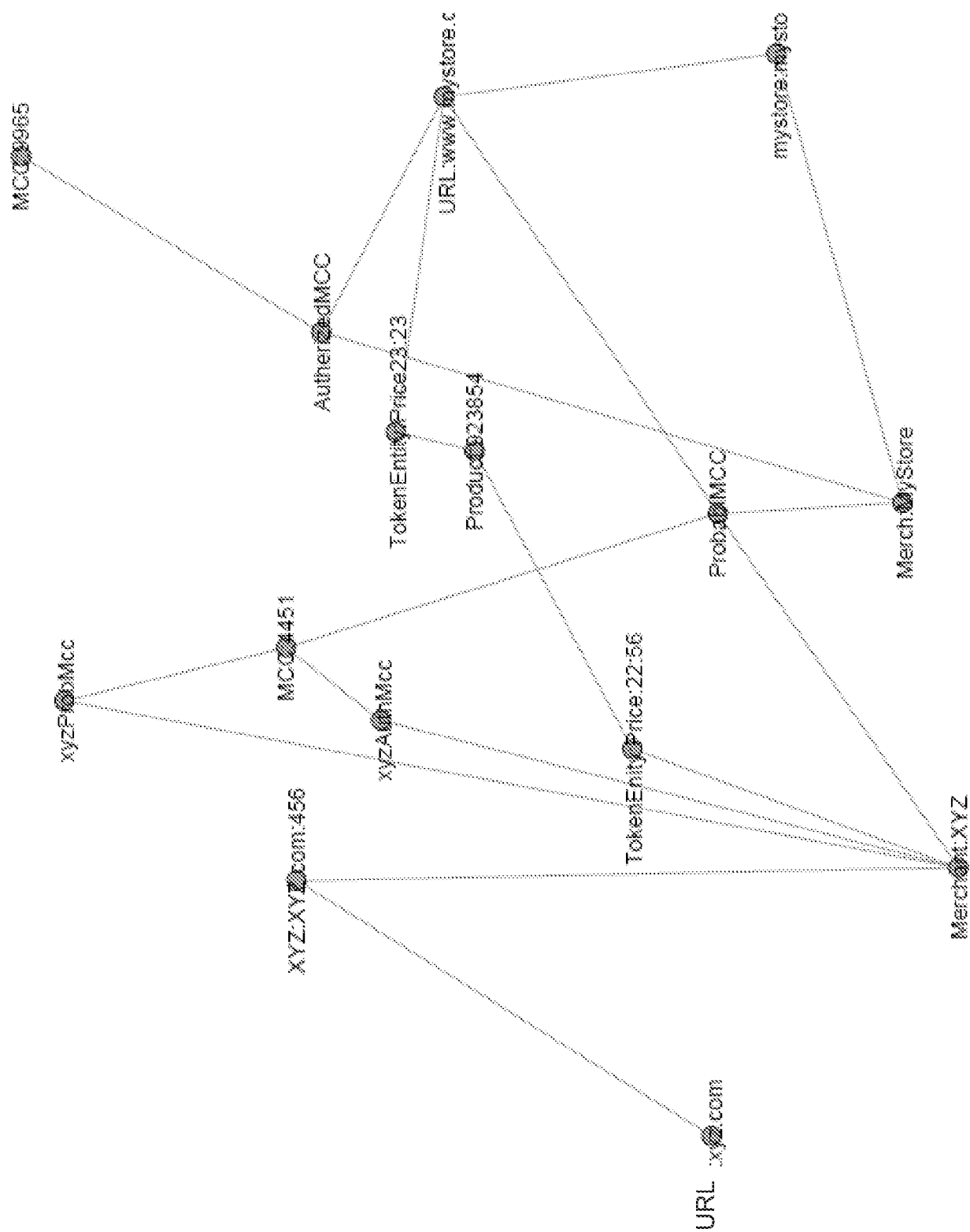


Figure 6A

Example Data Flow: MCB-Platform Merchant DB Updates 600a



3000

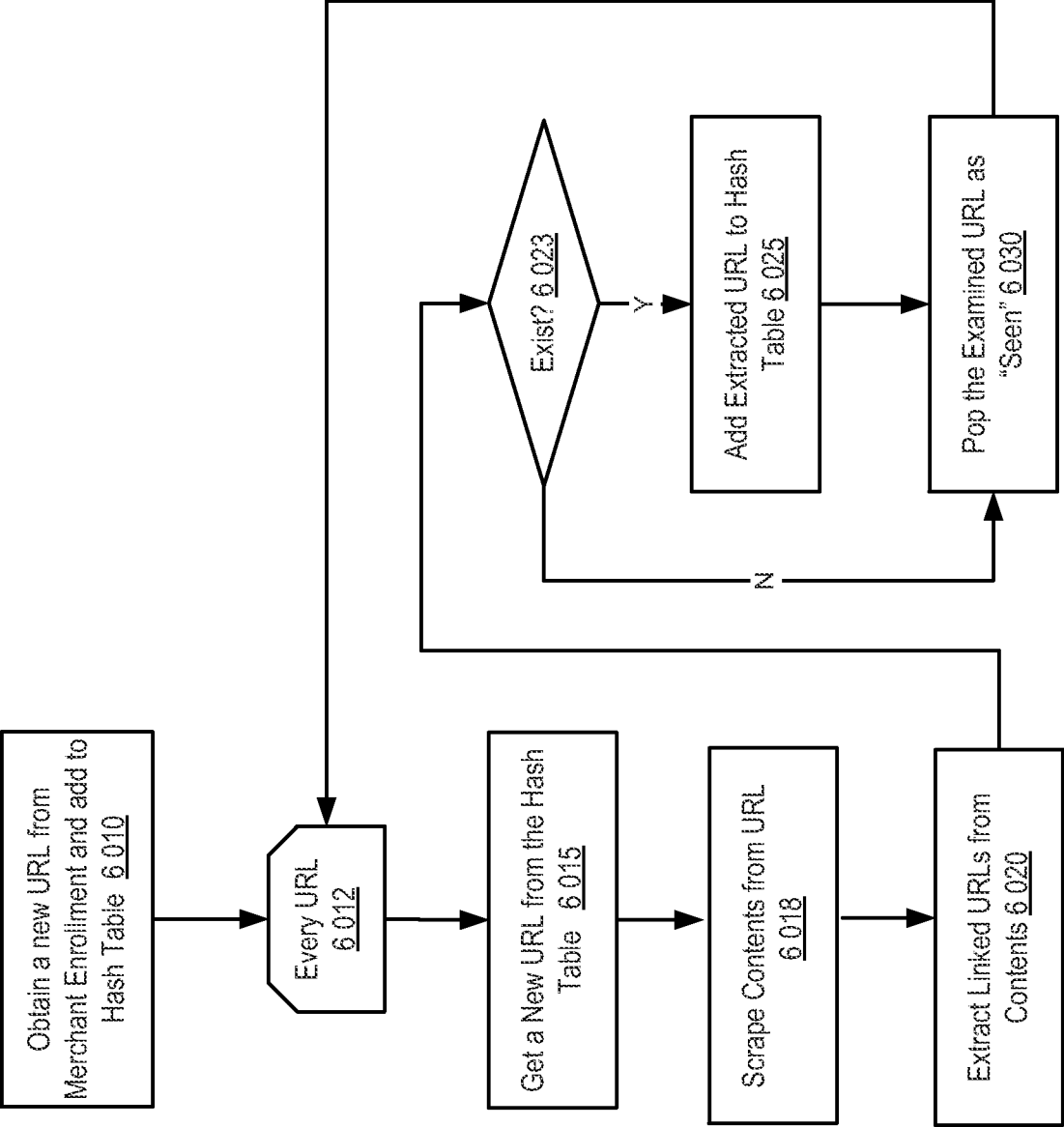


Figure 6C

Example Logic Flow: "Spider" Merchant Sites

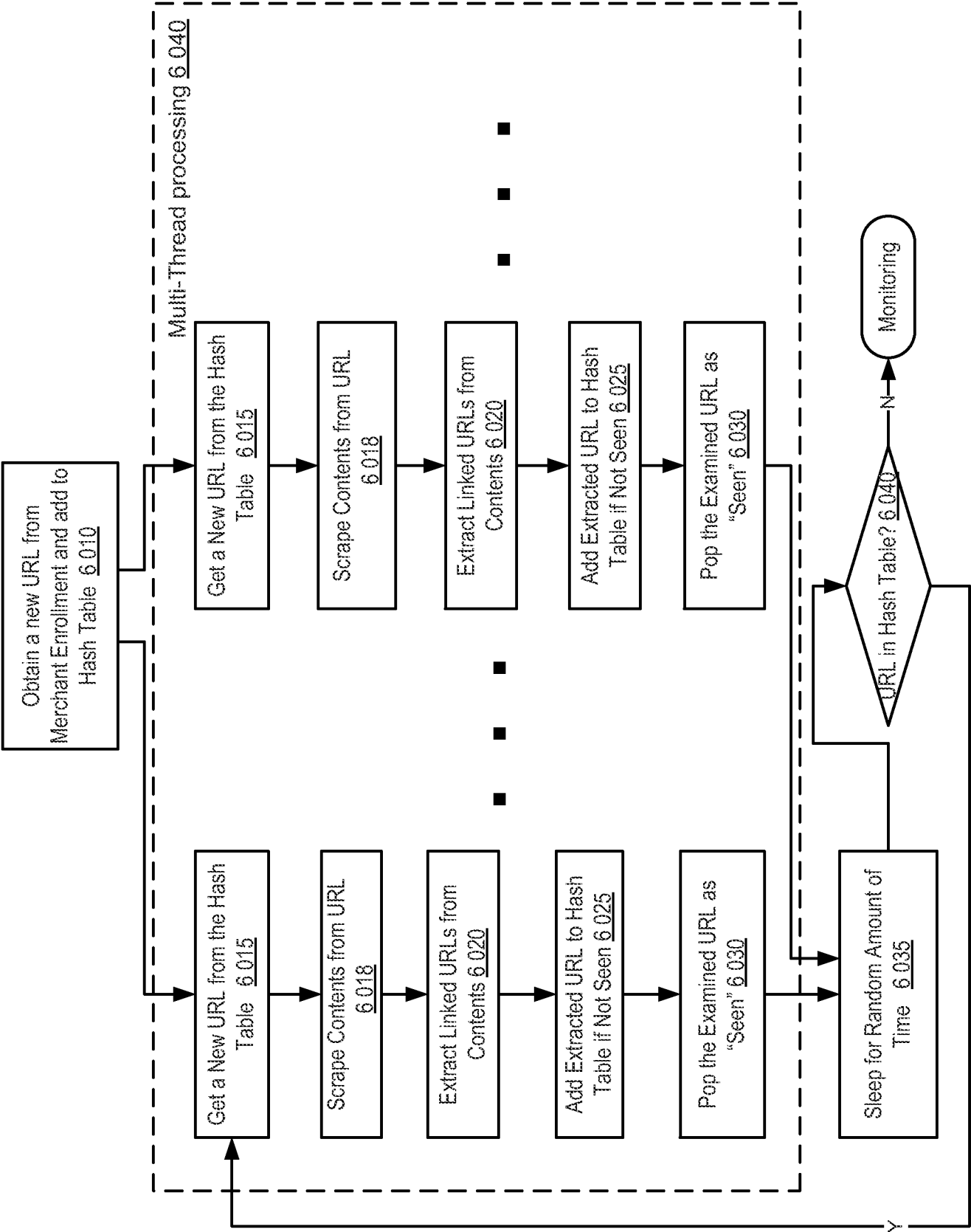


Figure 6D

Example Logic Flow: "Spider" Merchant Sites

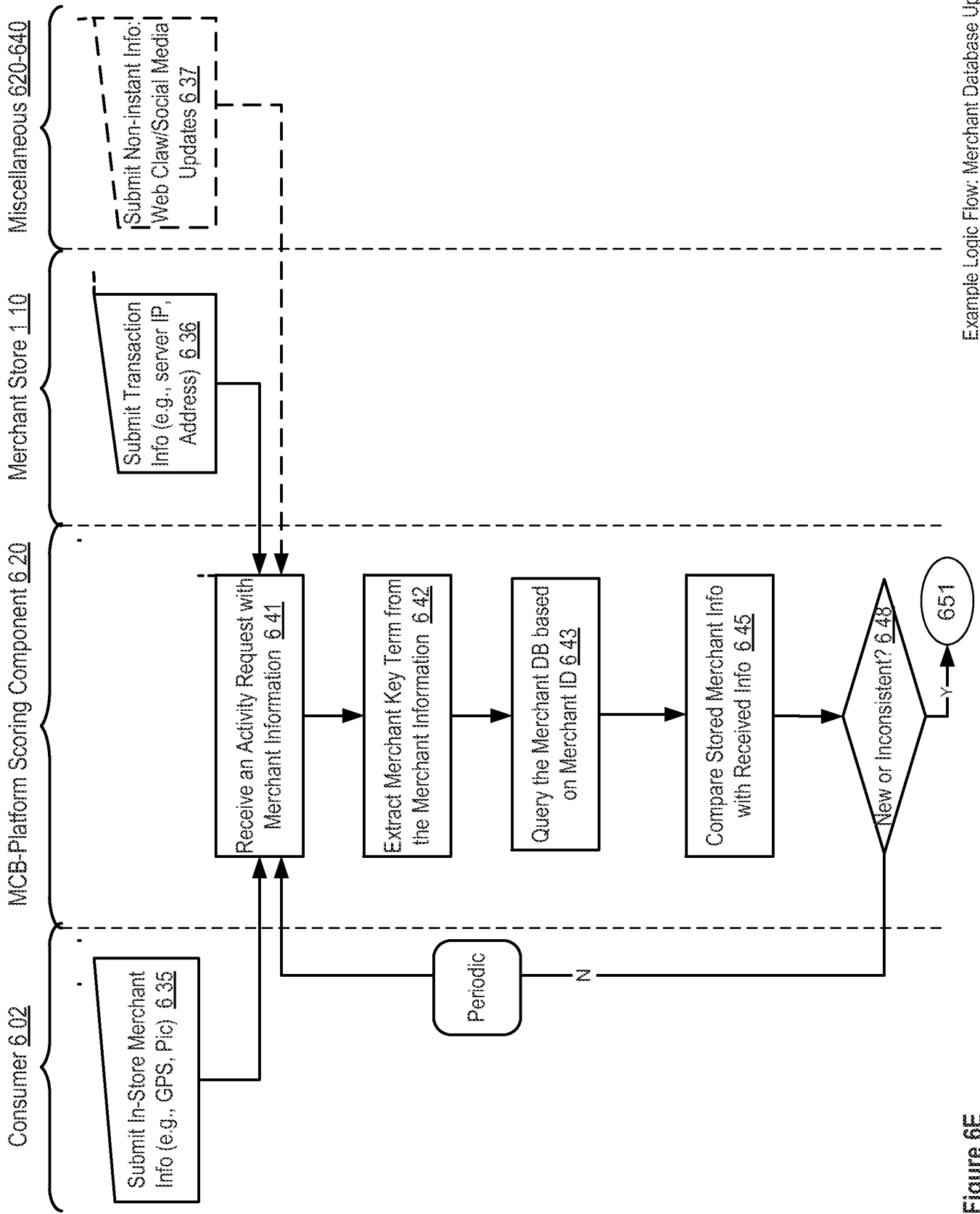


Figure 6E

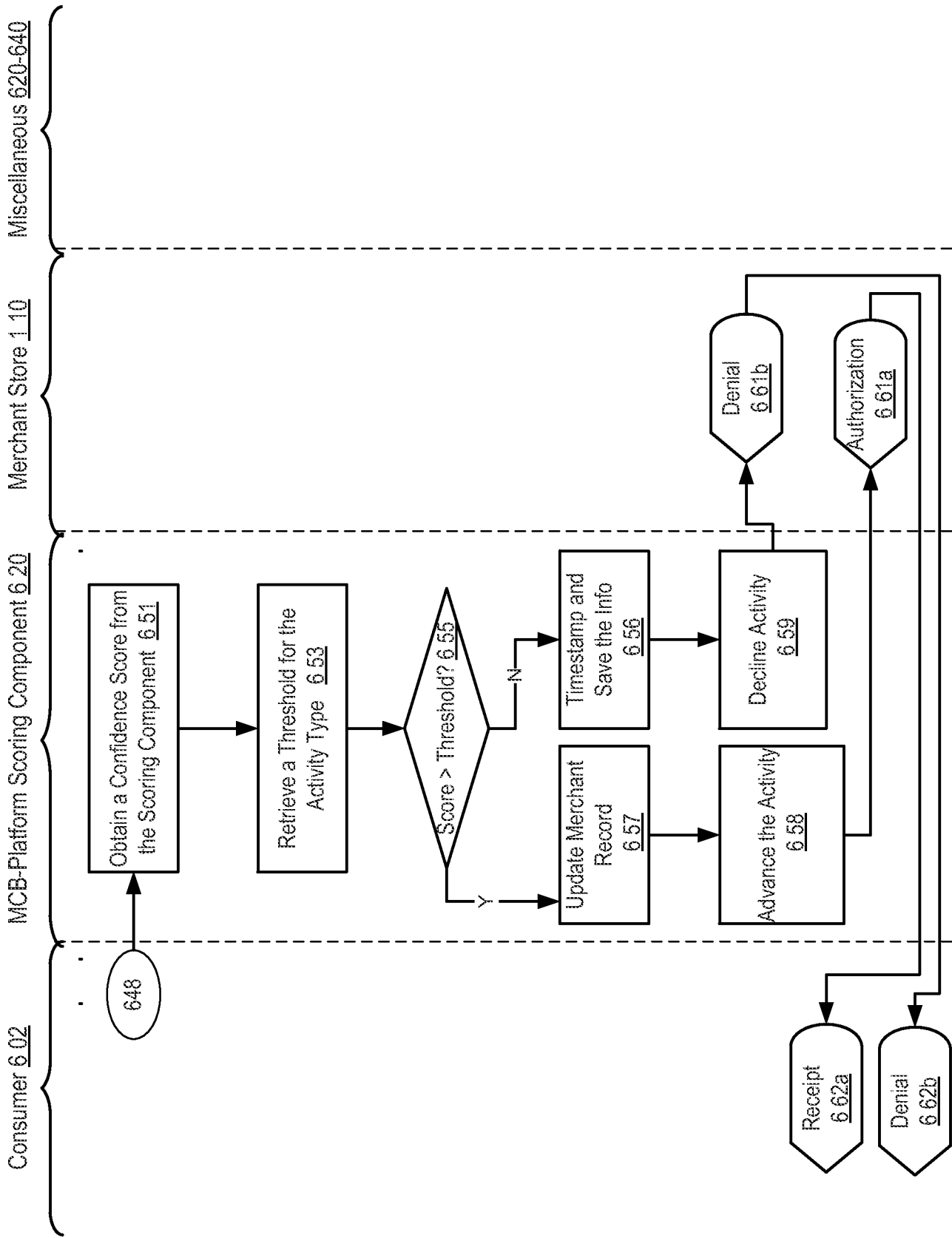


Figure 6F
Example Logic Flow: Merchant Database Update

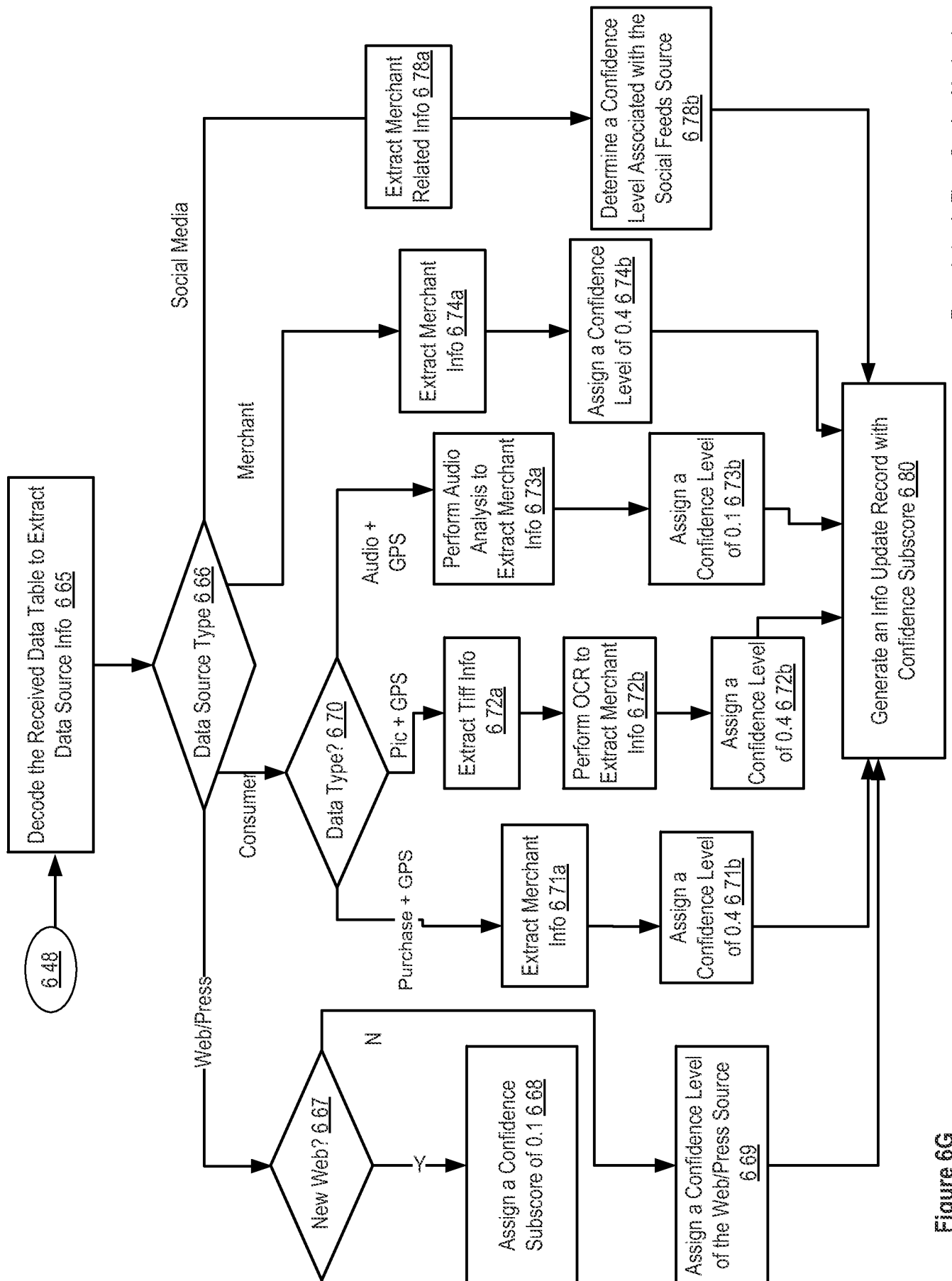


Figure 6G

Example Logic Flow: Scoring Mechanism

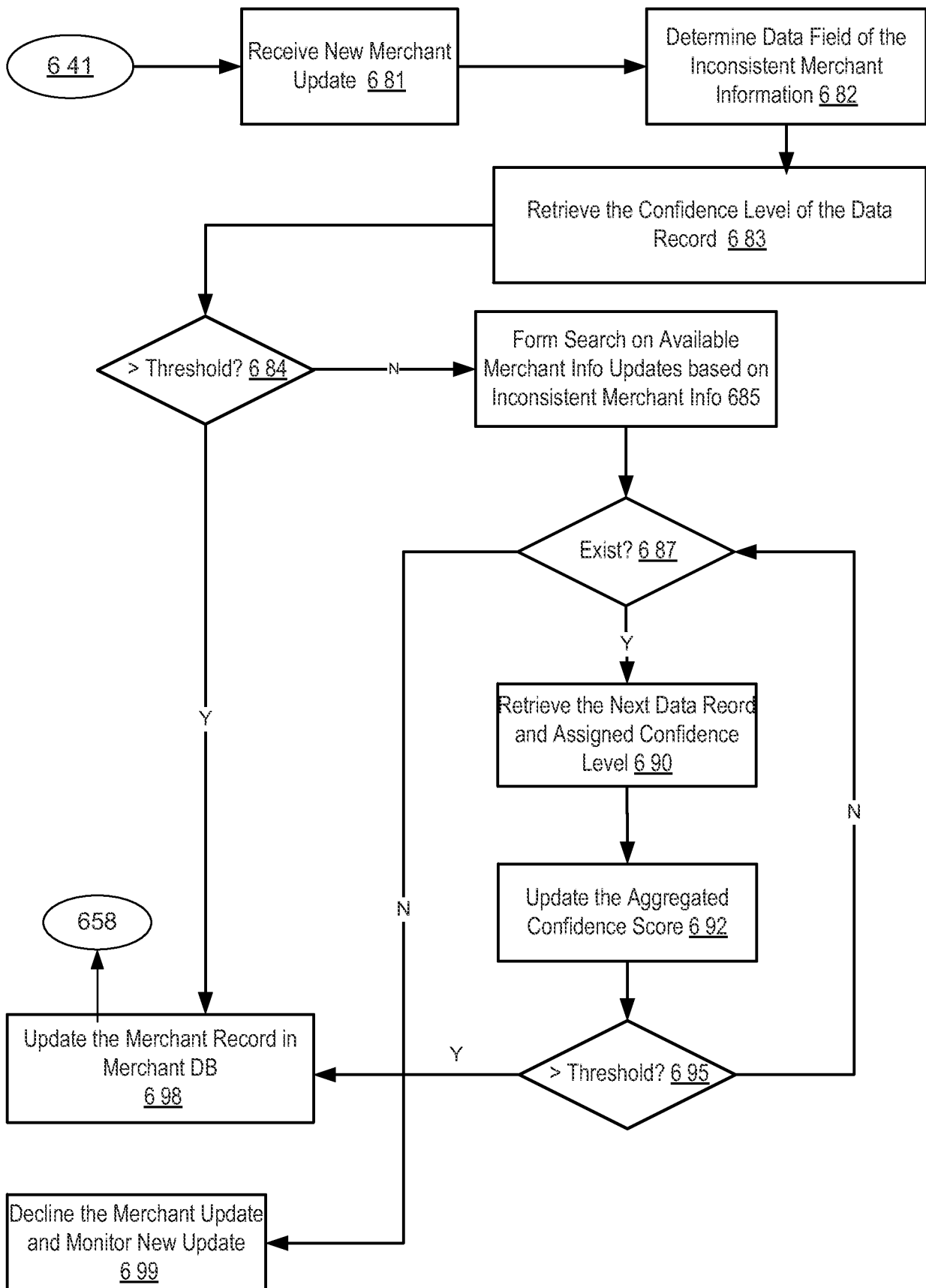


Figure 6H

Example Logic Flow: Progressive Scoring

[Find a TM | Contact Us | Site map](#)
[Sign Up](#)

[Payments & Transfers](#)
[Products & Services](#)
[Customer Service](#)

[1 Rewards > 2 Cardholders > 3 Benefits > 4 Terms & Conditions](#)

Create Your Own Card

Select Rewards 7.03

☐ **Airline Miles** 7.05
 Earn airline miles with each purchase.
Apply today and get 10,000 free miles

☐ **Cash Back** 7.06
 Get money back every time you use your card.
Apply today and get 1000 free points

☐ **Charitable Contributions** 7.07
 Contribute to your favorite cause with every purchase.
Apply today and get 1000 free points

☐ **Retail Rewards** 7.08
 Earn points to spend at your favorite retailers.
Apply today and get 10% off your first purchase

[View Details](#)

Your Account

Credit Limit

Total Credit Limit	\$24,000
Credit Use	\$0

APR as of 06/01/2010

Purchase APR	15.94%
Cash APR	20.74%

Cash Advance

Cash Advance Limit	\$7,000
Cash Advance Used	\$0

Balance Transfer

Yes

Rewards

Figure 7A

Create Your Own Card		1 Rewards > 2 Cardholders > 3 Benefits > 4 Terms & Conditions	
2 Set Up Cardholders	7 10		

Chris M Bennett Primary Cardholder Credit Account # 1-800-7...		Plastic 7 10a The standard, wallet-sized card. <i>Apply today and receive 10% off your first purchase.</i>		Mini Card 7 10b A smaller card that you can attach to your key chain. <i>Apply today and receive 10% off your first purchase.</i>		Visa payWave Micro Tag 7 10c Use your keychain to buy products with a wave. <i>Apply today and receive 10% off your first purchase.</i>		Mobile Payments 7 10d Make purchases and bank online with your phone. <i>Earn a rebate on your next phone purchase.</i>	
View Details		View Details		View Details		View Details			

Your Account Credit Limit Total Credit Limit \$24,000 Credit Line \$0		APR as of 06/01/2008 Purchasing APR 19.24% Cash APR 26.24%		Cash Advance Cash Advance Limit \$7,000 Cash Advance Limit \$0		Balance Transfer Yes		Rewards ☒ Cash Back	
--	--	---	--	---	--	--------------------------------	--	--	--

Payments & Transfers	Products & Services	Customer Service
--	---	----------------------------------

Fast ATM | Contact Us | Site map

2025

Example UI: Consumer Registration 7 00b

[Home](#)
[Payments & Transfers](#)
[Products & Services](#)
[Customer Service](#)

[Find a Tel | Contact Us | Site map](#)

[1 Rewards > 2 Cardholders > 3 Benefits > 4 Terms & Conditions](#)

Create Your Own Card

2 Choose a Card Image

Select an image for your card below

Chris M Bennett
Primary Cardholder
Credit Account (-JABM7)

Upload Image

Back

Save & Continue

Your Account

Credit Limit

Total Credit Limit	\$25,000
Credit Use	\$0

APR as of 06/01/2008

Purchase APR	16.24%
Cash APR	20.34%

Cash Advance

Cash Advance Limit	\$7,500
Cash Advance Used	\$0

Balance Transfer

No

Rewards

☒ Cash Back

Figure 7C

Find ATM | Contact Us | Site Map

Search

Account Overview

Payments & Transfers

Products & Services

Customer Service

1 Rewards > 2 Cardholders > 3 Benefits > 4 Terms & Conditions

Create Your Own Card

2 Card Setup Complete 716



Chris M Bennett card setup is complete.

Continue to select your Benefits

Primary Cardholder

Credit Account (144667)



Would you like to add additional cardholders to this account?

Yes

No

Add another cardholder

Continue to step 3, selecting Benefits

716

Back

Save & Continue

Your Account

Credit Limit

Total Credit Limit \$24,000

Credit Use \$0

APR as of 03/01/2018

Purchase APR 18.24%

Cash APR 20.74%

Cash Advance

Cash Advance Limit \$7,300

Cash Advance Used \$0

Balance Transfer

Yes

Rewards

Cash Back

Figure 7D

Find ATM | Contact Us | Site map

Log In

Payments & Transfers

Products & Services

Customer Service

1 Rewards > 2 Cardholders > 3 Benefits > 4 Terms & Conditions

Create Your Own Card

3 Choose Benefits

7.20

Questions? I can help...

Special Packages

Add-Ons & Options

Core Benefits

These benefits come with every card.

Benefits

☒
Emergency Card Replacement & Cash Disbursement

7.20a

☒
Lost/Stolen Card Reporting

7.20b

☒
Zero Liability

7.20c

☐ Call for an emergency cash advance or card replacement.

☐ Pieces of mind in case your card is ever stolen.

☐ Shop worry-free at millions of merchants.

Annual Fee

Dollar Value

\$3

\$3

\$100

\$40

Annual Fee Total \$ 3

Save & Continue

Your Account

Credit Limit

Total Credit Limit \$24,000

Credit Use \$0

APR as of 03/01/2013

Purchase APR 15.24%

Cash APR 20.94%

Cash Advance

Cash Advance Limit \$7,000

Cash Advance Used \$0

Balance Transfer

Yes

Rewards

Active Miles

Benefits

Total annual Fee \$ 3

Emergency Card Replacement

Lost/Stolen Card Reporting

Zero Liability

Figure 7E

Find ATM Contact Us Site map		Search	
Create Your Own Card		1 Rewards > 2 Cardholders > 3 Benefits > 4 Terms & Conditions	
3 Choose Benefits	Questions? I can help...		
Core Benefits	Add-Ons & Options		
Special Packages Select one of the following benefit packages.			
☒ Travel Package 7 22a Enhance your travel experience with benefits that include: • Auto Rental Collision Damage Waiver • Companion Airline Ticket • Lost Luggage Reimbursement • Travel Accident Insurance			
☐ Retail Package 7 22b Increase your purchasing power with benefits that include: • Purchase Security • Purchase Extended Protection • Price Protection • Return Protection			
Annual Fee Total \$ 3			
Save & Continue			
Payments & Transfers		Products & Services	
Customer Service			
Your Account			
Credit Limit	Total Credit Limit	\$24,000	
Credit Use	Credit Use	\$0	
APR as of 06/01/2008	Purchase APR	18.26%	
	Cash APR	29.74%	
Cash Advance	Cash Advance Limit	\$7,300	
	Cash Advance Used	\$0	
Balance Transfer	Yes		
Rewards	Airline Miles		
Benefits	Total annual Fee \$ 3		
	Emergency Card Replacement		
	Lost/Stolen Card Reporting		
	Zero Liability		
	Auto Rental Collision Damage W		
	Companion Airline Ticket		
	Lost Luggage Reimbursement		
	Travel Accident Insurance		

72697

Home	Find A Tar Contact Us Site Map	Payments & Transfers	Products & Services	Customer Service
----------------------	--	--	---	----------------------------------

1 Rewards > 2 Cardholders > 3 Benefits > 4 Terms & Conditions

Create Your Own Card

3 Choose Benefits

[Questions?](#)
I can help...

Travel Package
Protect yourself and enhance your travel experience with these features:
Benefits

- ☒ **Auto Rental Collision Damage Waiver**
Receive collision/theft damage insurance coverage when you use your card to pay for the full cost of a car rental.
- ☒ **Companion Airline Ticket**
Receive a complimentary companion ticket during your first year of card membership.
- ☒ **Lost Luggage Reimbursement**
Receive a reimbursement for lost luggage when you use your card to pay for your airline tickets.
☒ \$250 per Claim
☐ \$1000 Per Claim

Cardholder Summary
View details about your credit limit, APR, cash advance rates, and more.

Credit Limit
Total Credit Limit: \$24,000
Credit Line: \$0

APR as of 06/06/2008
Purchase APR: 18.24%
Cash APR: 28.74%

Cash Advance
Cash Advance Limit: \$7,500
Cash Advance Used: \$0

Balance Transfer
Yes

Rewards
Airline Miles

Benefits
Total annual Fee: \$ 3
Emergency Card Replacement ?
Lost/Stolen Card Reporting
Zero Liability
Auto Rental Collision Damage W
Companion Airline Ticket
Lost Luggage Reimbursement
Travel Accident Insurance

Annual Fee Total \$ 3

2019

Create Your Own Card		1 Rewards > 2 Cardholders > 3 Benefits > 4 Terms & Conditions	
3 Choose Benefits	Questions? I can help...		
Core Benefits	Add-On Options		
Add-On & Options Choose additional benefits targeted to your individual needs.			
Benefit	Annual Fee	Dollar Value	
☒ Identity Protection Service ☐ Protection from unauthorized use of your card or account information.	\$2	\$29	
☒ ID Theft Restoration ☐ Restores your ID in case it's stolen.	\$2	\$2	
☒ Year-End Summary Statement ☐ Receive a detailed annual report of your card expenditures.	\$0	\$0	
☒ Concierge Services ☐ Receive 24/7 access to concierge services.	\$0	\$0	
☒ Apply rewards point to offset annual fee ☐ Use your reward points to pay benefit fees.	\$0	\$0	
		Annual Fee Total \$ 5	
Save & Continue			

Your Account	Credit Limit
Total Credit Limit	\$24,000
Credit Used	\$0
APR as of 03/01/2008	
Purchase APR	13.24%
Cash APR	20.74%
Cash Advance	
Cash Advance Limit	\$7,500
Cash Advance Used	\$0
Balance Transfer	
Yes	
Rewards	
Airline Miles	
Benefits	Total Annual Fee: \$ 5
☒ Emergency Card Replacement	
☒ Lost/Stolen Card Replacing	
☒ Zero Liability	
☒ Auto Rental Collision Damage Wa	
☒ Companion Airfare Ticket	
☒ Lost Luggage Reimbursement	
☒ Travel Accident Insurance	
☒ Monthly Protection Service	

750617

Create Your Own Card

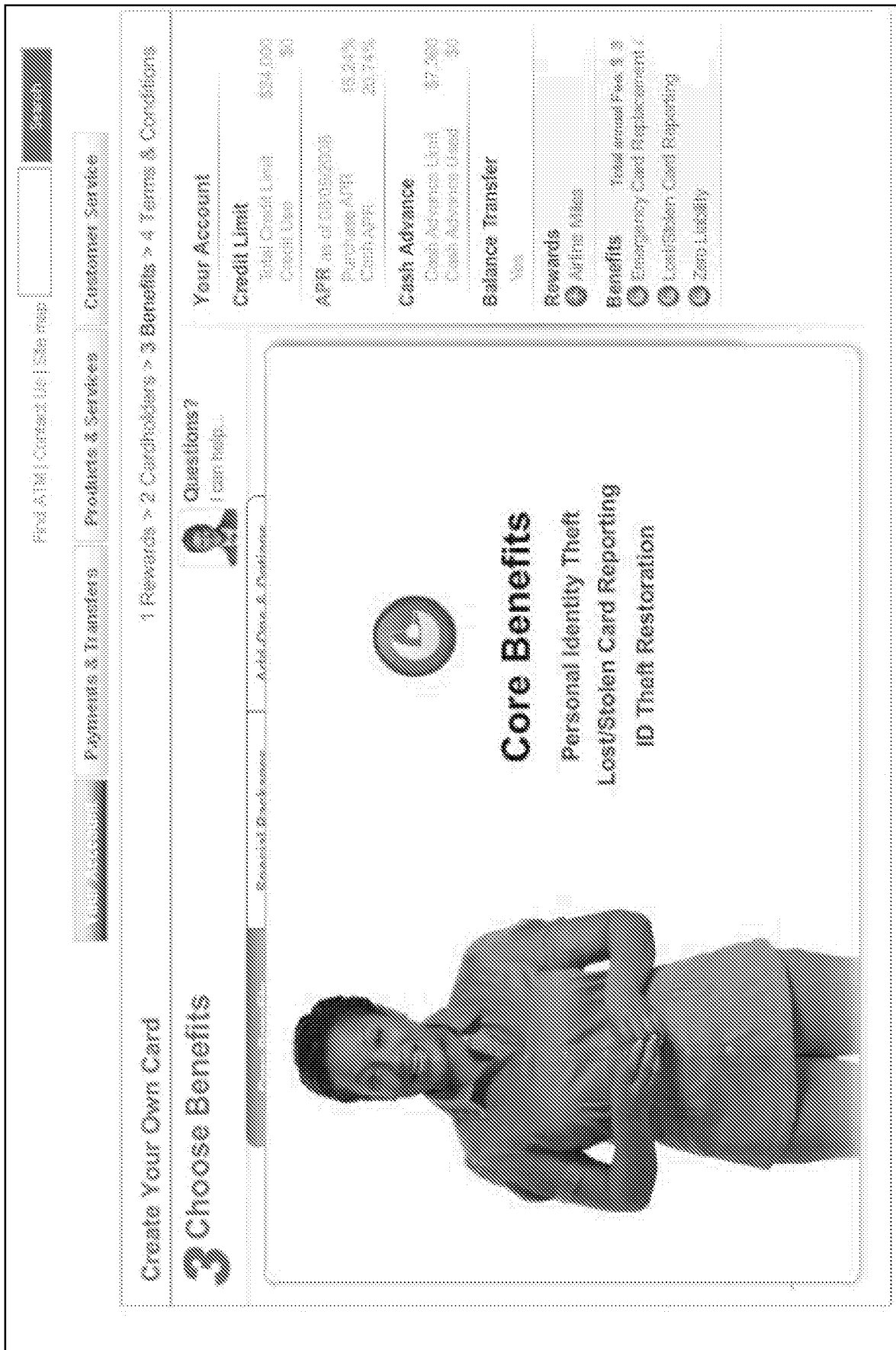
4 Terms & Conditions
For each of your benefits, you can download and review the terms and conditions.

Your Benefits	
	Emergency Card Replacement & Cash Disbursement View Terms & Conditions
	Lost/Stolen Card Reporting View Terms & Conditions
	Phone Protection View Terms & Conditions
	Purchase Extended Protection View Terms & Conditions
	Purchase Security View Terms & Conditions
	Personal Identity Theft View Terms & Conditions
	ID Theft Restoration View Terms & Conditions

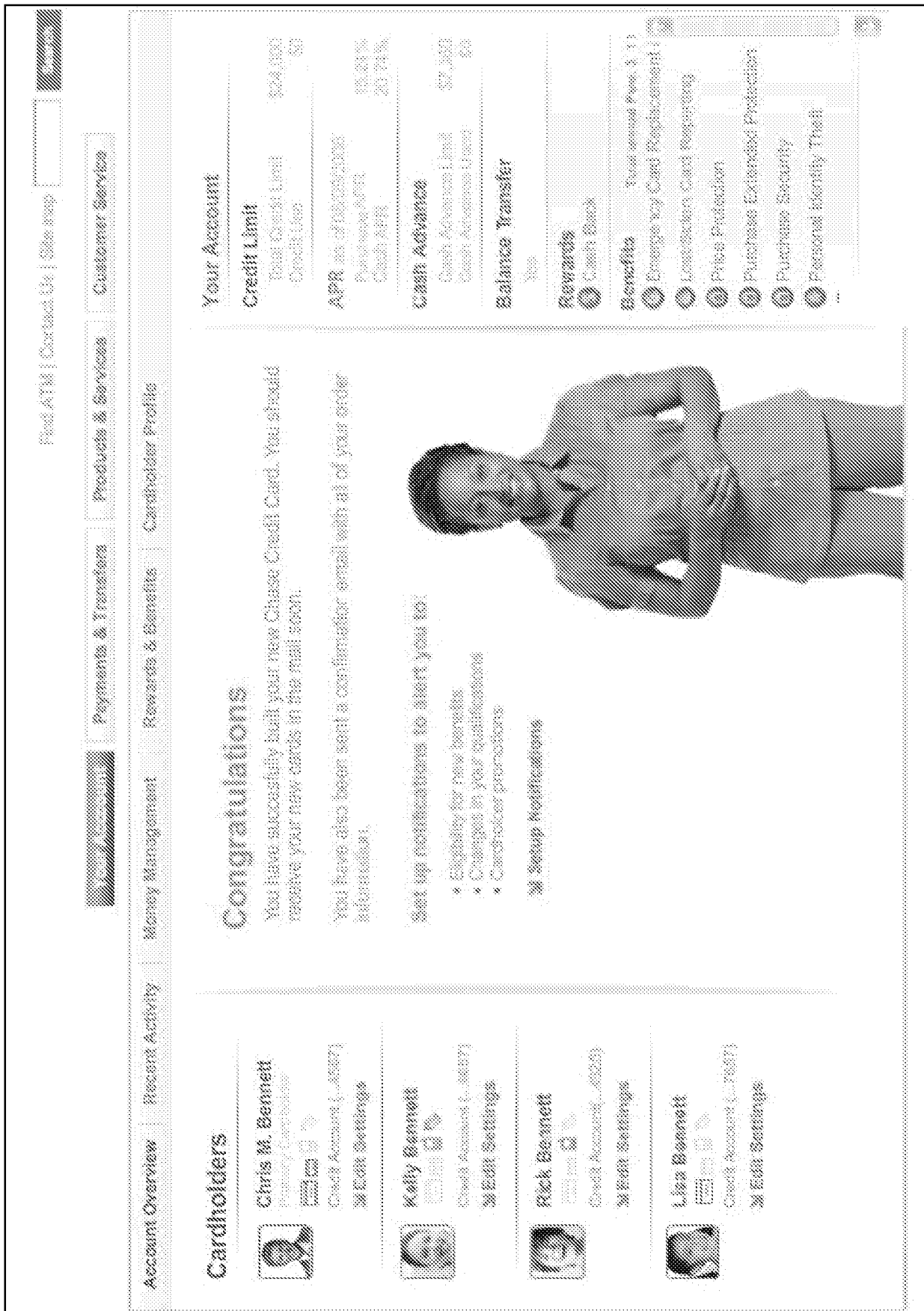
☐ I have downloaded the file above, read and agree to the Terms & Conditions of my Chase account.

[Back](#) [Continue](#)

7000



2006



7596
7597
7598
7599
7600

Find ATM | Contact Us | Site map

Search

Products & Services

Payments & Transfers

Cardholder Profile

Rewards & Benefits

Money Management

Recent Activity

Account Overview

Notifications

See alerted to unauthorized transactions, receive monthly activity reports and sign up for other notifications.

7:30

Select your notifications:

☐	Notify me about my retail benefits when I make retail purchases	☐	☐	☐	☒	Send test message
☐	Notify me about my travel benefits when I purchase travel	☐	☐	☐	☐	Send test message
☐	Alert me when I become eligible for new benefits	☐	☐	☐	☐	Send test message
☐	Send me an annual reminder to review my benefits	☐	☐	☐	☐	Send test message

Notification Accounts

Email	bernet12@gmail.com	☒	Send test message
Text messages (SMS)	415-555-1234	☒	Send test message
Automated Video Message	415-555-1234	☒	Send test message

Done

Figure 7

Online Banking Sign Out
Locations · Help · En Español

Accounts Overview Bill Pay Transfers Investments Customer Service
Account Details My Portfolio Alerts Open an Account

8.10 Visa Signature Card

The Safest, Easiest Way to Pay

Whether you're online or on the go, Visa V is the safest, easiest way to pay anywhere. Activate Visa V with your Visa Signature Card now.

- * Secure and convenient
- * Pay with your phone
- * Pay friends on Facebook
- * Redeem offers online or on the go

Sign Up Now

8.05

Account: Select Account

Balance Summary

\$3,822.12 Available Balance as of today

Account balance history
Tap to view more info

Account Numbers & Fees

Show account number
Personal schedule of fees

Turn off Debit Info

**Go Green.
Use Paperless Statements.
Learn more.**

Date	Description	Debit	Credit
01/14/	CROSSROADS CAFE SAN FRANCISCO CA Amount may change - waiting for final amount from merchant	\$10.10	
01/12/	CHECKCARD APL STUNES 888-732-7793 CA ON 01/10/2011	\$9.79	
01/10/	THE ENGRAMBAM WEST STONE RIDGE NY ON	\$43.50	
01/09/	EASTW-BABY MOUNTAIN VIEW CA	\$34.25	
01/01/	PEASANT PIES SAN FRANCISCO CA	\$12.10	
01/01/	KICKAROO.COM DOLGERSVILLE NY	\$40.05	
12/77/	MET ASTORIA RABAT SAN FRANCISCO	\$80.05	
12/24/	SEE JANE RUN SAN F SAN FRANCISCO	\$140.05	
12/24/	TRADER JOE'S # 01-11 CUSTOMER 1552 PURCHASE #940100 SAN FRANCISCO CA	\$180.15	
12/22/	MET*WWW.METTLIX.COM/CE 01/09 PURCHASE #12902101039422 METTLIX COM, CA	\$10.00	

Customer Service

Manage alerts

Show me more services...

Urgle 8A

Example Ut: Consumer Registration 8 00a

[illegible]

2025

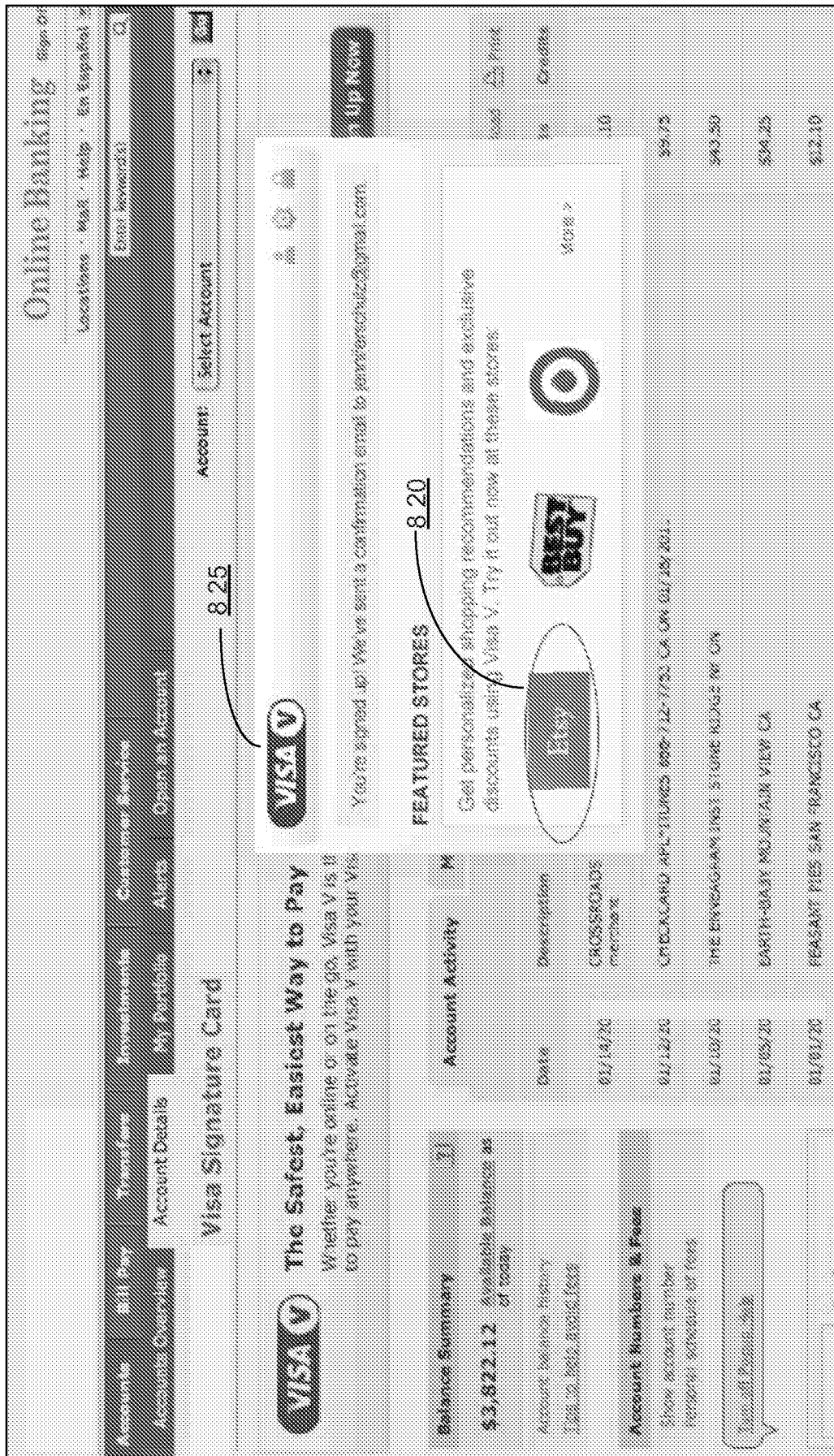


Figure 8C

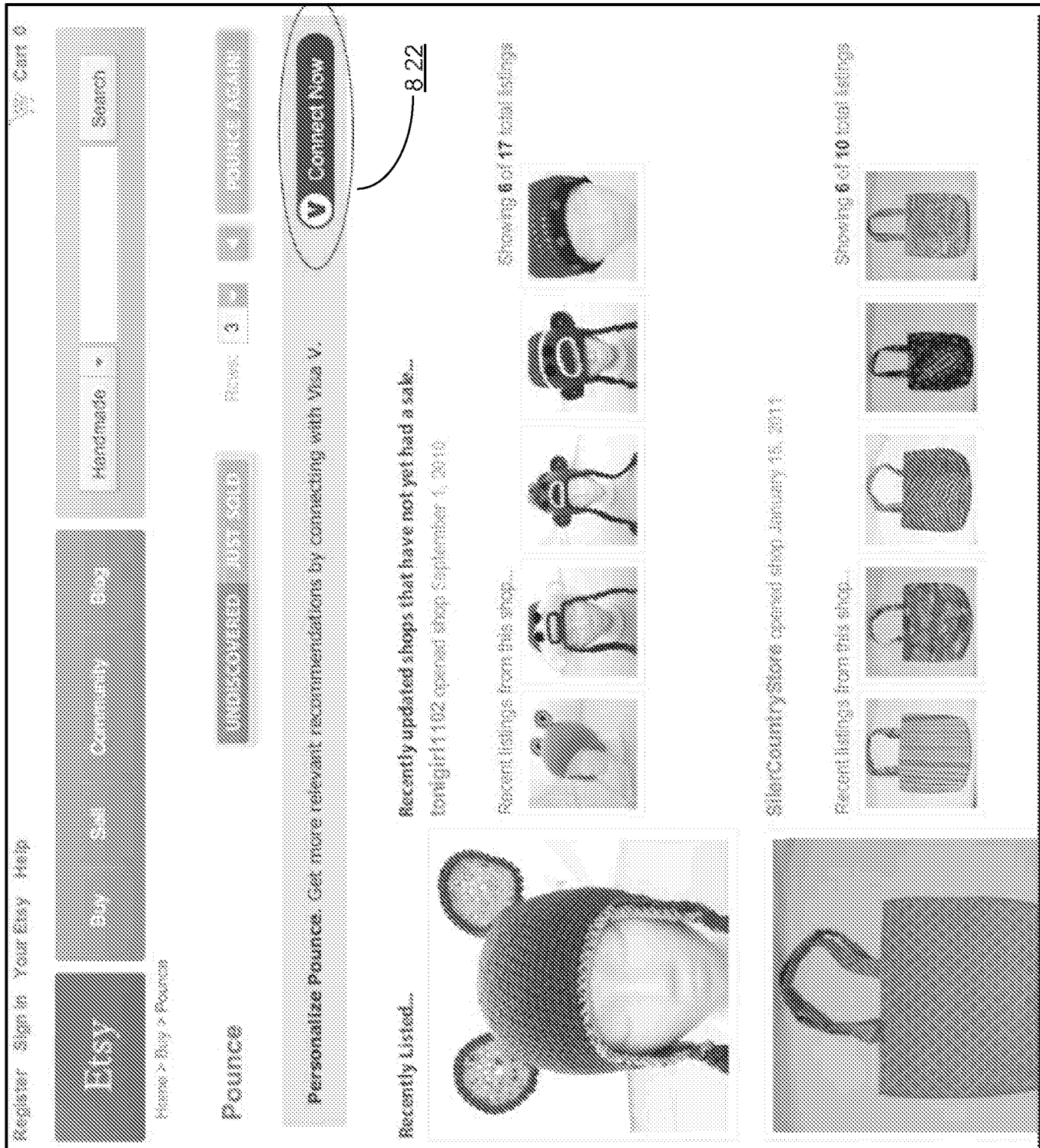


Figure 8D

Example UI: Receiving Offers 8.00d

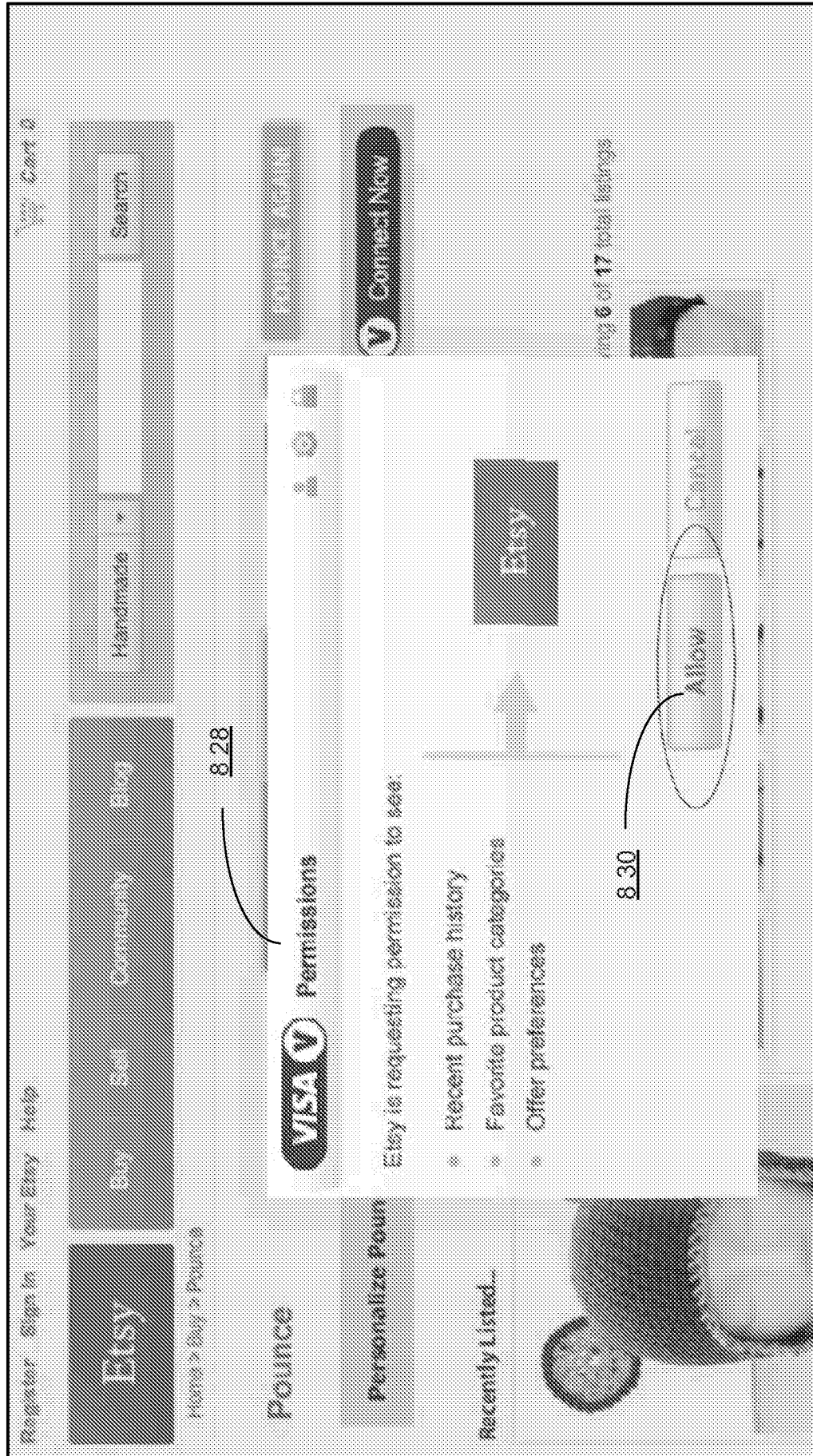


Figure 8E

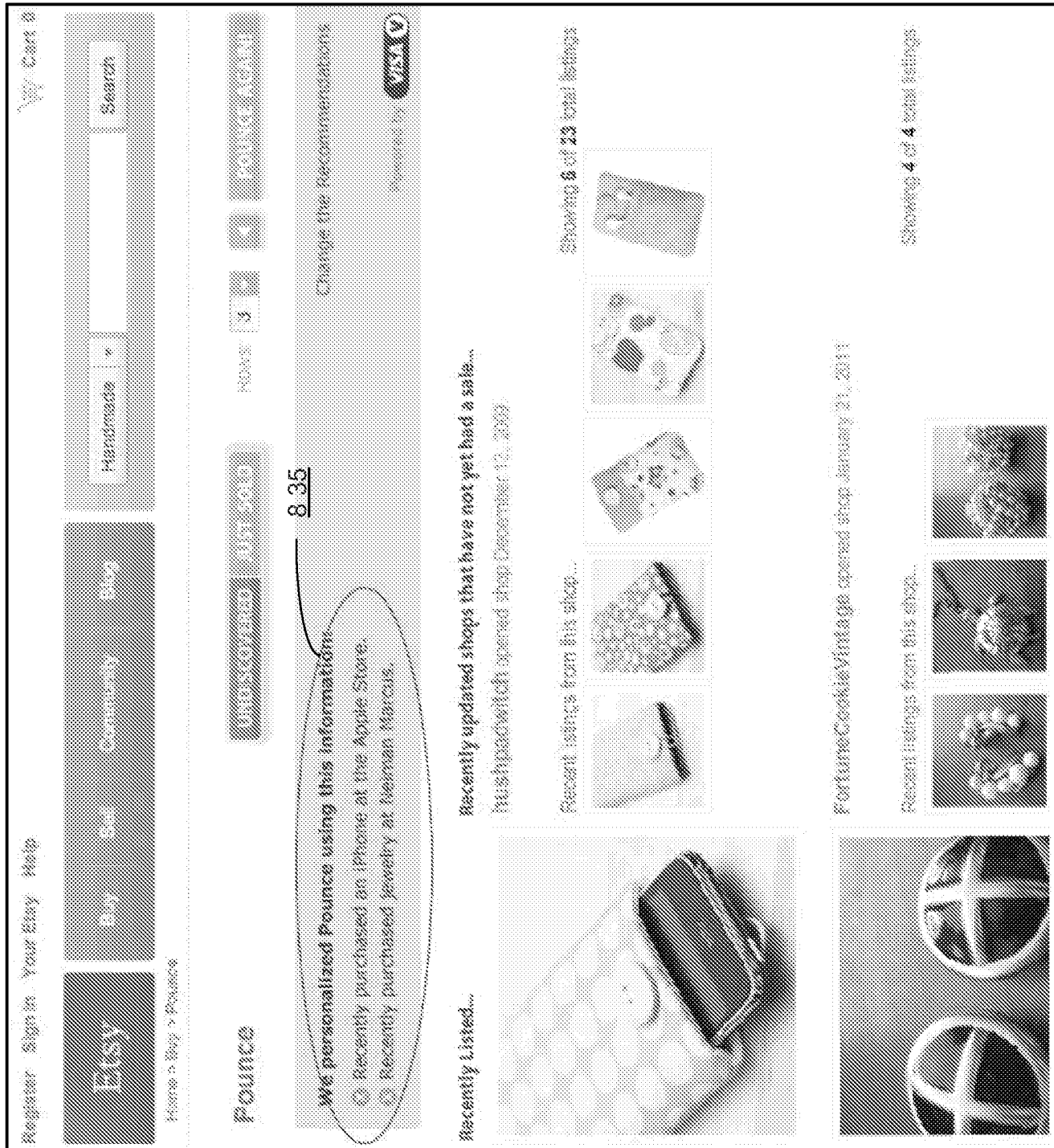


Figure 8F



Figure 8G

Example UI: Offer Redemption 8.00g



Figure 8H

Example UI: Offer Redemption 8 00h

facebook

Search

Home Friends Account

Manage Campaigns

9.05a

1. Design Your Coupon

Design Your Coupon FAQ

Title 21 characters left [?]

Save 10% off kayaking

Body Text 123 characters left [?]

Save 10% off Chippewa rentals when you share with your friends.

Image required [?]

Choose File No file chosen

9.05b

2. Campaigns, Pricing and Scheduling

Ad Campaigns and Pricing FAQ

Campaign & Budget

Campaign Name: January Coupon Campaign

Budget (USD): 50.00 Per day 5 [?]

What is the most you want to spend per day? (max 1,000 USD)

Choose an existing campaign [?]

Schedule

Today at 3:00 pm Pacific Time

at 3:00 pm Pacific Time

☐ Run my campaign continuously starting today

Pricing

☐ Pay for impressions (CPM) [?]

☐ Pay for Clicks (CPC) [?]

☒ Pay for Conversions (CPA) [?]

Max Bid (USD) (How much are you willing to pay per conversion? (min 0.01 USD)) [?]

1.00

Learn More

Figure 9A

Example UI: Offer Bridging via Social Media 9.00a

9.05c

3. Targeting

Ad Targeting FAQ

Location

Country (1)

United States (x)

By State/Province (1)

By City (1)

Demographics

Age (1)

18 - Any

Require exact age match (1)

Sex (1)

All

Men

Women

Likes & Interests

Enter an interest (1)

Connections on Facebook

Connections (1)

Target users who are connected to:

Enter your Page, Event, Group, or App (1)

Target users who are not already connected to:

Enter your Page, Event, Group, or App (1)

Friends of connections:

Target users whose friends are connected to:

Enter your Page, Event, Group, or App (1)

Show Advanced Targeting Options

132,795,100 people

Estimated Reach

Who live in the United States

Age 18 and older

Review

Review Coupon

Figure 9B

Example UI: Offer Bridging via Social Media 9.00b

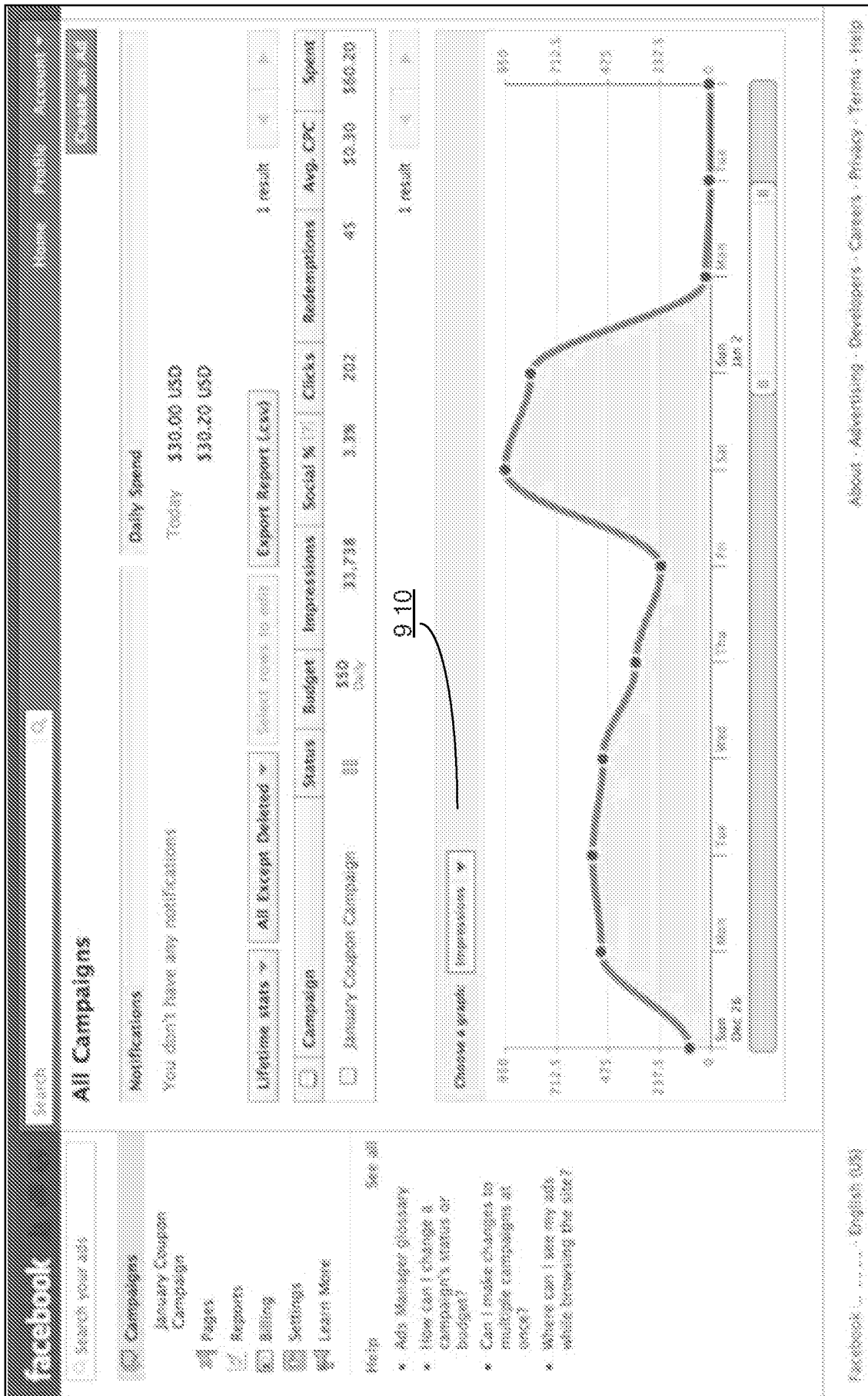


Figure 9C

Example UI: Offer Bridging via Social Media 9 00c

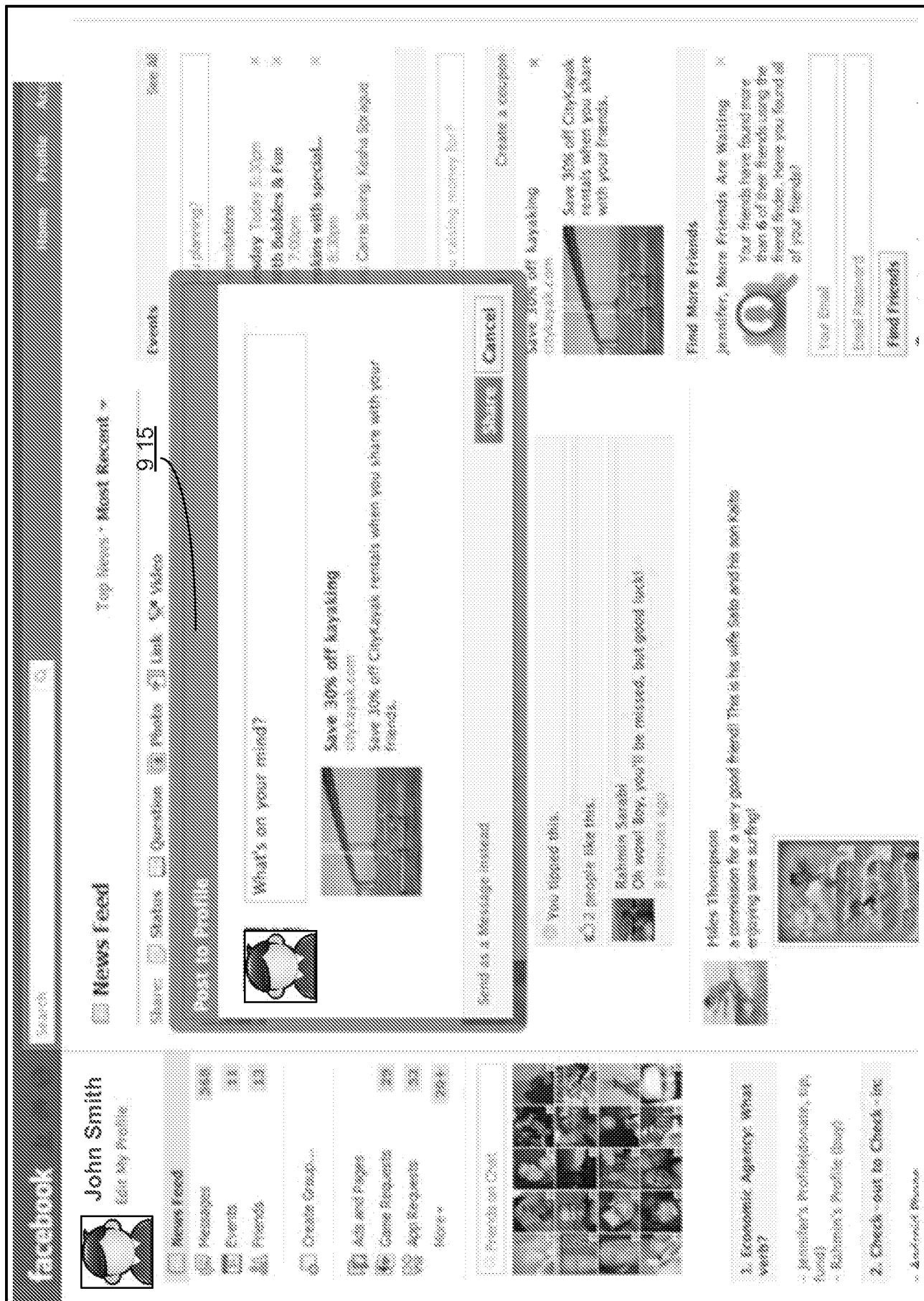


Figure 9D

Example UI: Offer Bridging via Social Media 9 Dd



Example UI: Offer Bridging via Social Media 9 00e

Figure 9E



Figure 9F

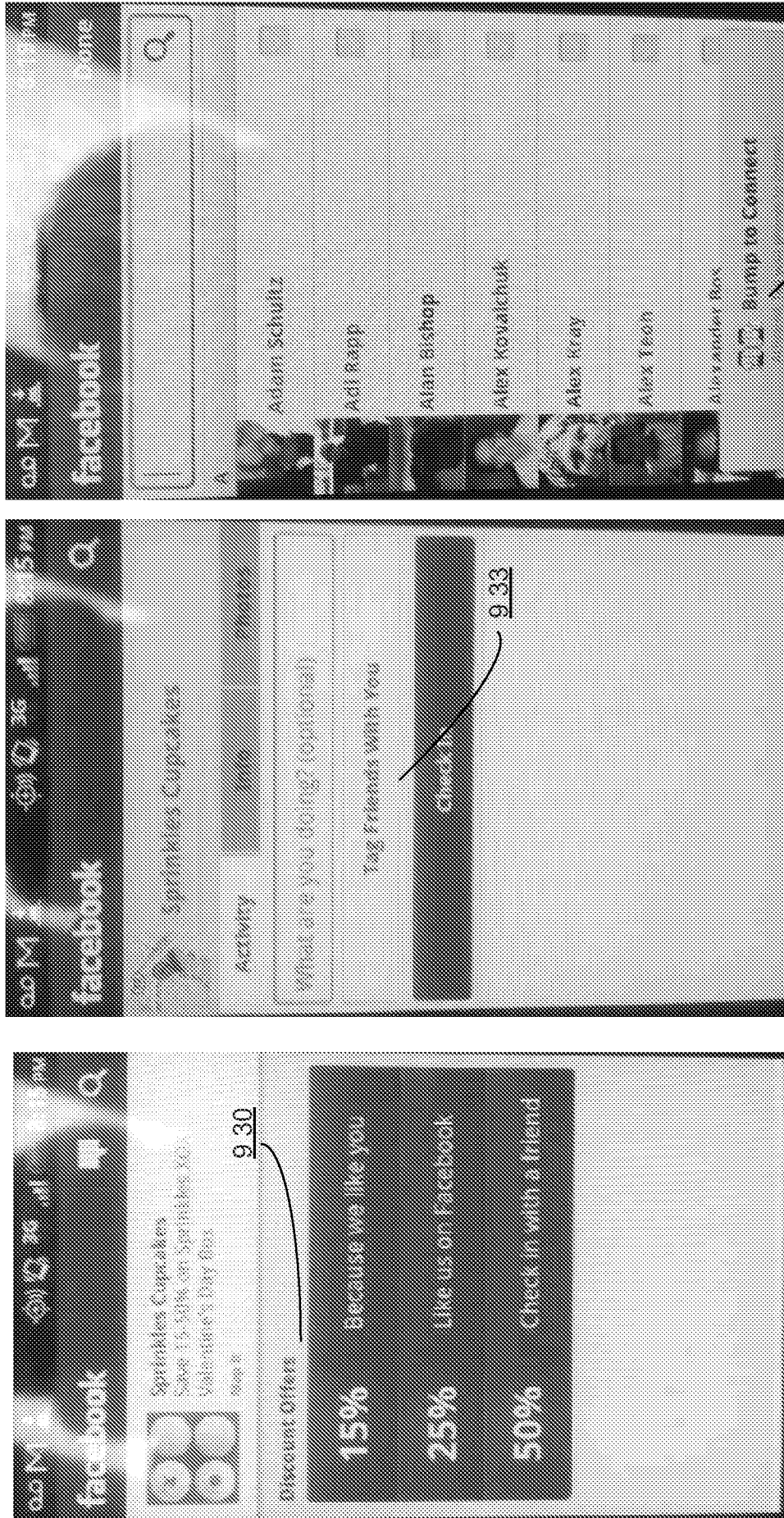


Figure 9C

Example Mobile UI: Offer Bridging via Social Media 9.00g

9 40



9 45

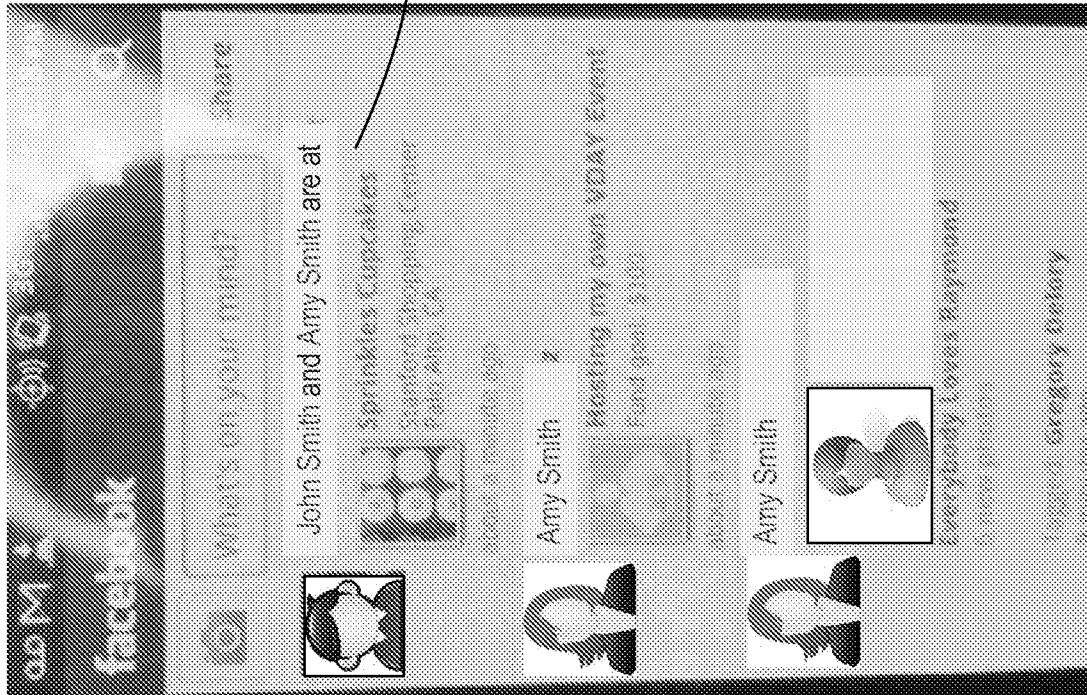
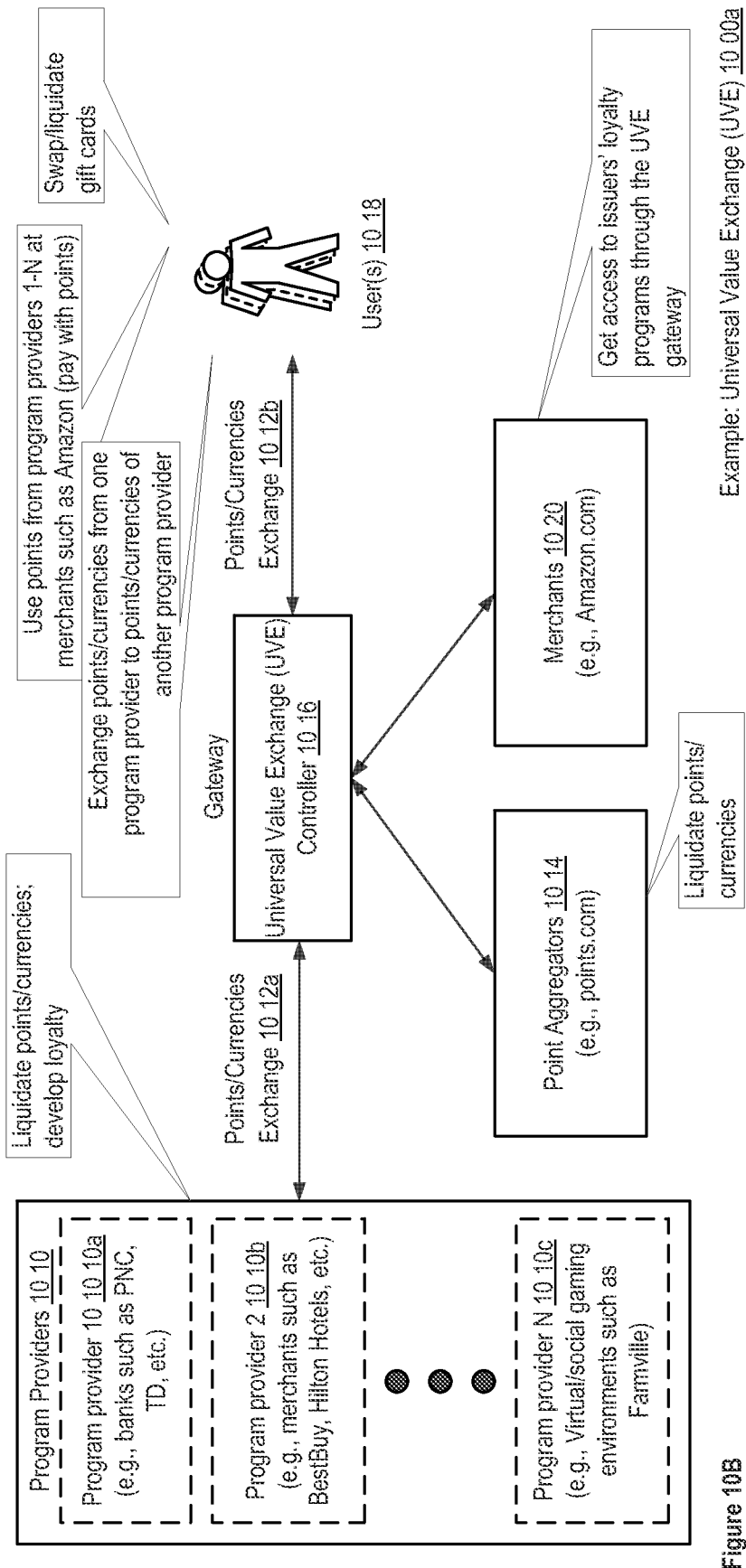
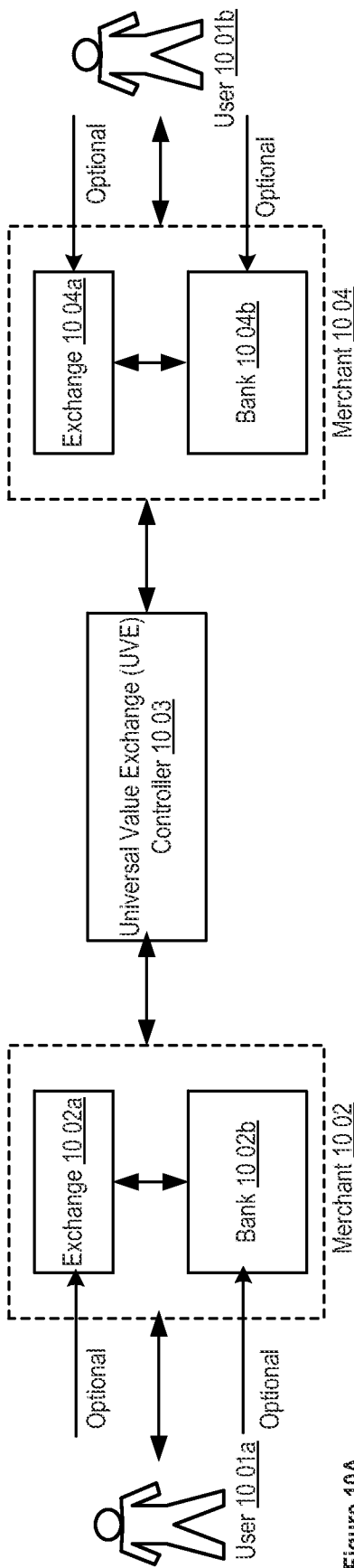


Figure 9H

Example Mobile UI: Offer Bridging via Social Media 9 00h



Example: Universal Value Exchange (UVE) 10 00a

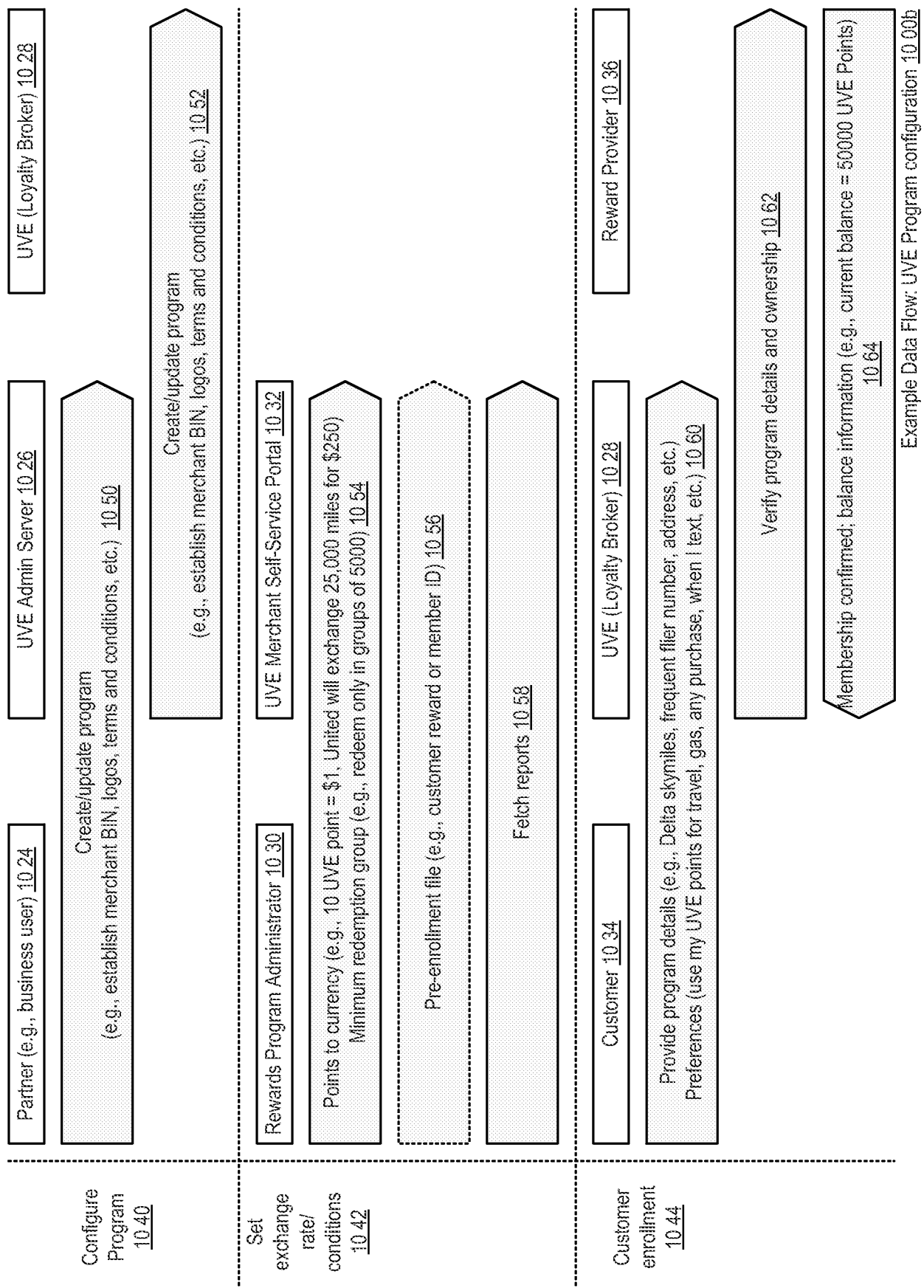


Figure 10C

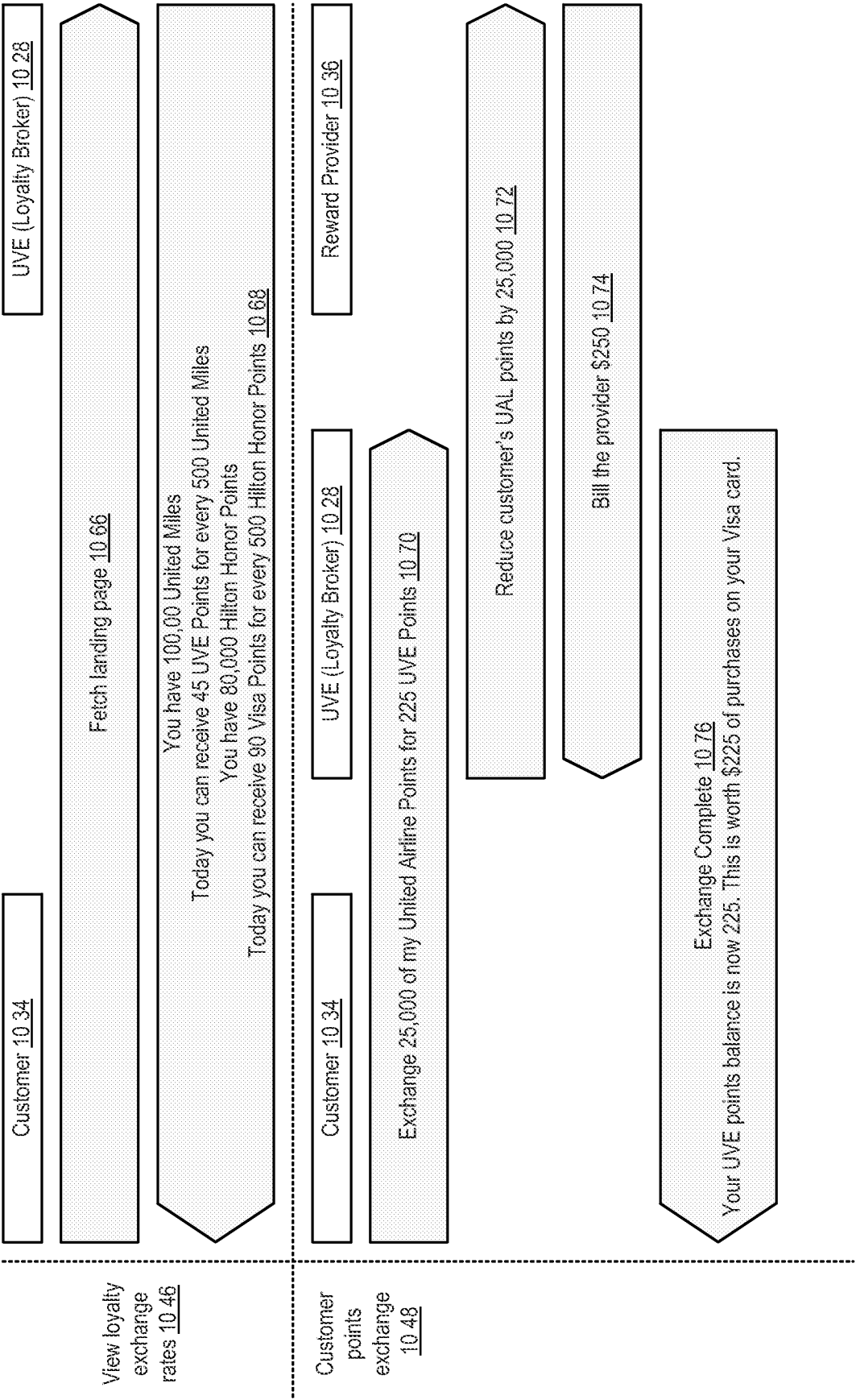


Figure 10D Example Data Flow: UVE Program configuration 10 00b

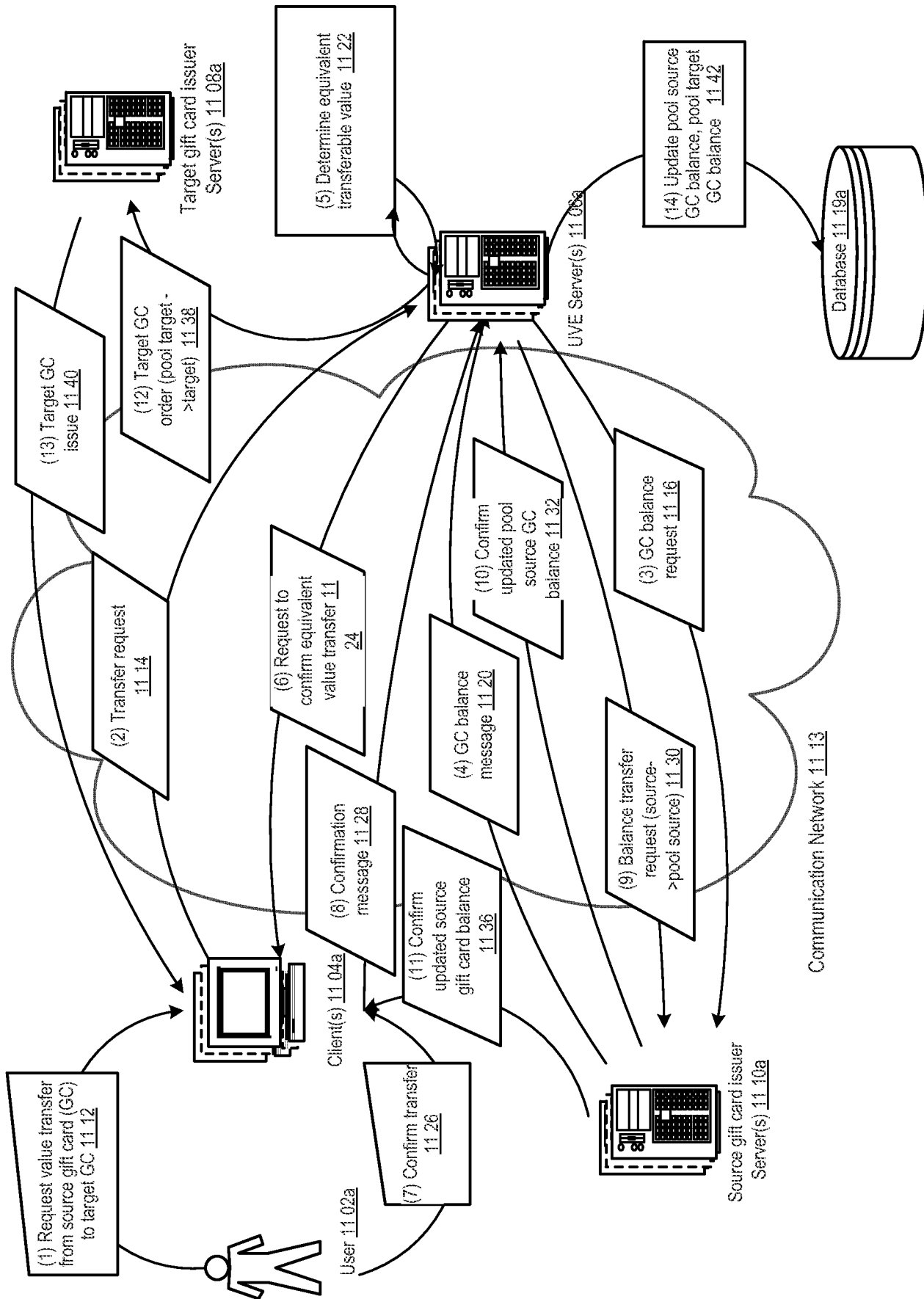


Figure 11A

Example Data Flow: Closed Loop Gift Card Value Exchange/Transfer 11 01a

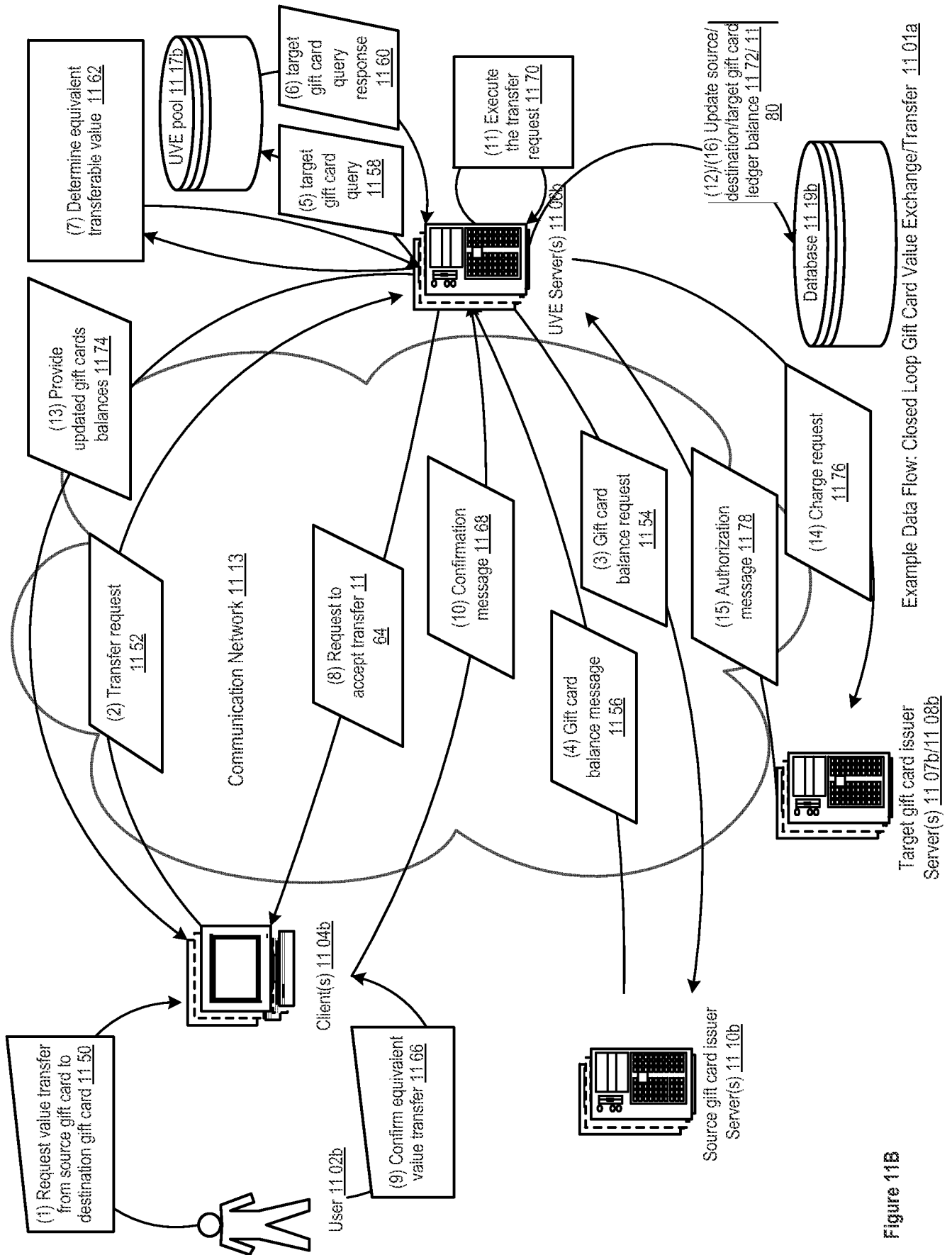


Figure 11B

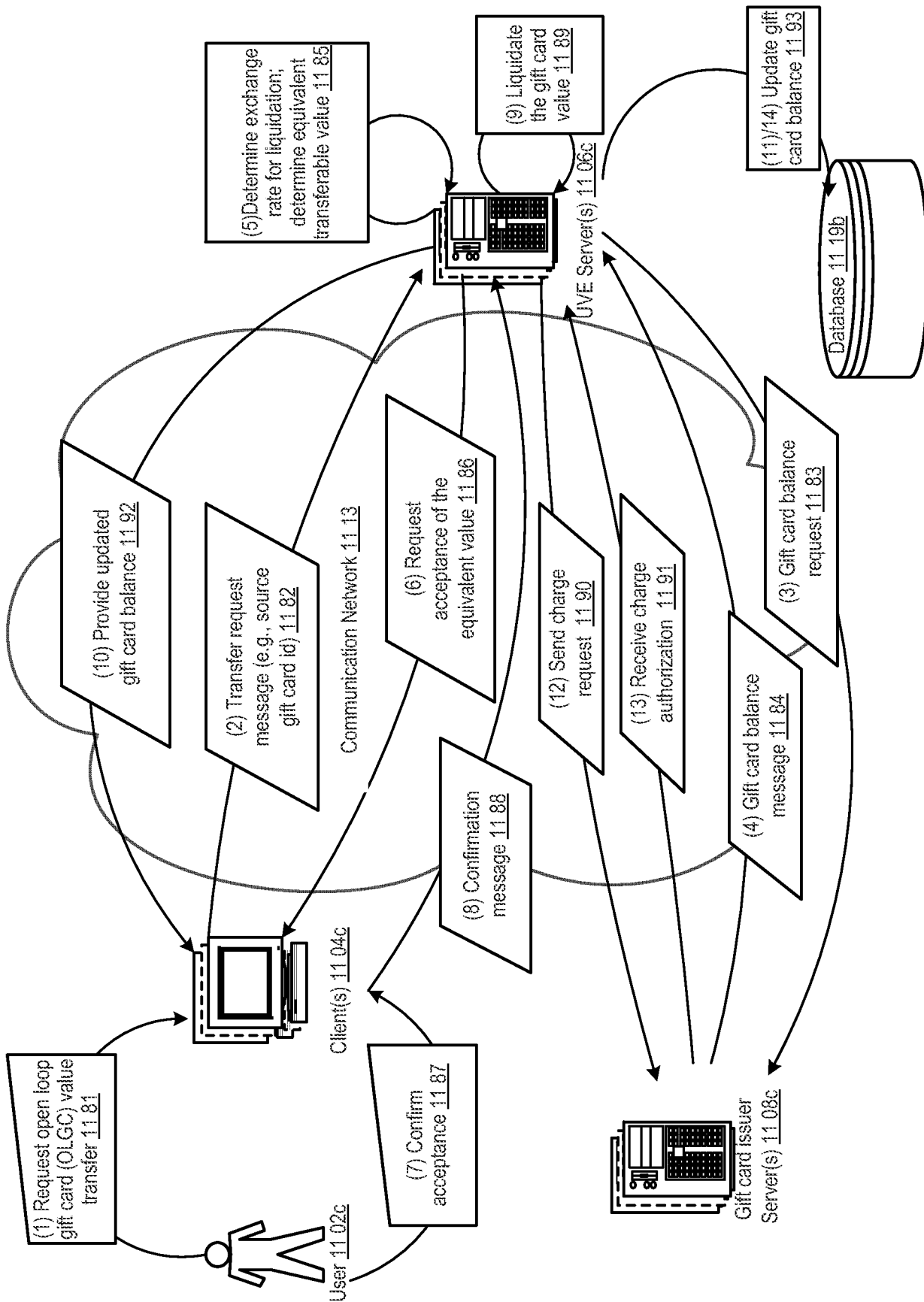
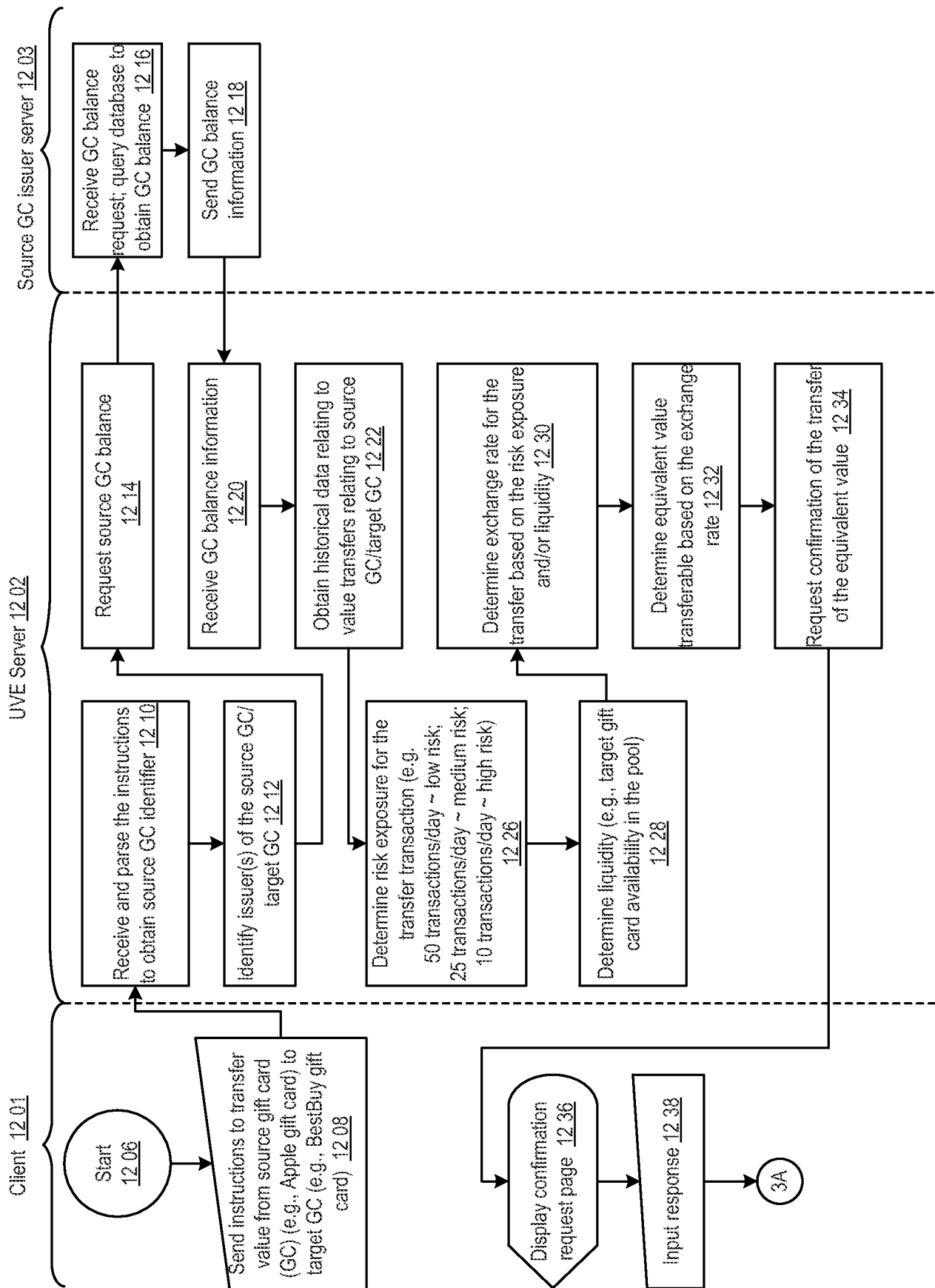


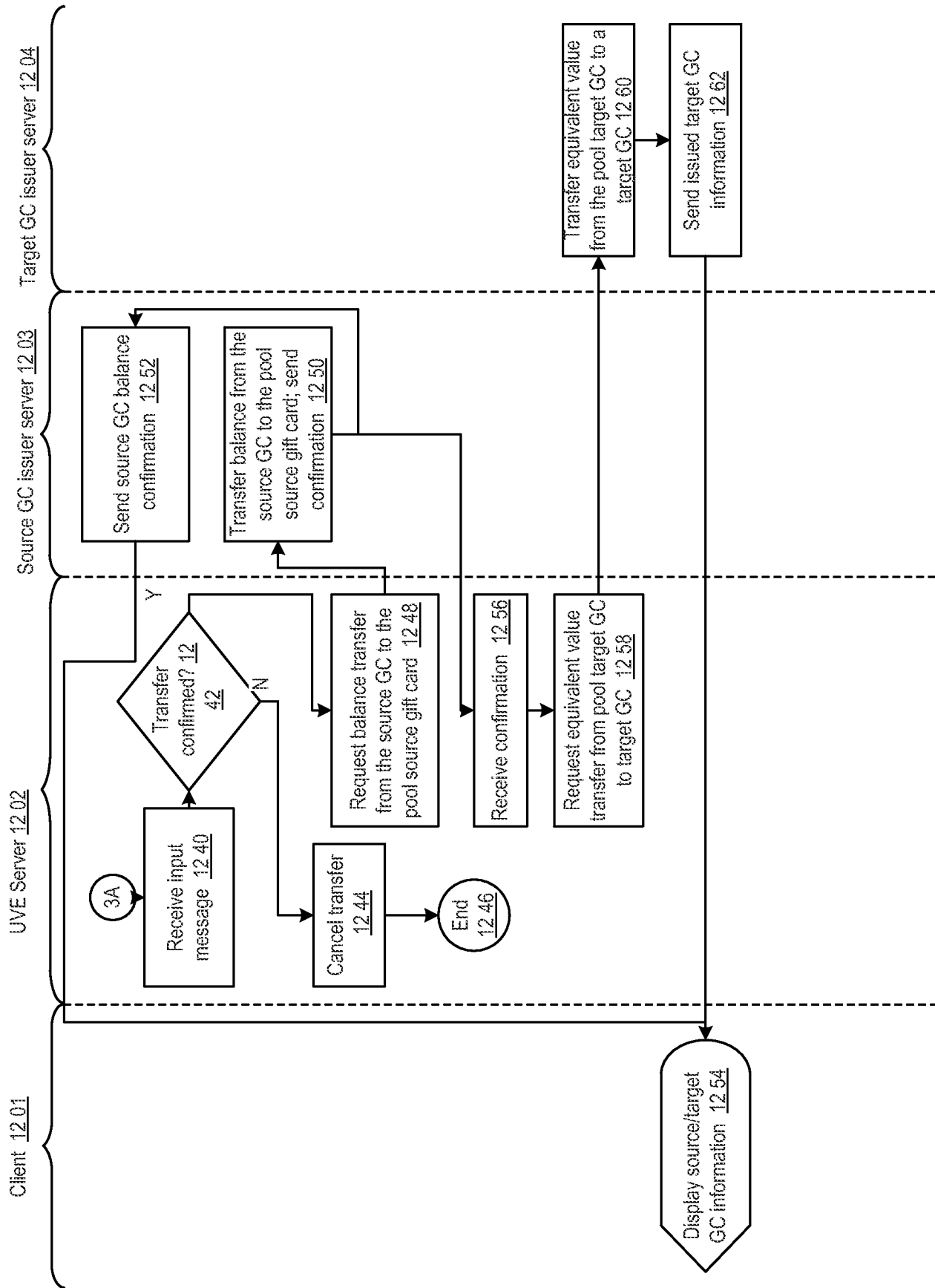
Figure 11C

Example Data Flow: Open Loop Gift Card Value Exchange/Transfer 201c



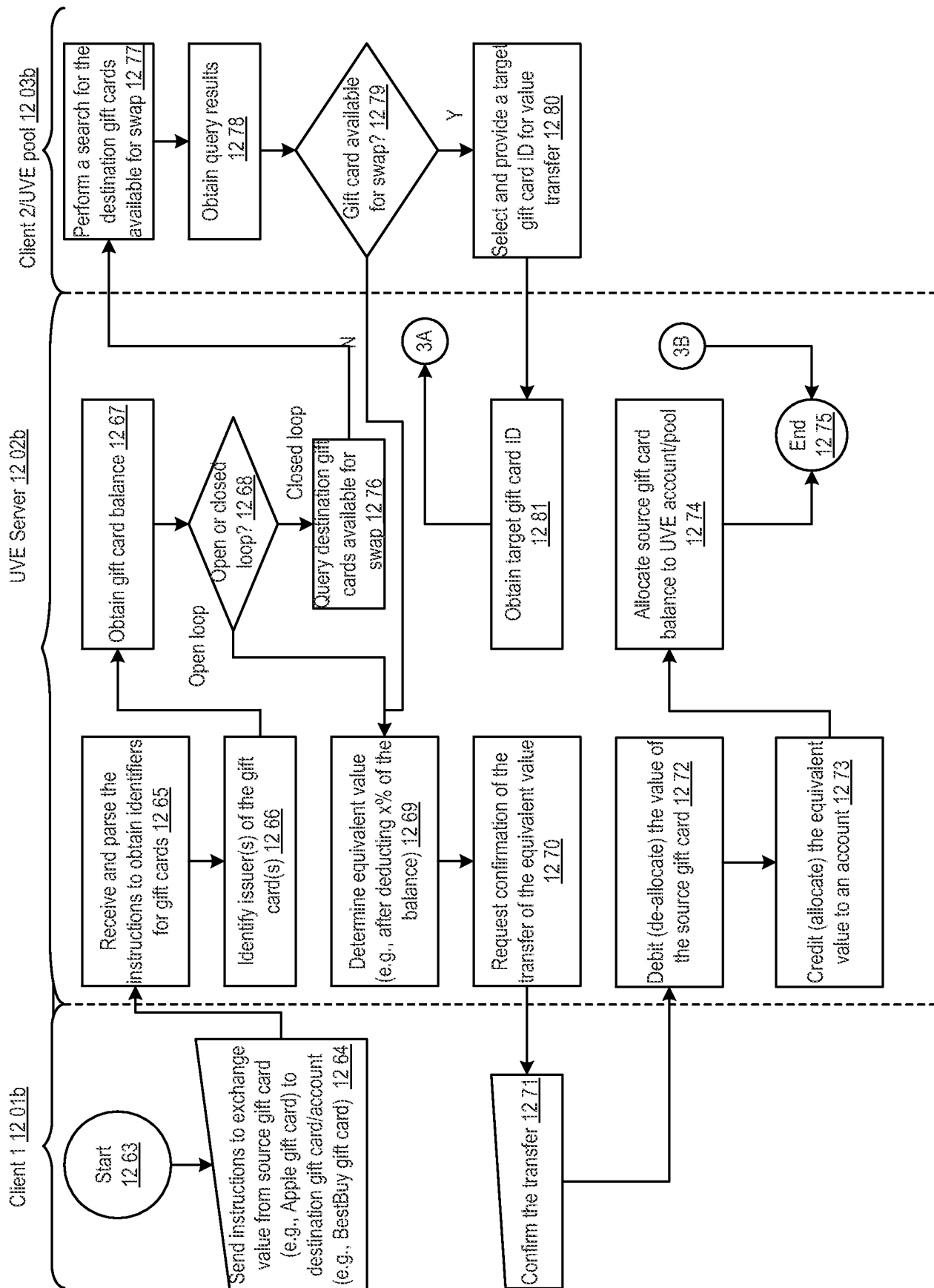
Example Logic Flow: Closed Loop Gift Card Universal Value Exchange (CLGC -UVE) Component 12.00a

Figure 12A



Example Logic Flow: Closed Loop Gift Card Universal Value Exchange (CLGC -UVE) Component 12.00a

Figure 12B



Example Logic Flow: Gift Card Universal Value Exchange (GC-UVE) Component 12.00b

Figure 12C

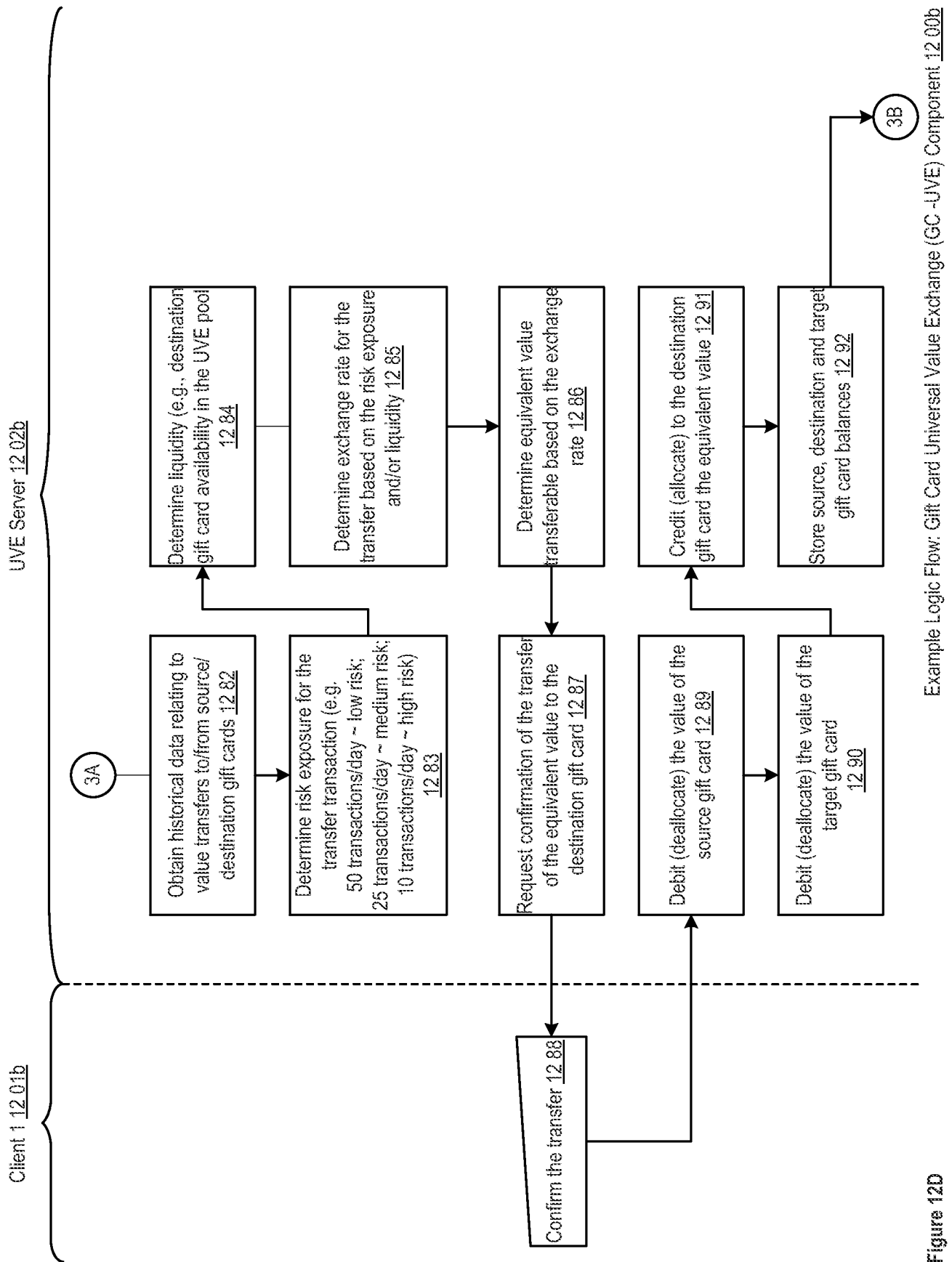
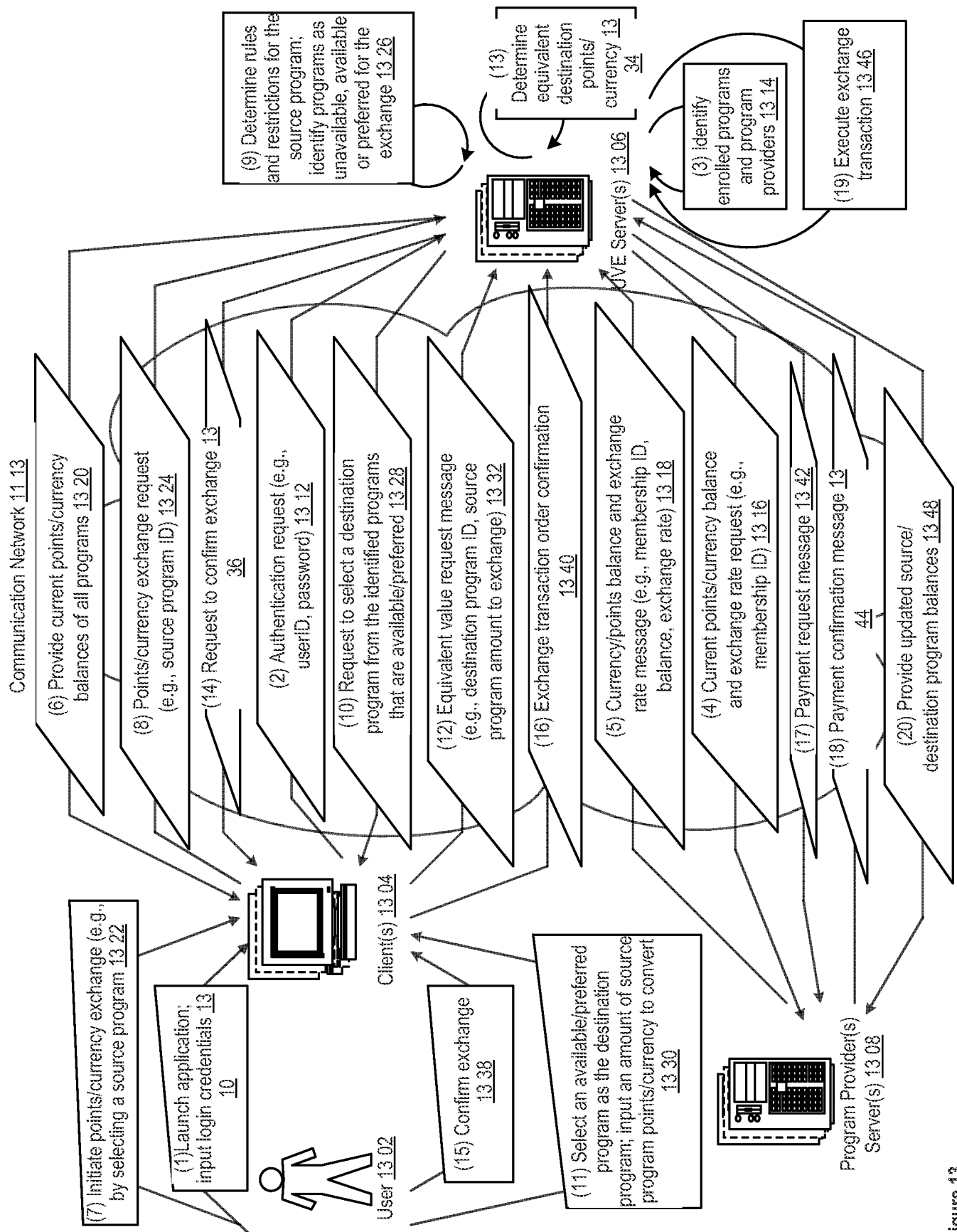


Figure 12D



Example Data Flow: Source/Destination Universal Value Exchange 13.01

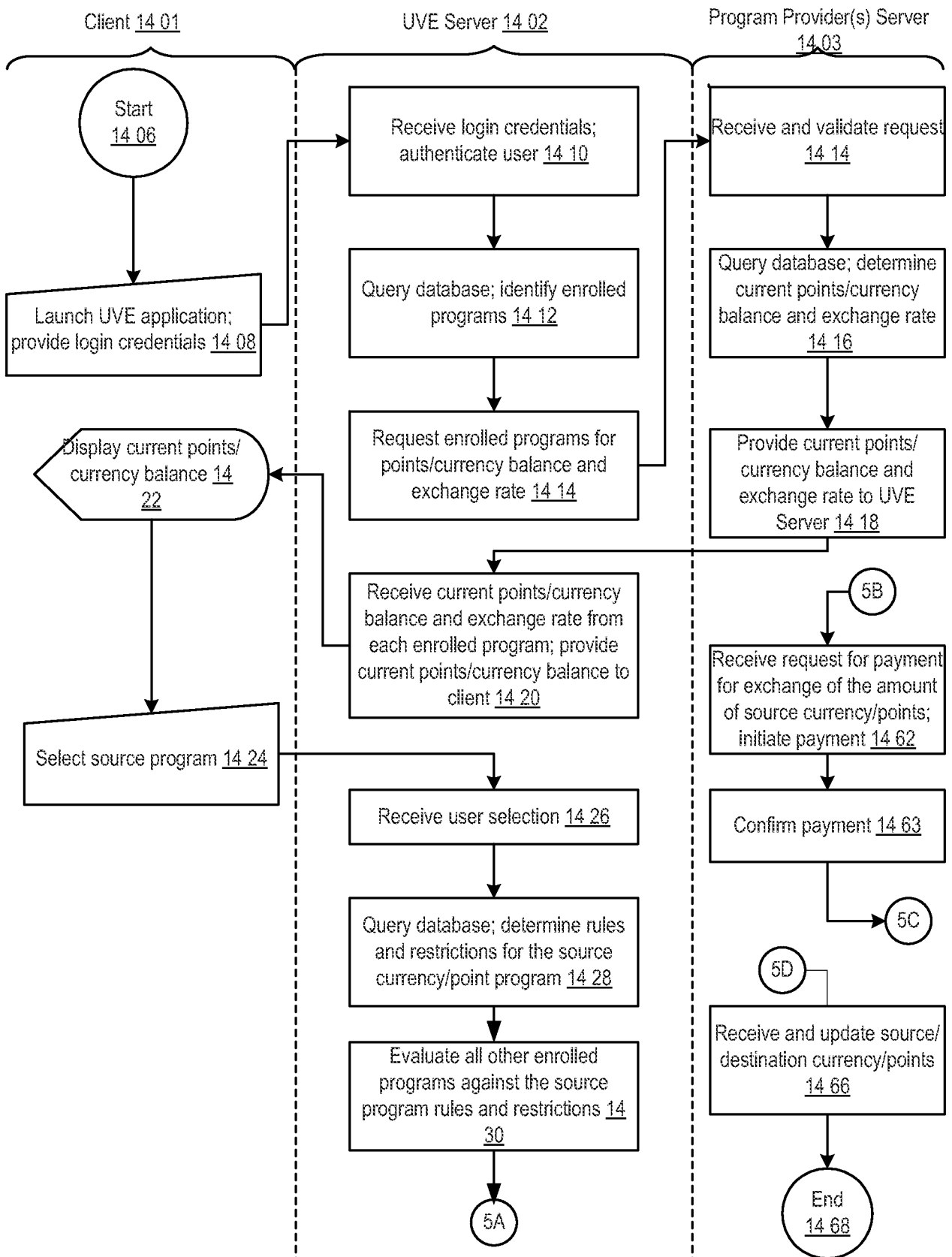
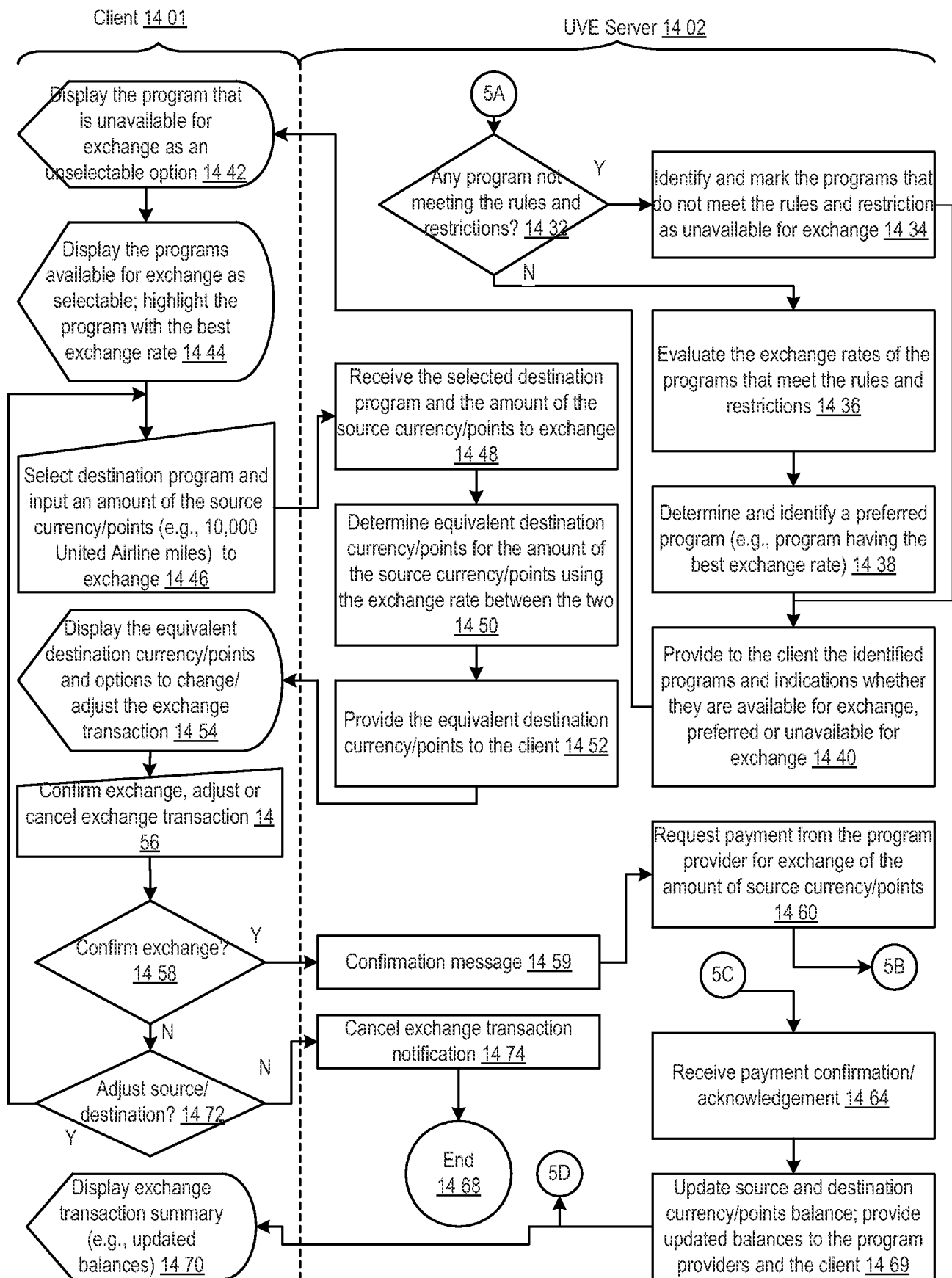
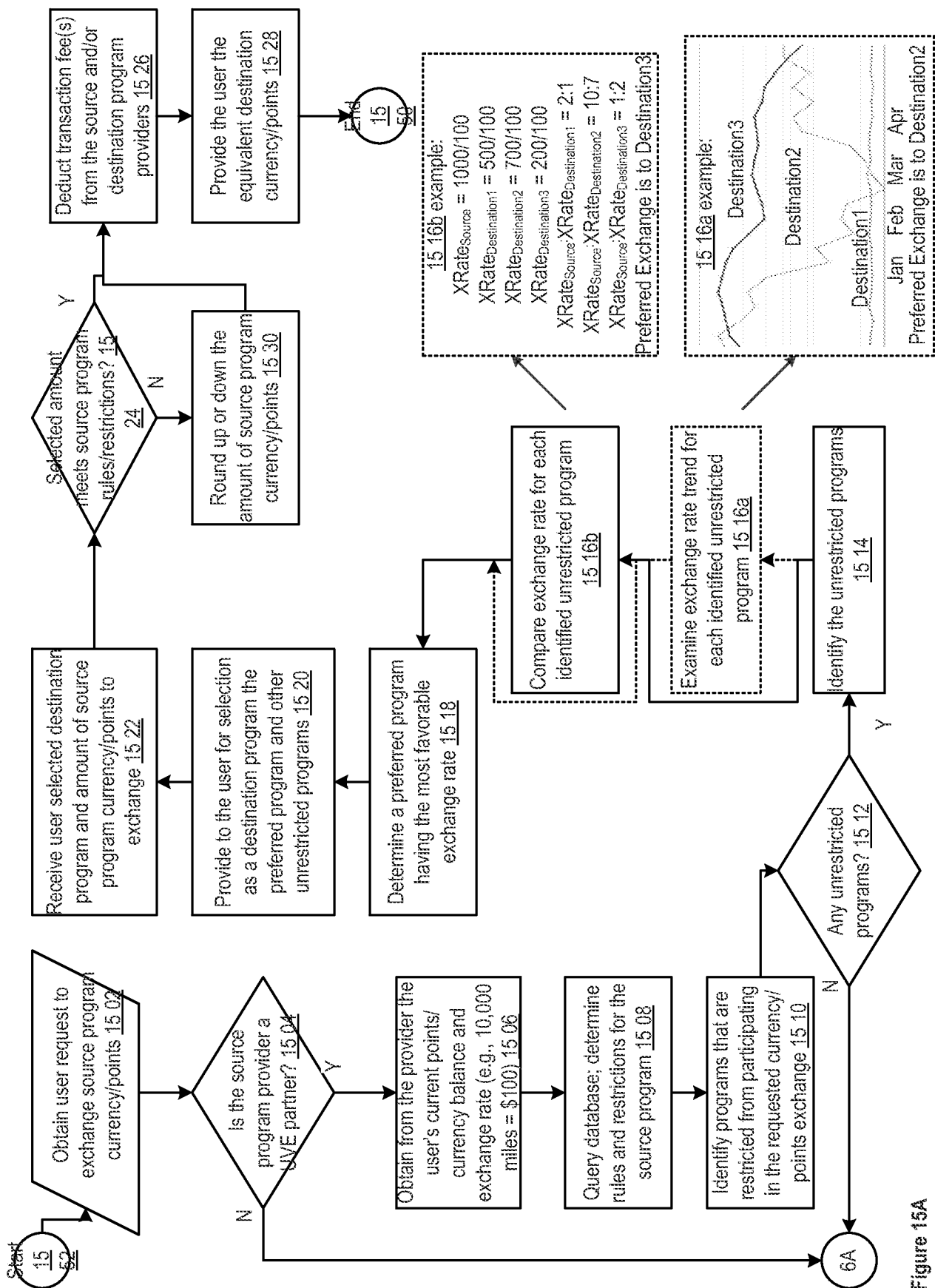


Figure 14A

Example Logic Flow: Source/Destination Universal Value Exchange (SD-UVE) Component 501





Example Logic Flow: Equivalent Value Determination (EVD) Component 15 01

Figure 15A

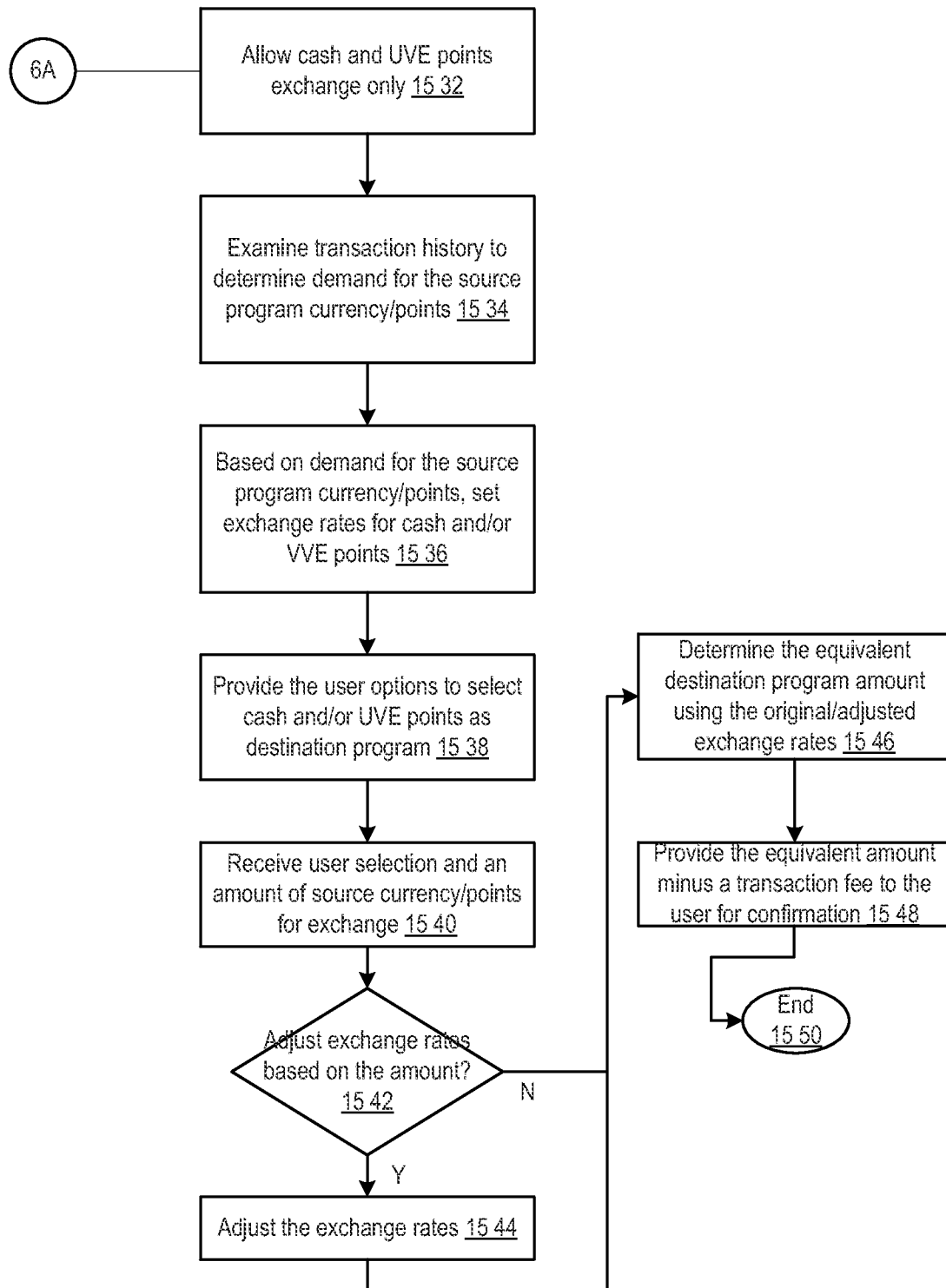


Figure 15B

Example Logic Flow: Equivalent Value Determination Component 15 01

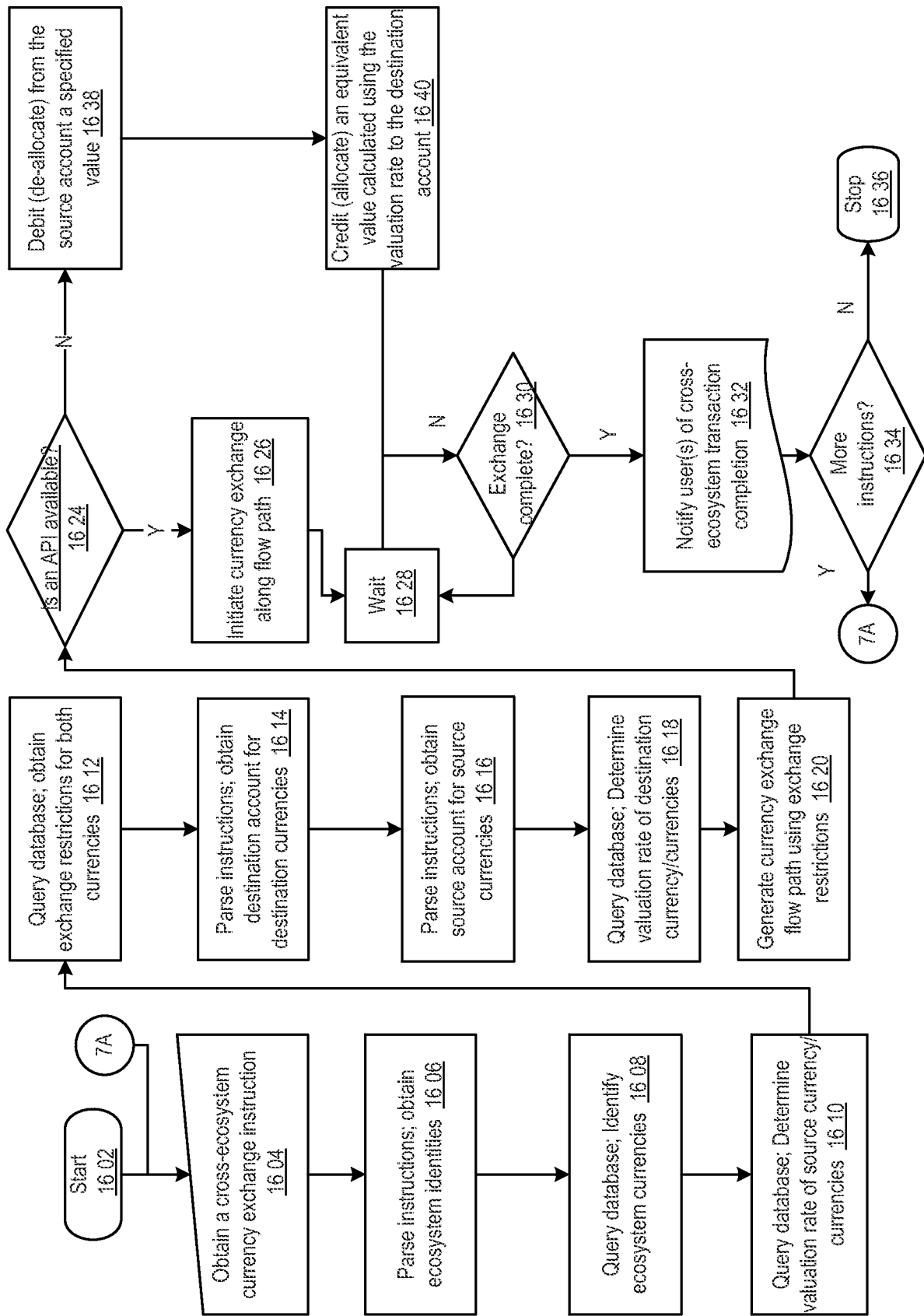


Figure 16 Example Logic Flow: Cross-Ecosystem Universal Value Exchange (CE-UVE) Component 16 01

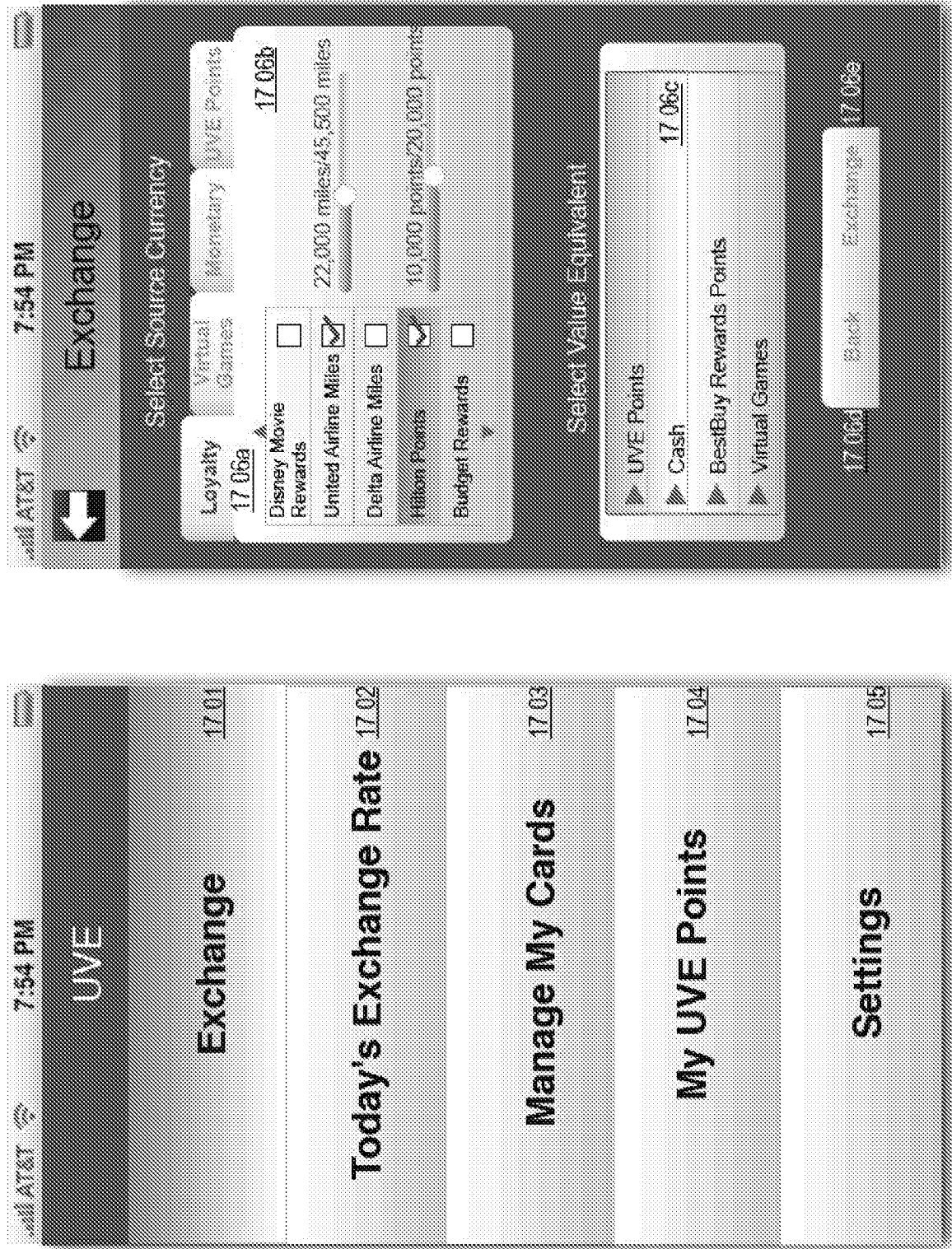
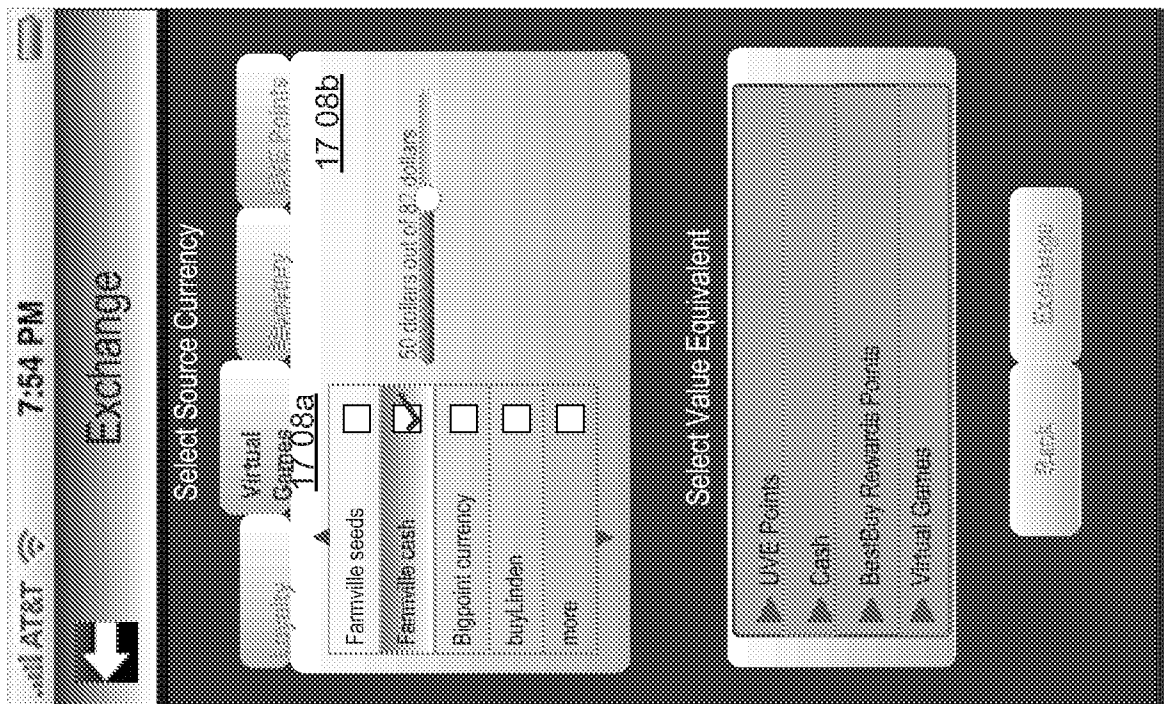
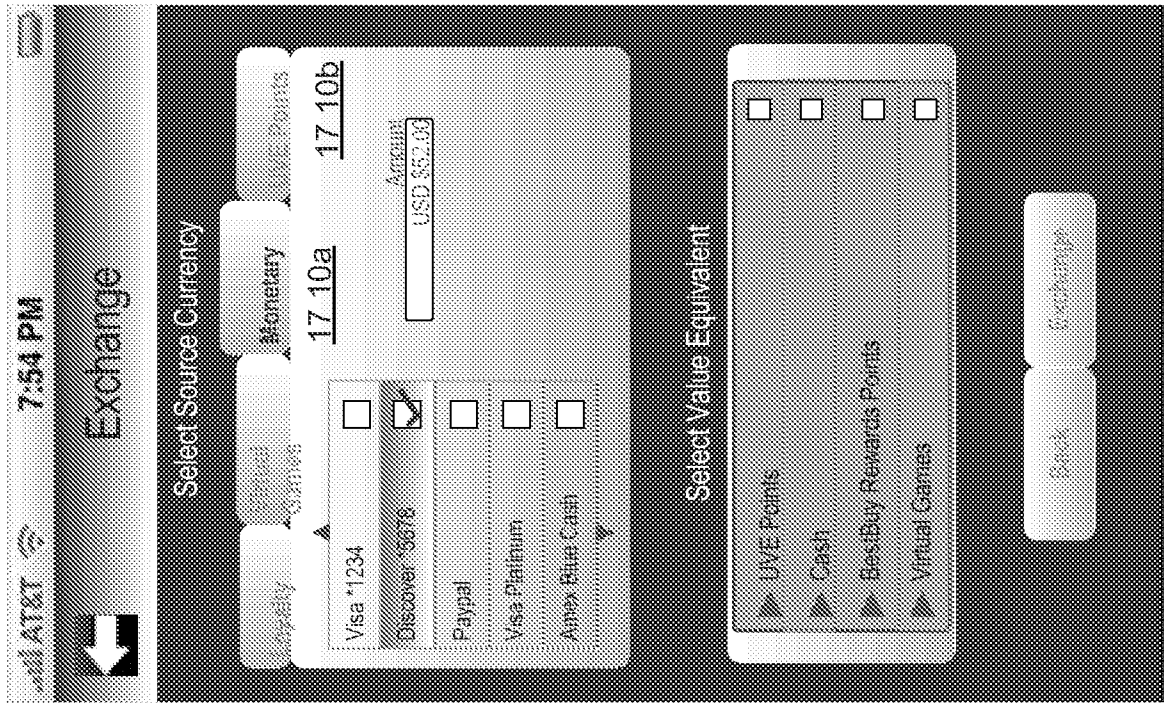


Figure 17A

Example: Universal Value Exchange -- Exchange Mode



Example: Universal Value Exchange -- Exchange Mode

Figure 17B

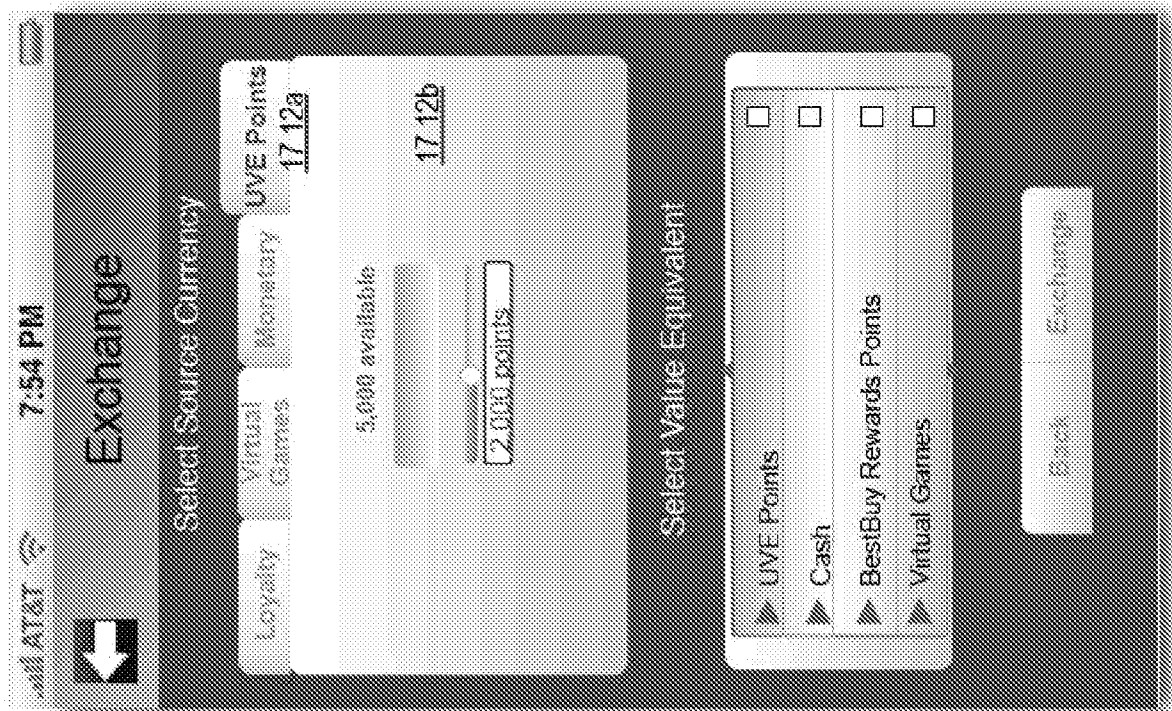
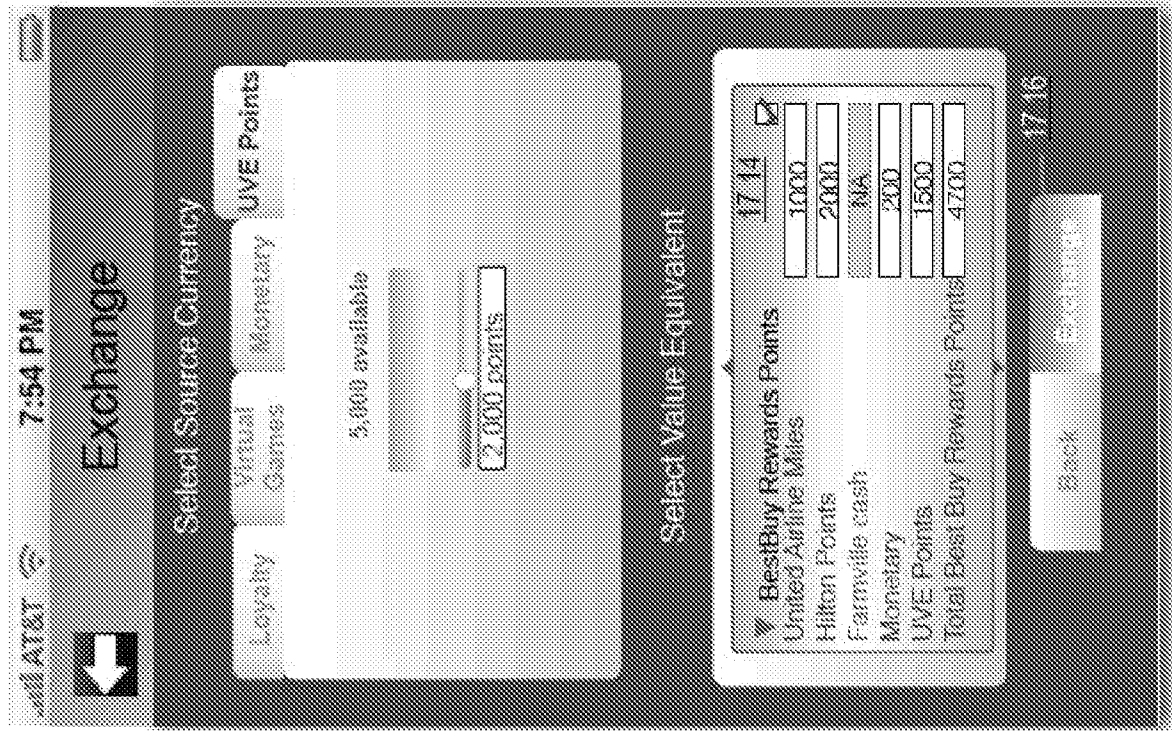


Figure 17C

Example: Universal Value Exchange -- Exchange Mode

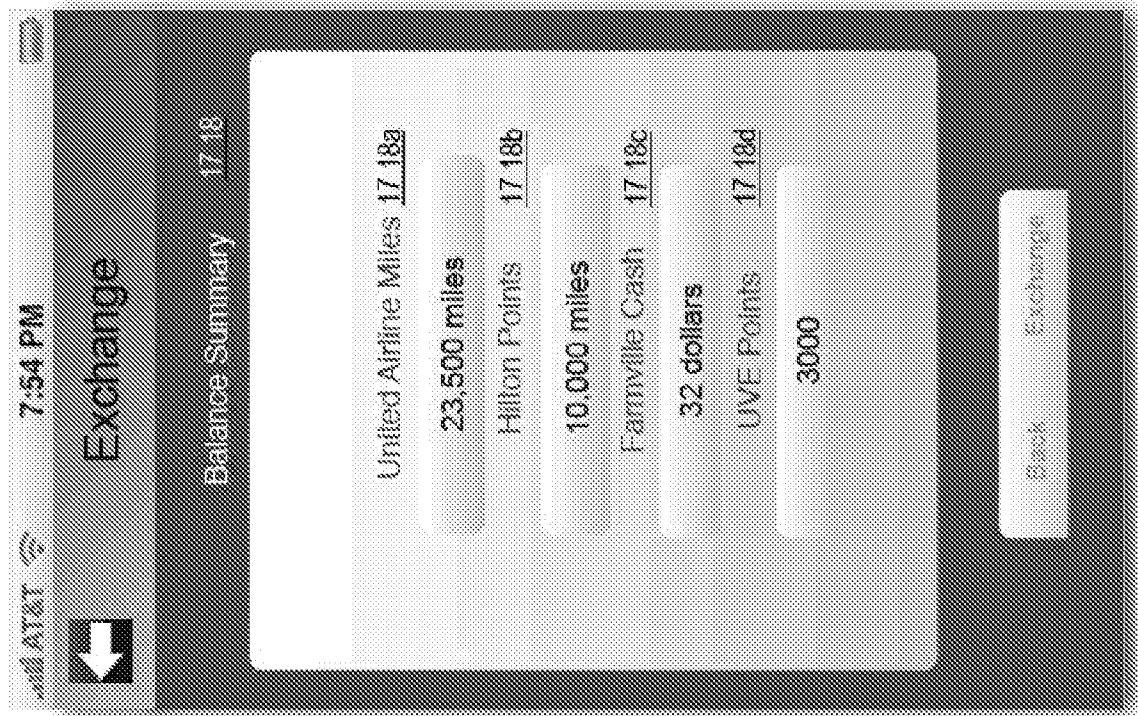


Figure 17D

Example: Universal Value Exchange -- Exchange Mode

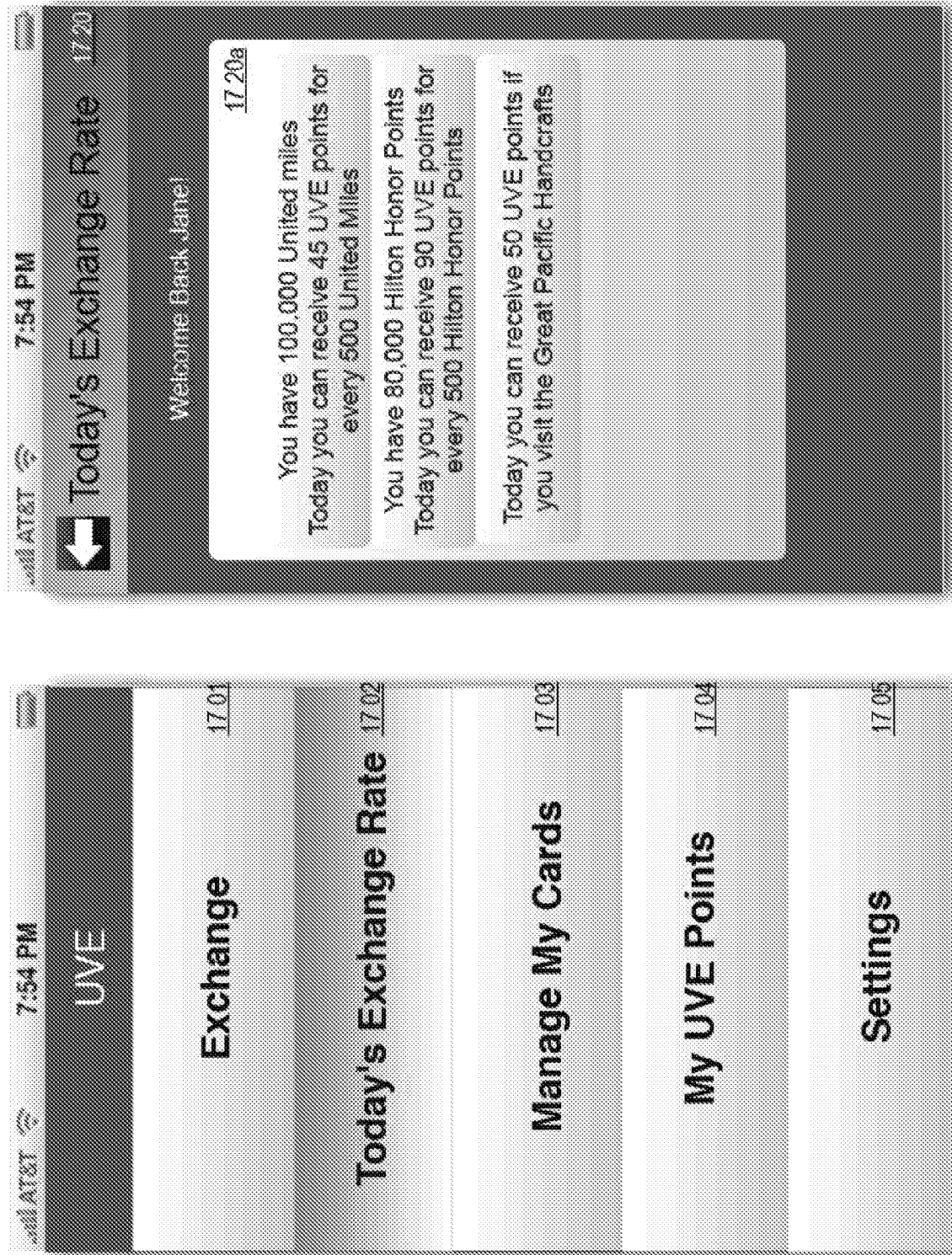


Figure 17E

Example: Universal Value Exchange -- Exchange Rate Mode



Figure 17F

Example: Universal Value Exchange -- Management mode

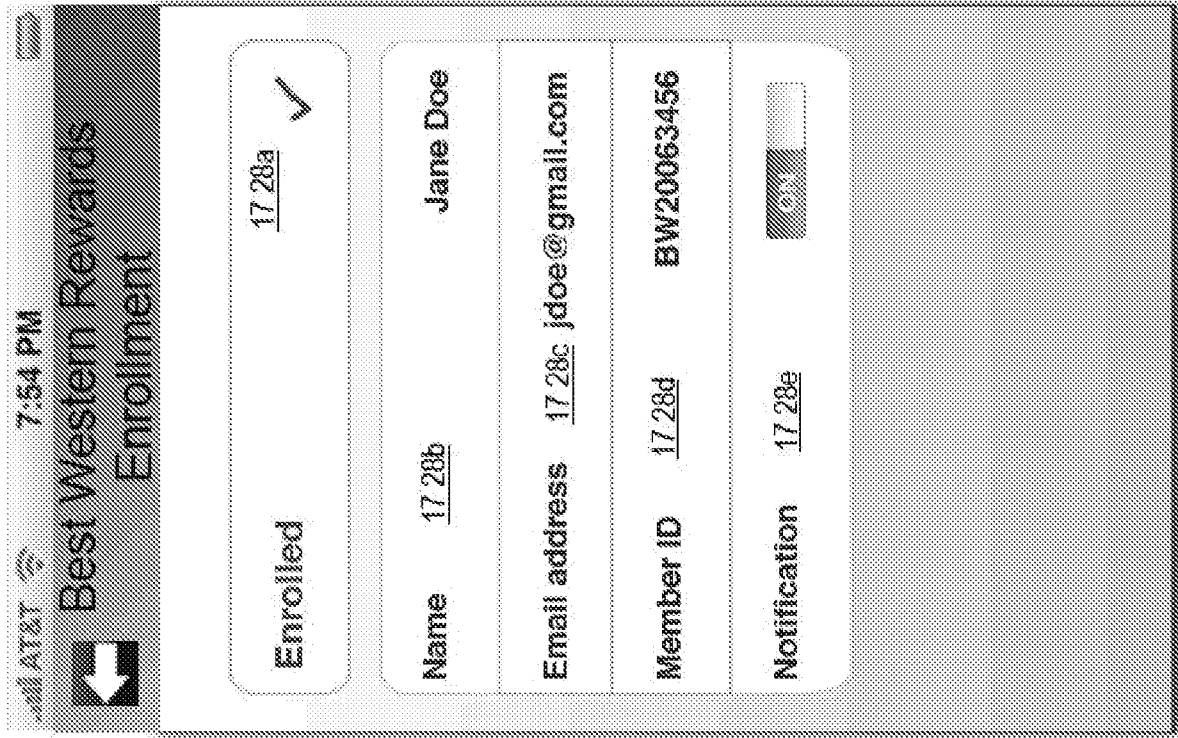


Figure 17G Example: Universal Value Exchange -- Management Mode

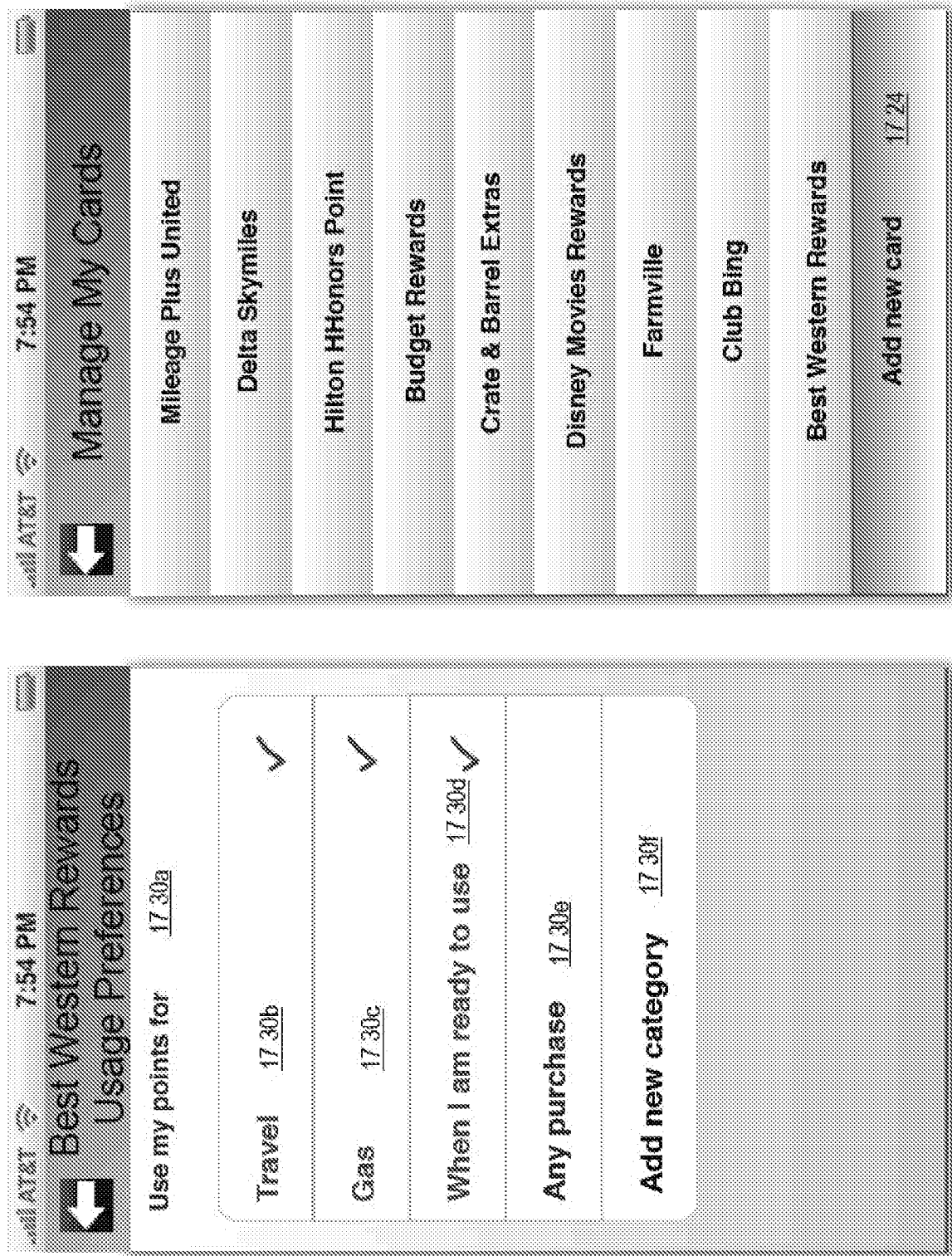


Figure 17H Example: Universal Value Exchange -- Management Mode

AT&T

7:54 PM

Best Western Rewards

Add New Card

Name

17 32a

Jane Doe

X

Email

17 32b

jdoe@gmail.com

X

Member ID

17 32c

000000162678792

X

Username

17 32d

JDoe

X

Password

17 32e

●●●●●●●●

X

Short name

17 32f

Lufthansa miles & more

X

17 34

Add Card

Cancel

17 36

Figure 17i

Example: Universal Value Exchange -- Management Mode

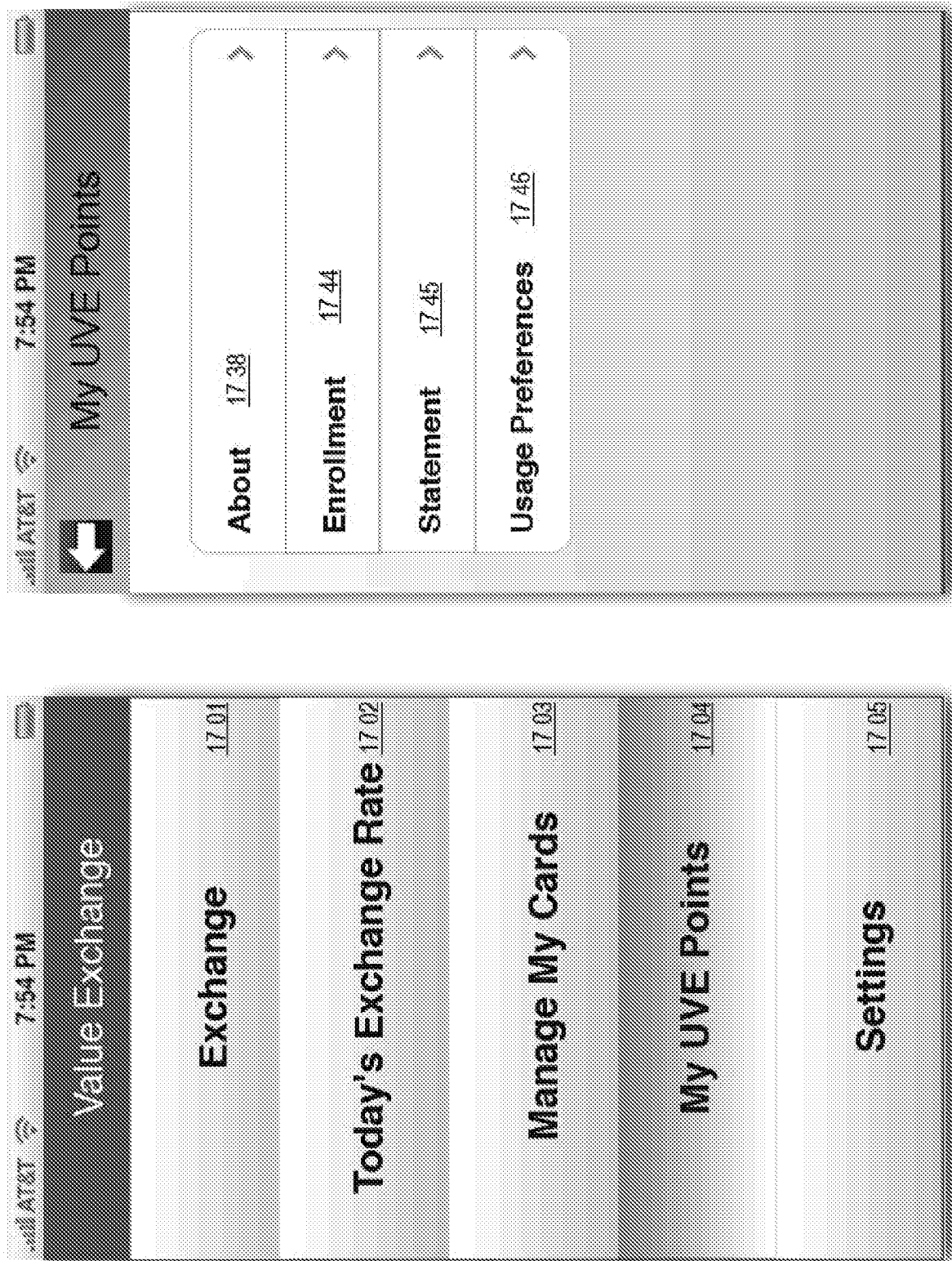


Figure 17J

Example: Universal Value Exchange -- UVE Points Mode

AT&T 7:54 PM

My UVE Points Enrollment

Enrolled 17.44a ✓

Name 17.44b Jane Doe

Email address 17.44c jdoe@gmail.com

Phone 17.44d 5036093324

Visa Card No. 17.44e 4444444444444444

Security Code 17.44f 2345

Billing Zip Code 17.44g 97123

AT&T 7:54 PM

My UVE Points Usage Preferences

Use my UVE points for

Travel 17.46a

Gas 17.46b ✓

When I am ready to use 17.46c

Groceries 17.46d

Add new category 17.46e

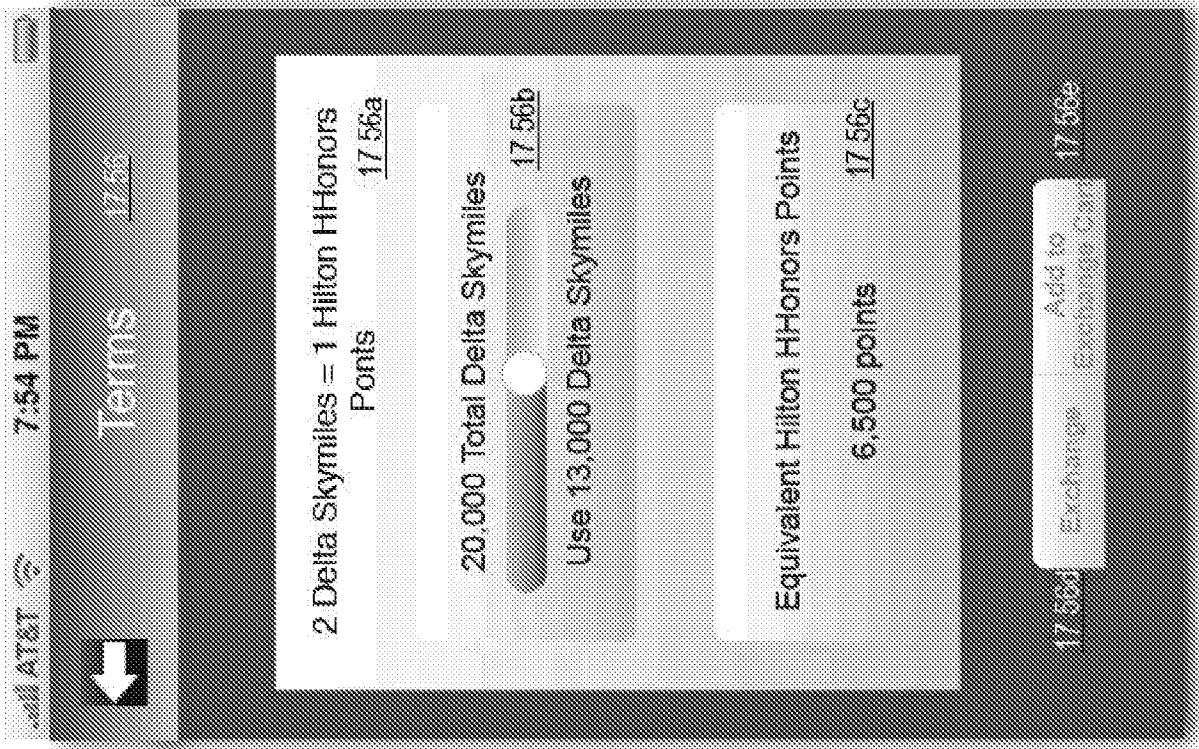
Figure 17K

Example: Universal Value Exchange -- UVE Points Mode

AT&T	7:54 PM	
←	Source	17.50
Mileage Plus United		17.50a
Delta Skymiles		17.50b
Hilton HHonors Point		17.50c
BestBuy Rewards		17.50d
Cash		17.50e
Card 1		17.50f
Card 2		17.50g

AT&T	7:54 PM	
←	Destination	17.52
Mileage Plus United		17.52a
Delta Skymiles		17.52b
Hilton HHonors Point		17.52c
BestBuy Rewards		17.52d
Cash		17.52e
Card 1		17.52f
Card 2		17.52g

Figure 17L Example: Universal Value Exchange -- Source/Destination Exchange mode



Example: Universal Value Exchange -- Source/Destination Exchange Mode

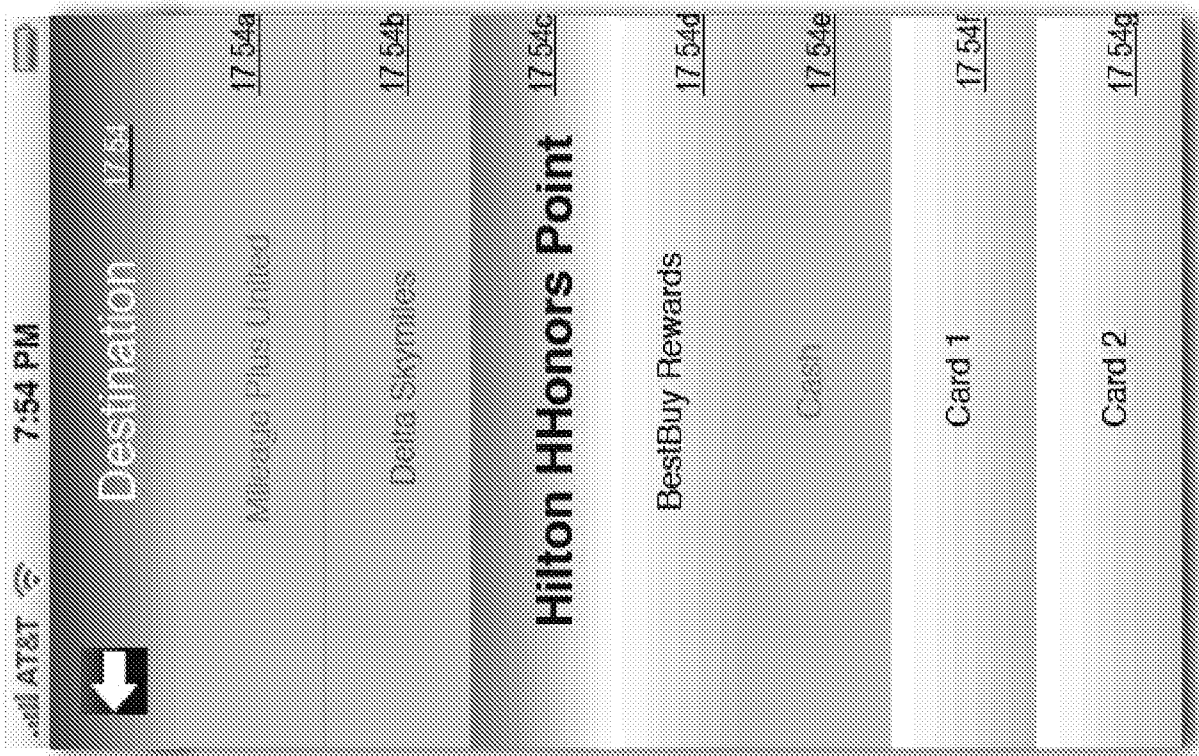


Figure 17M

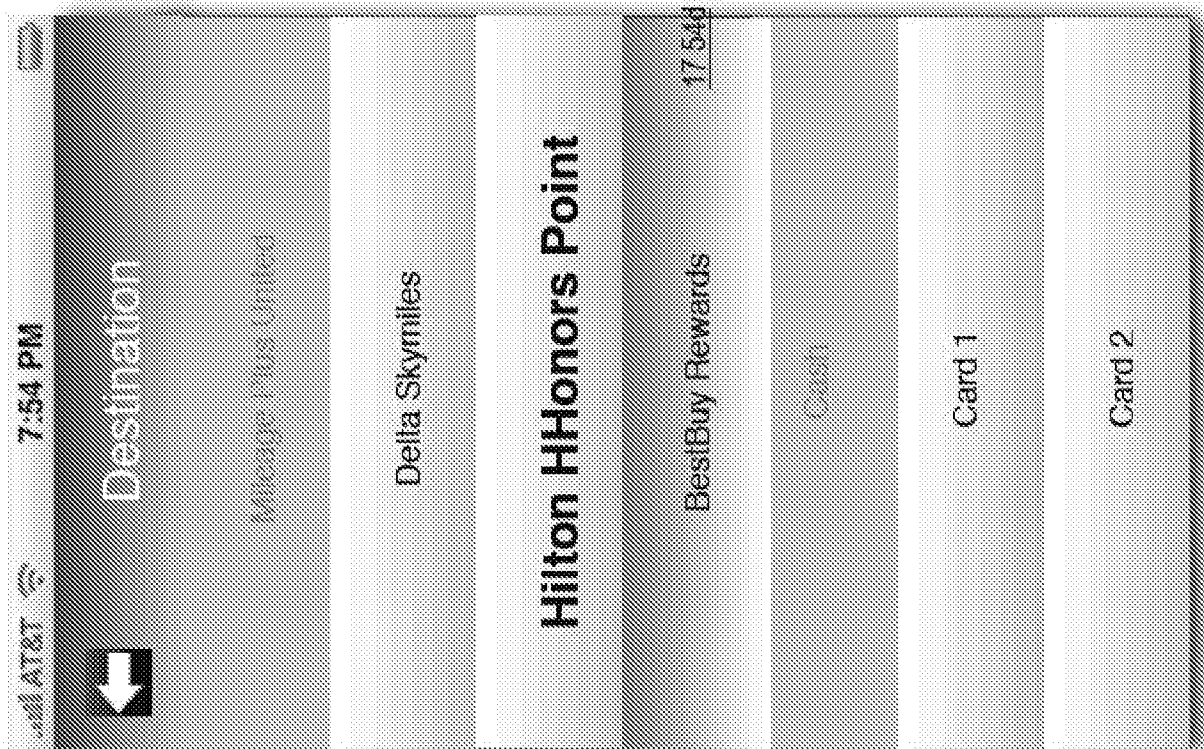
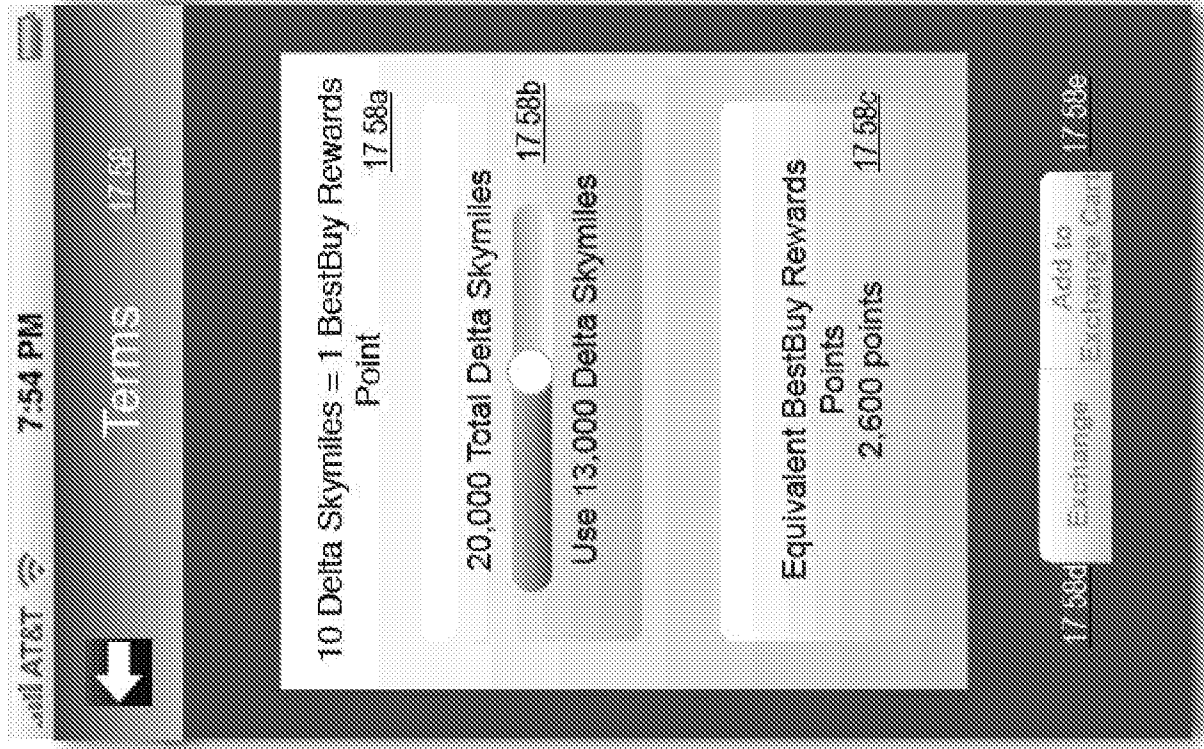


Figure 17N Example: Universal Value Exchange -- Source/Destination Exchange Mode

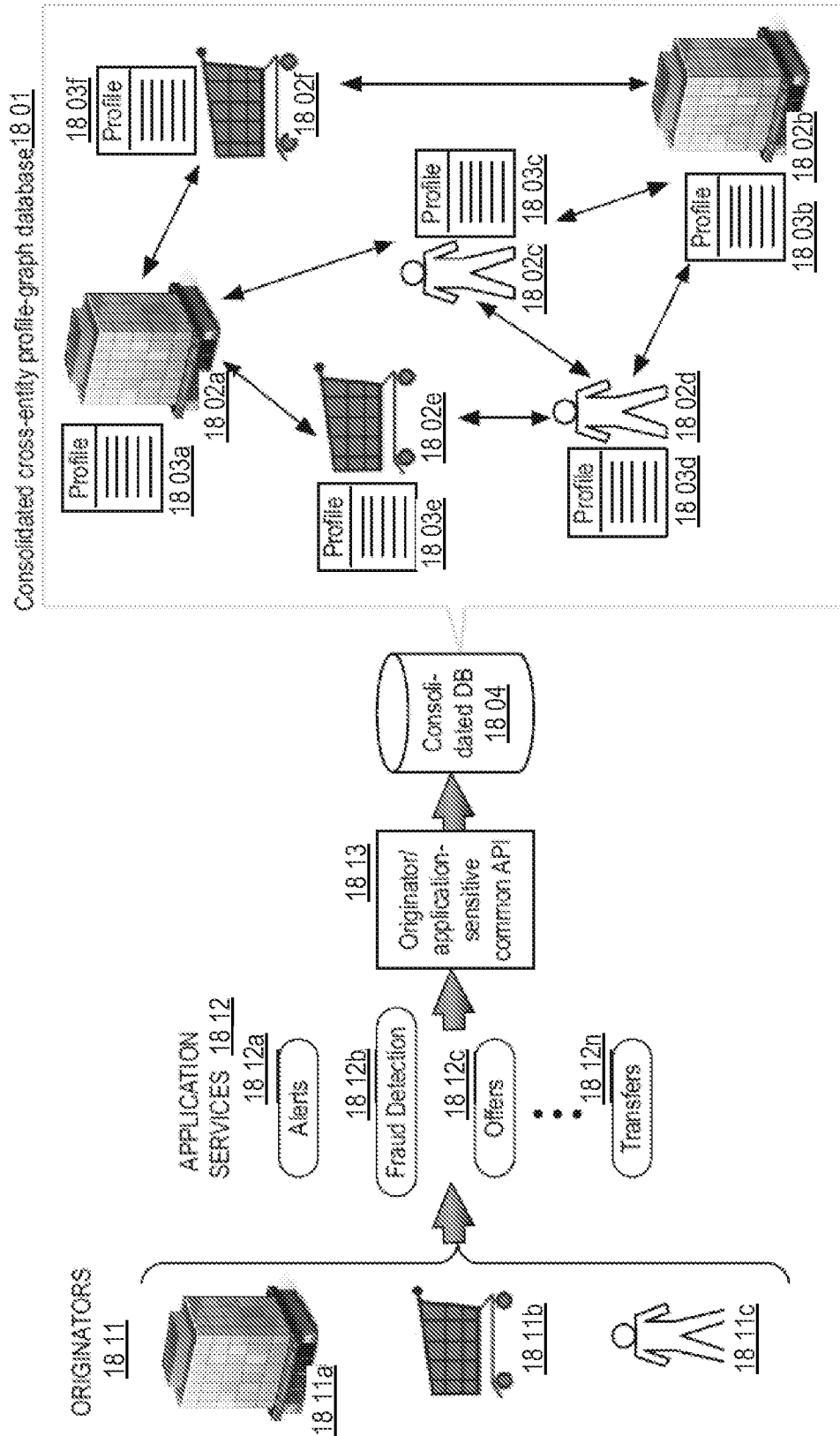


FIGURE 18 Example: Centralized Personal Information Platform

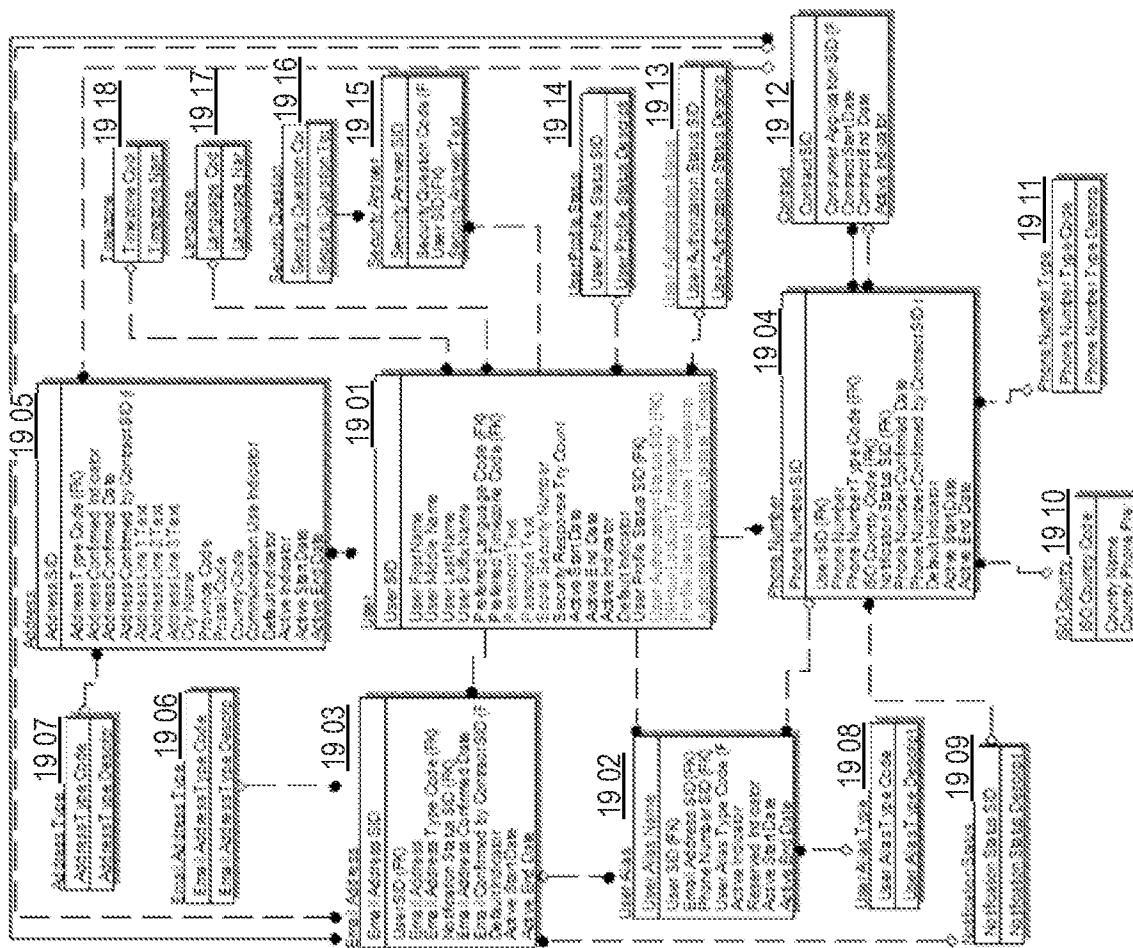


FIGURE 19A

Example: User Profile Attributes Data Model

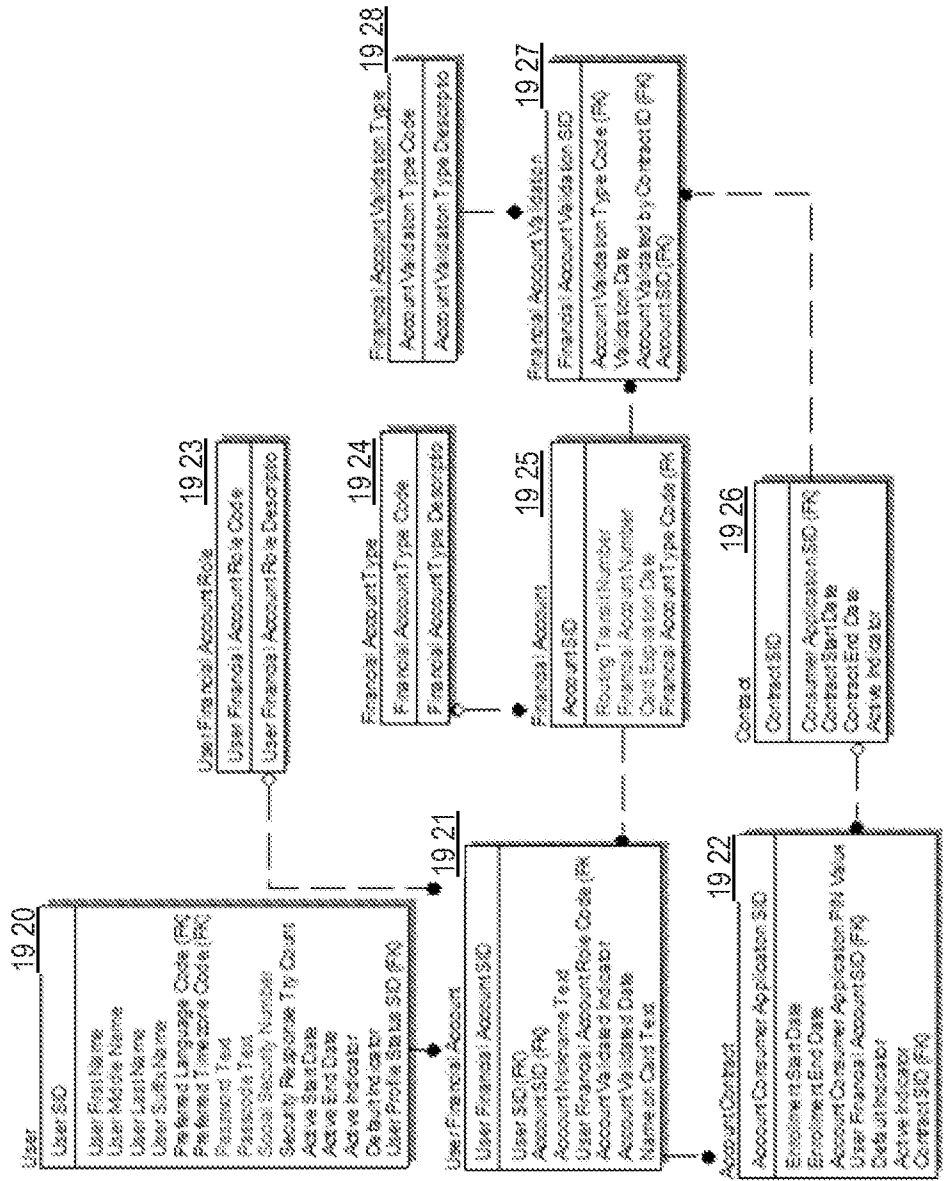


FIGURE 19B

Example: User Profile Attributes Data Model

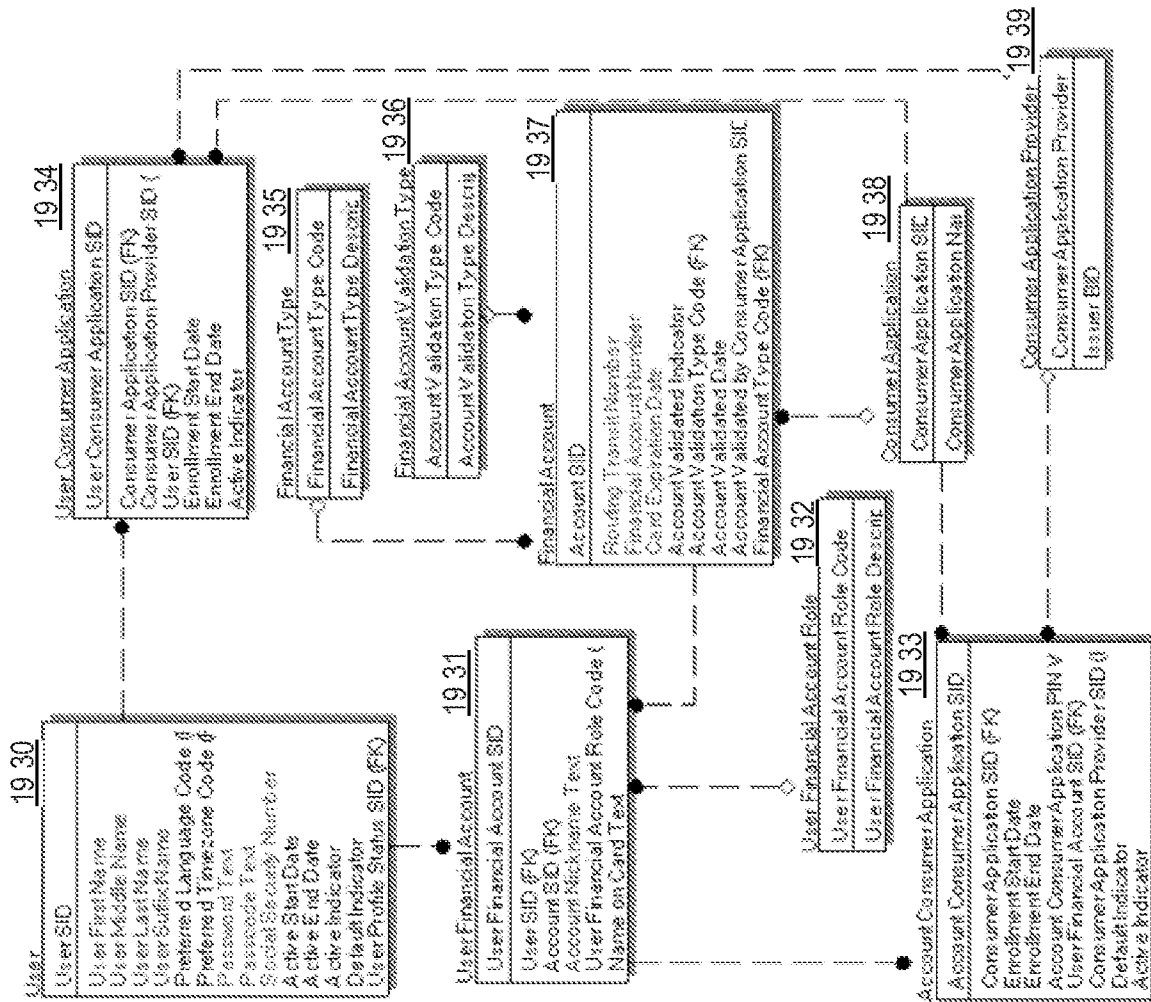


FIGURE 19C

Example: User Profile Attributes Data Model

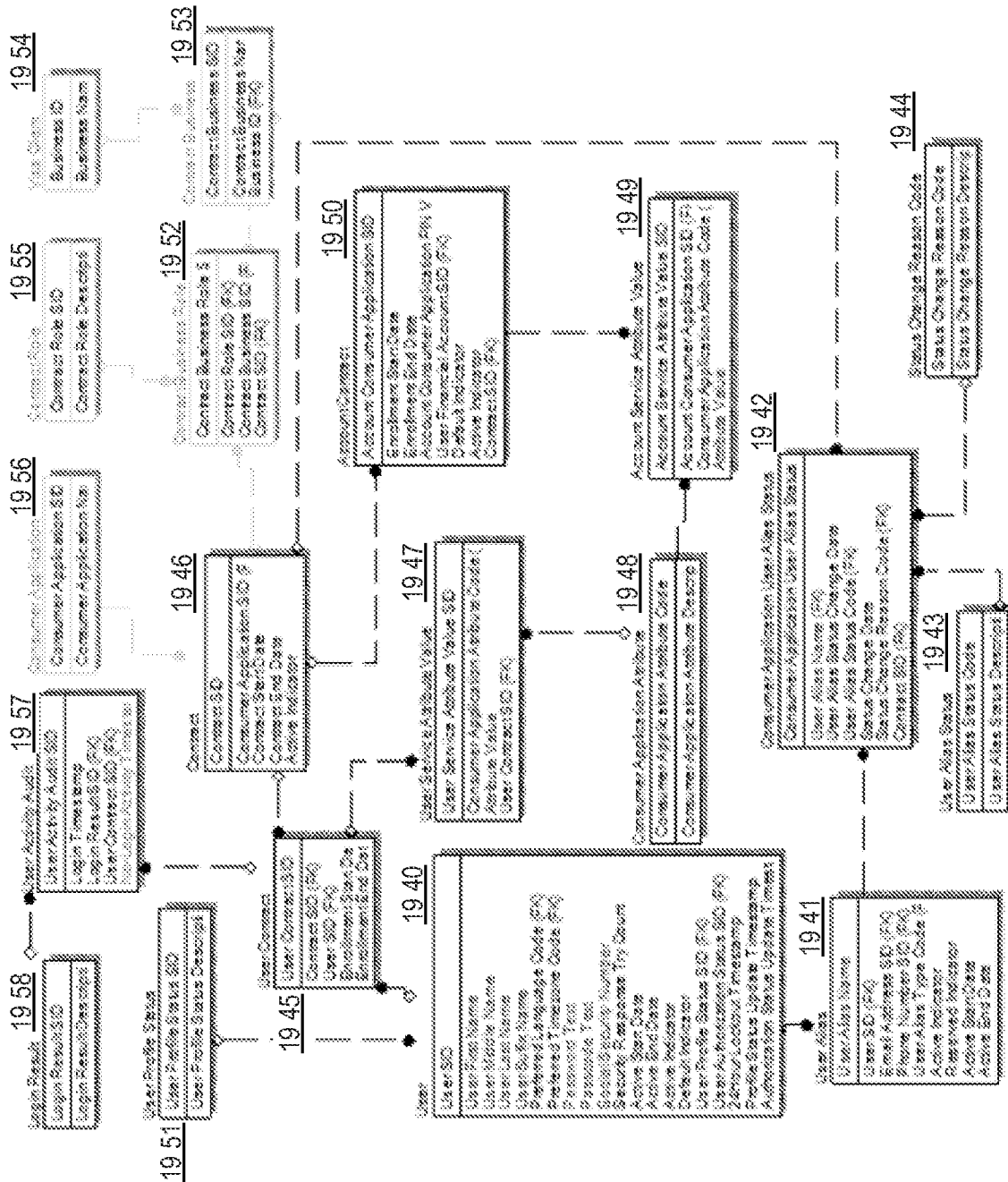


FIGURE 19D

Example: User Profile Attributes Data Model

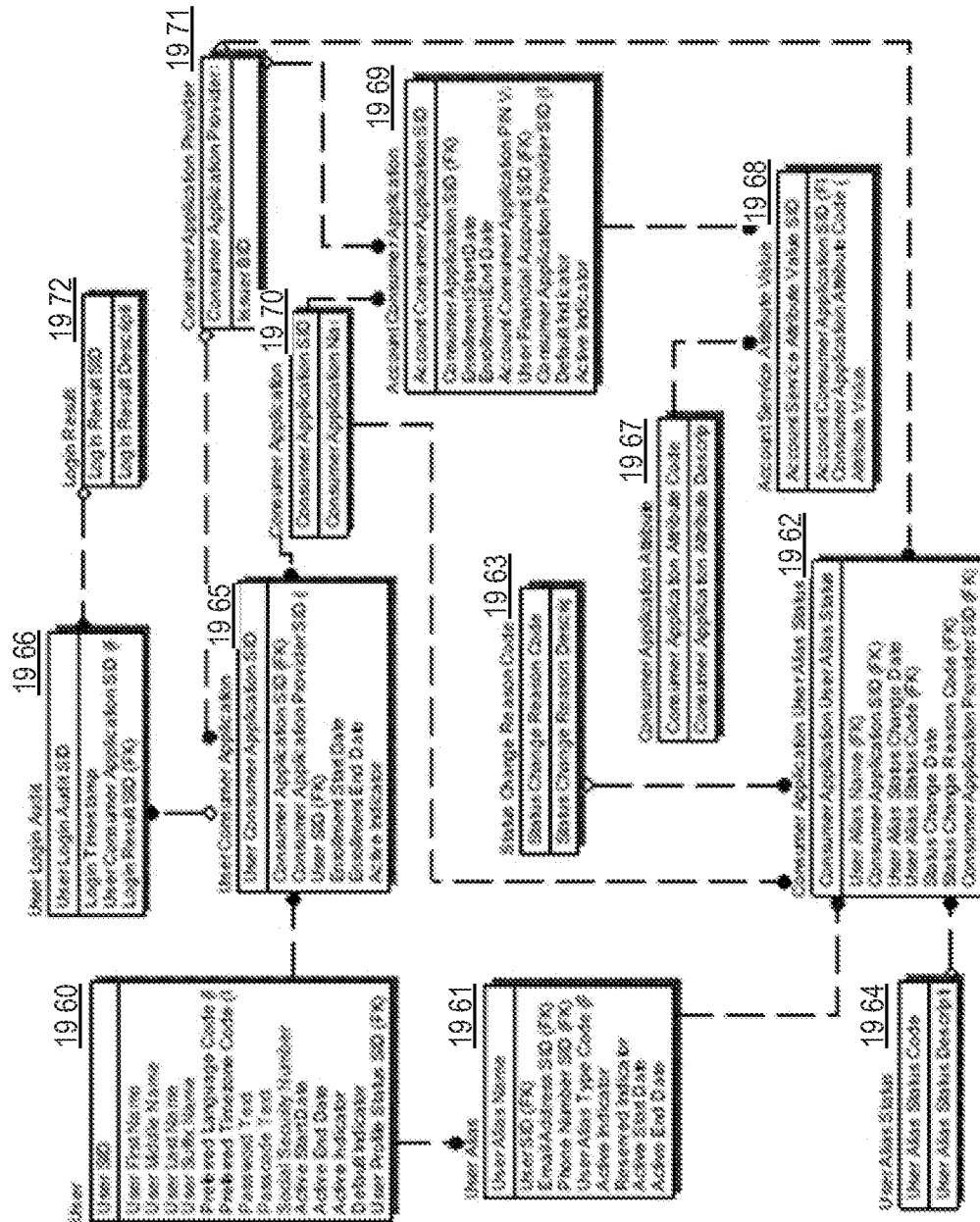


FIGURE 19E

Example: User Profile Attributes Data Model

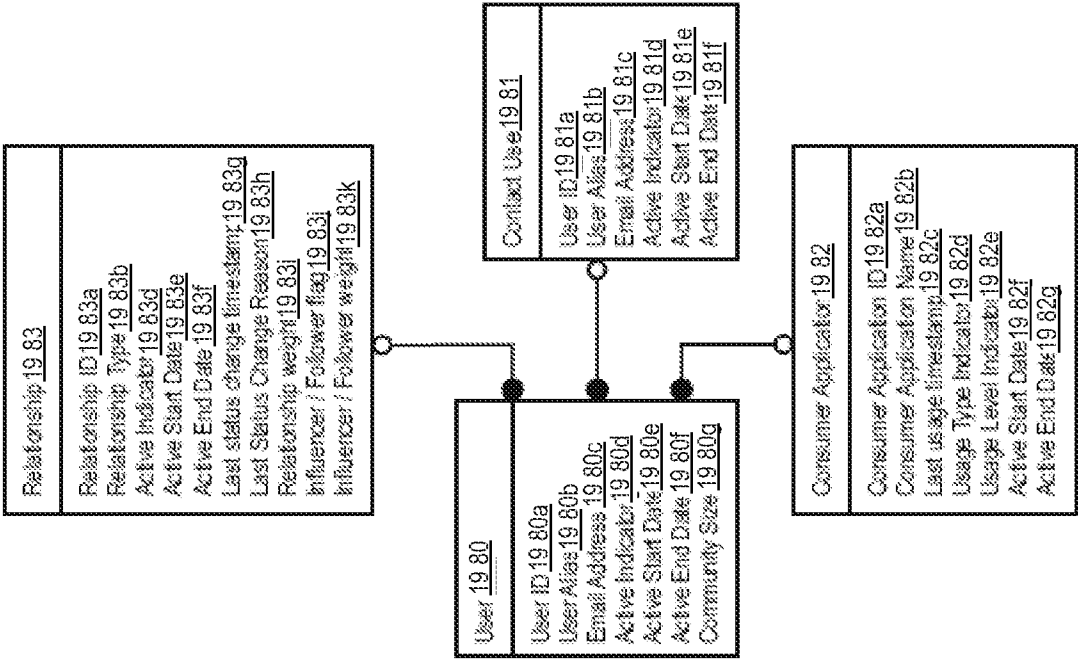


FIGURE 19F

Example: User Profile Attributes Data Model

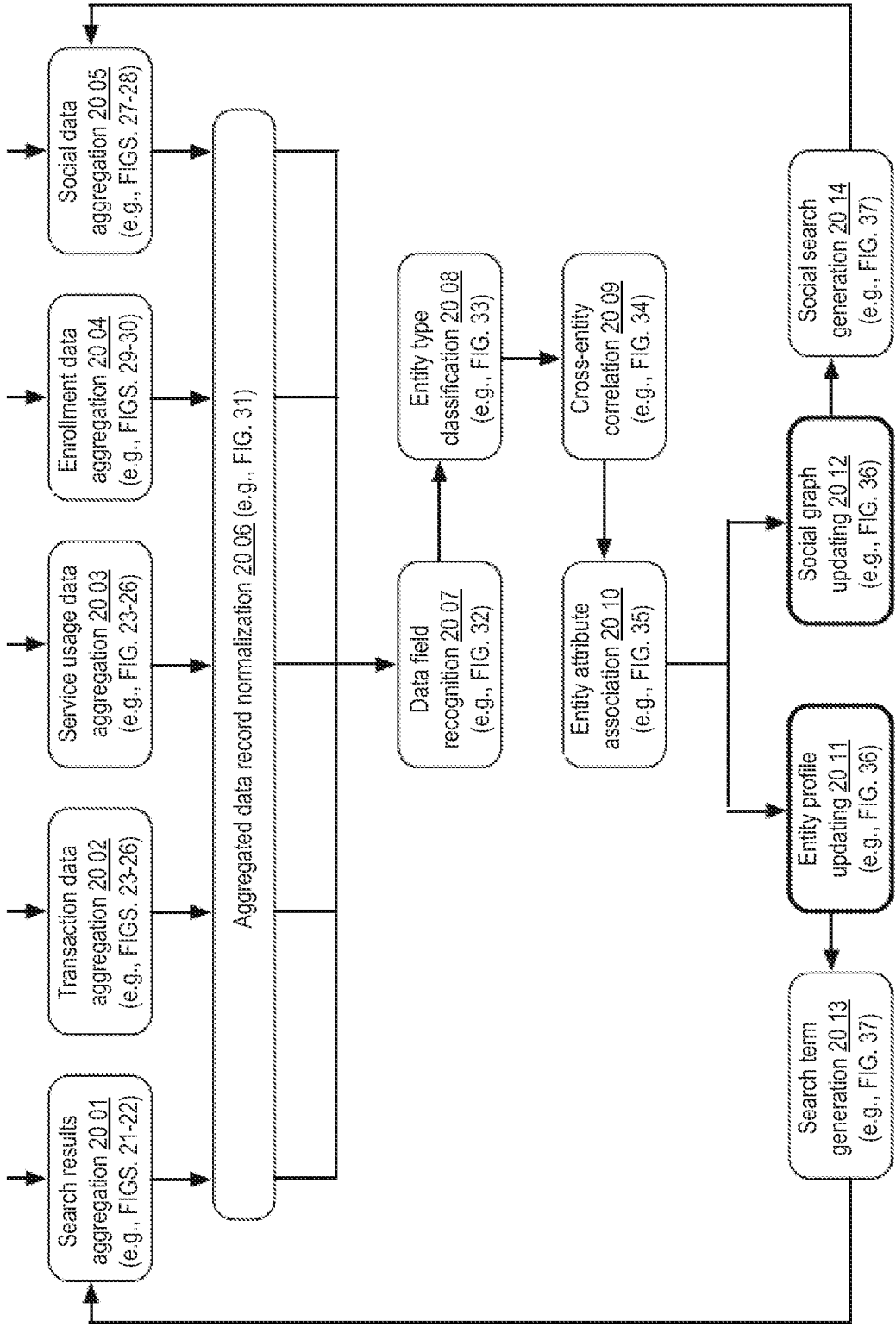


FIGURE 20

Example: Centralized Personal Information Platform Components

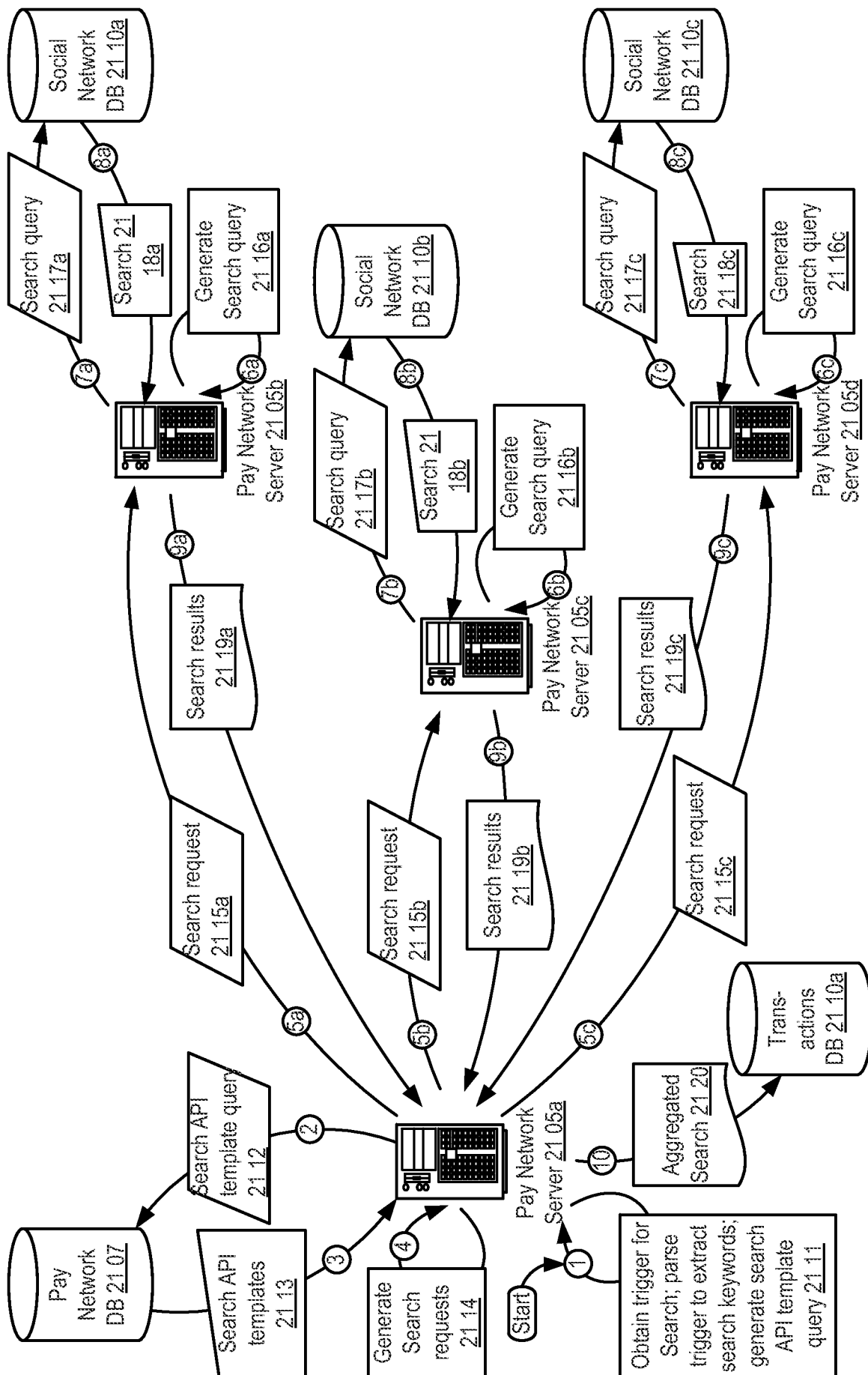
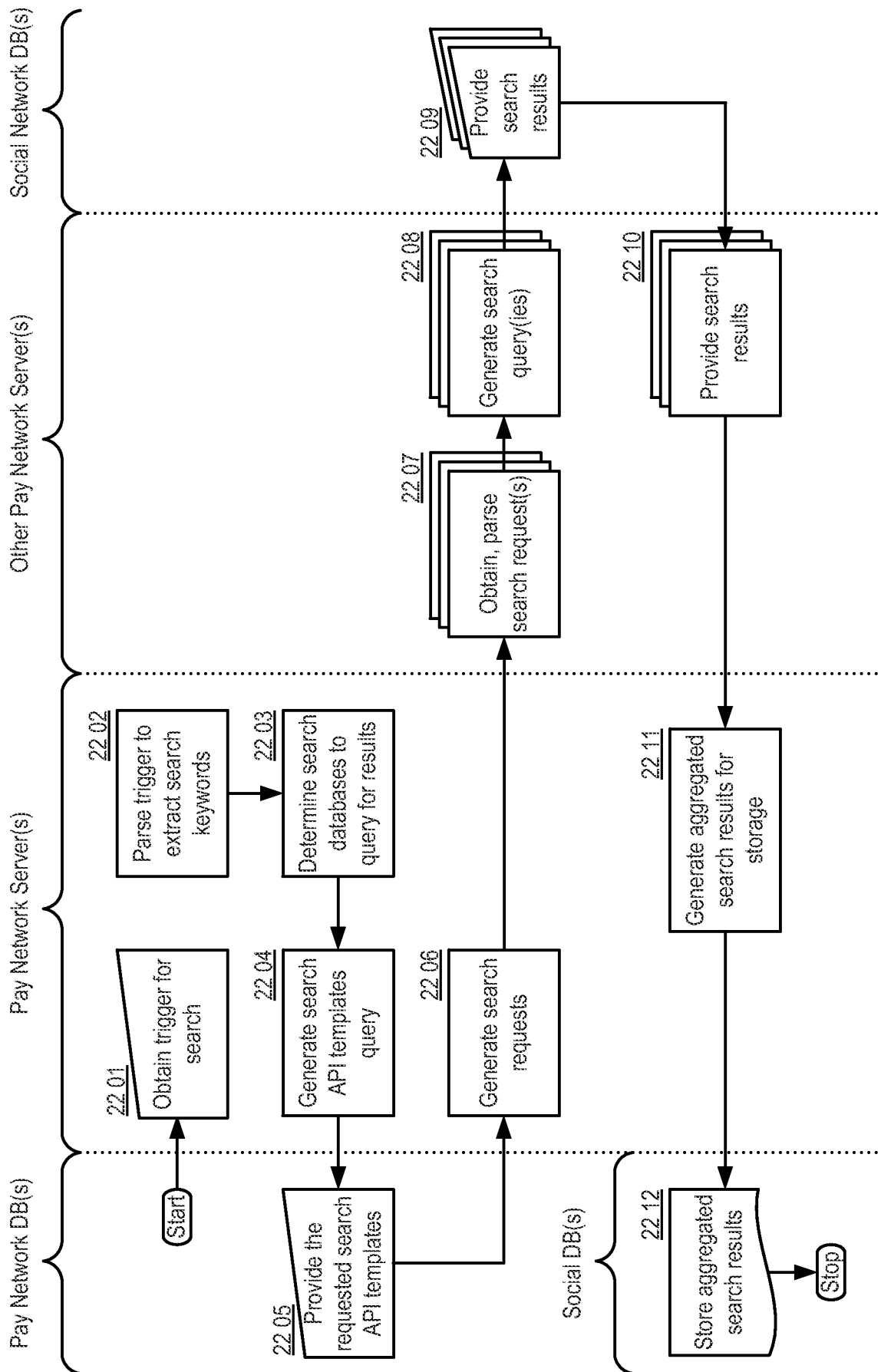


FIGURE 21

Example: Search Results Aggregation



Example: Search Results Aggregation ("SRA") component 2200

FIGURE 22

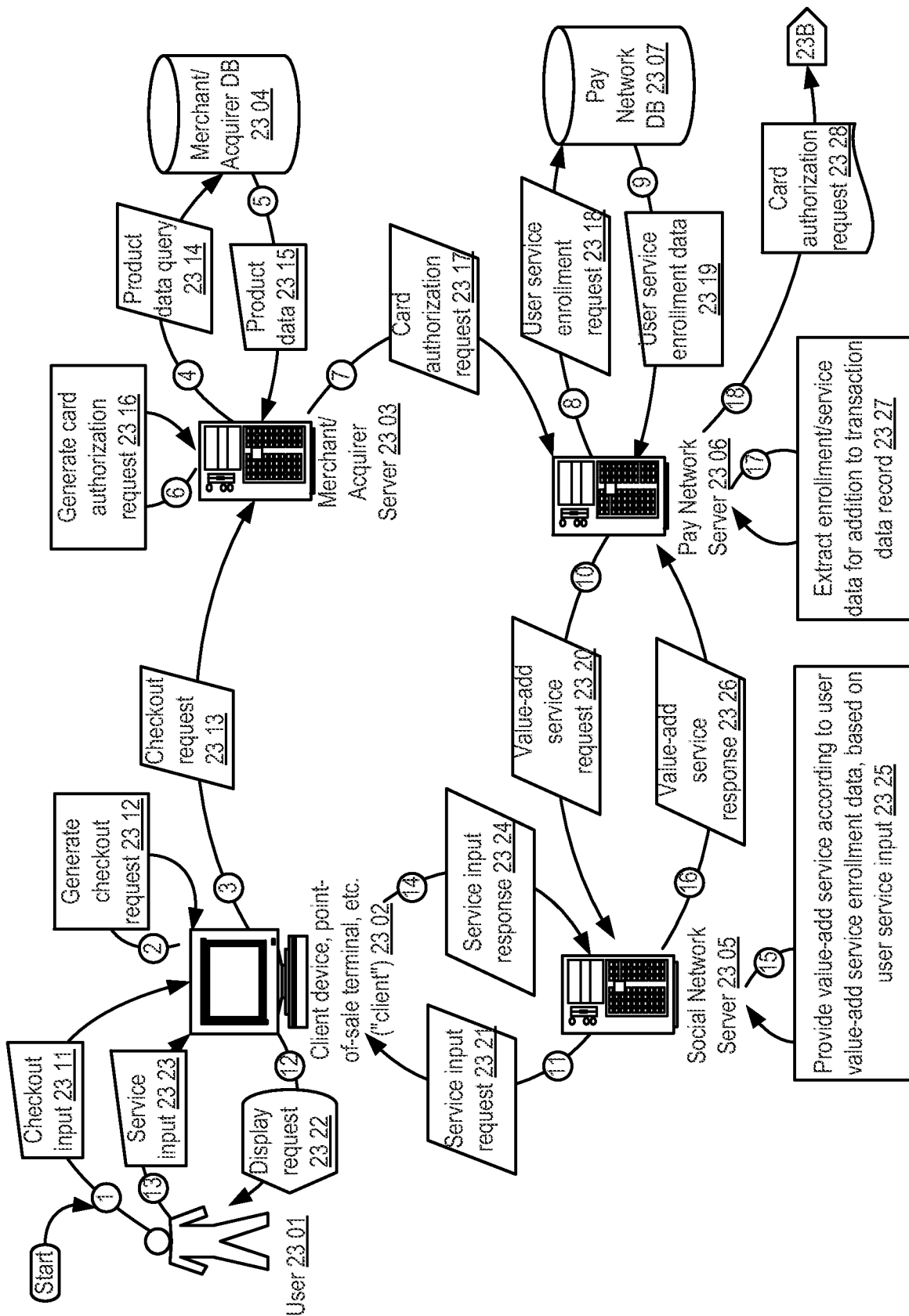


FIGURE 23A

Example: Card-Based Transaction Execution

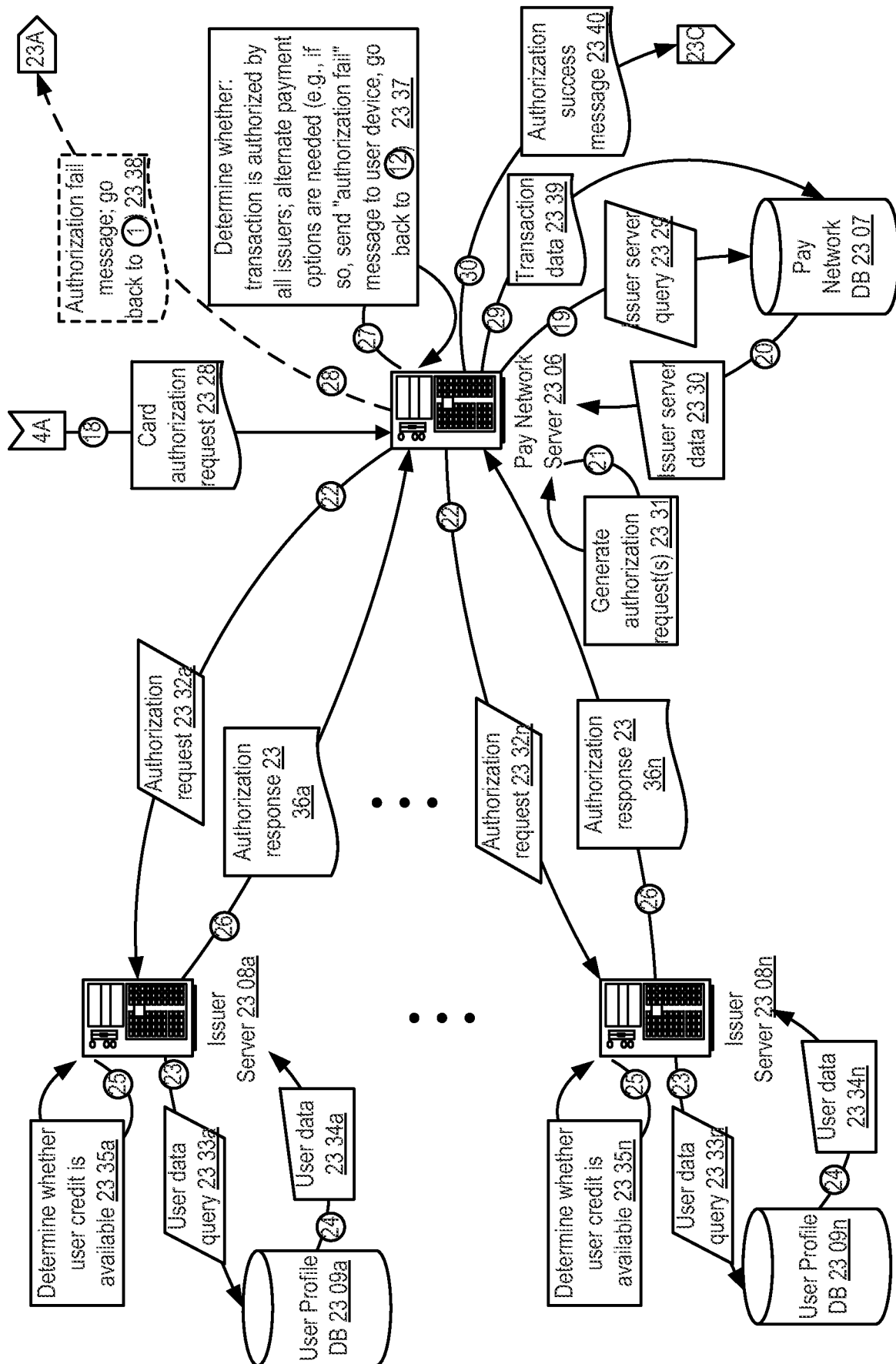


FIGURE 23B

Example: Card-Based Transaction Execution

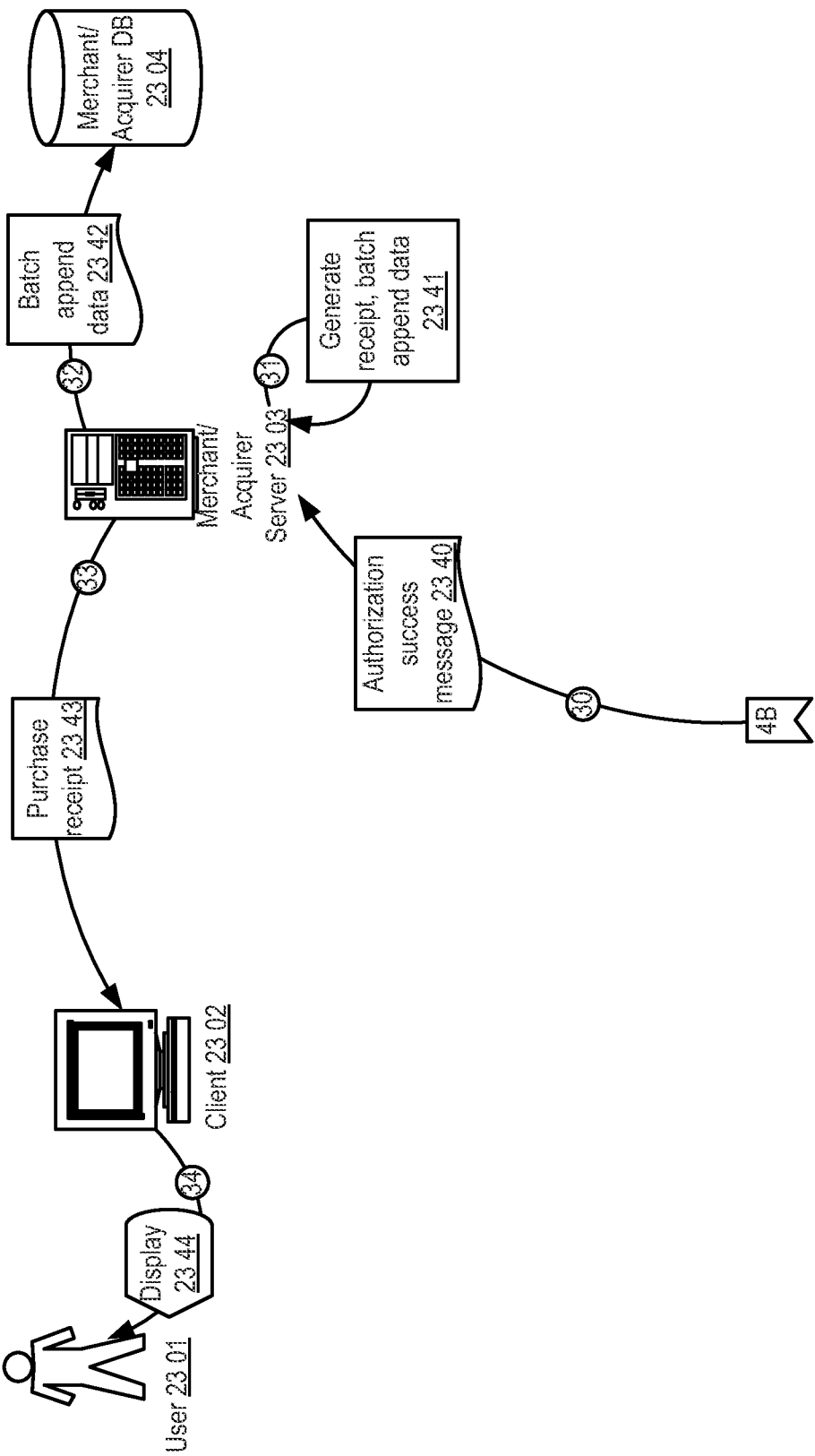
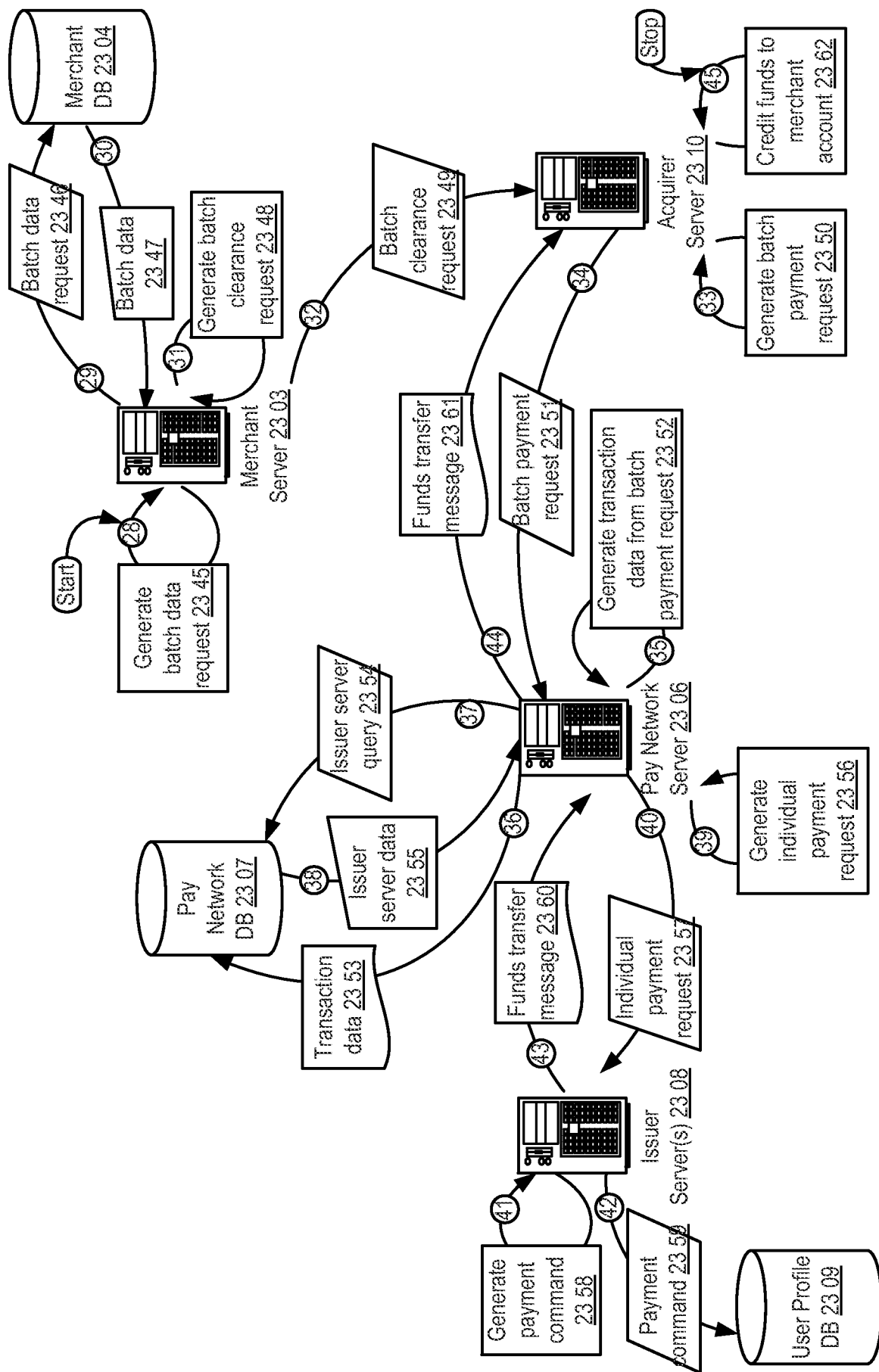


FIGURE 23C

Example: Card-Based Transaction Execution



Example: Card-Based Transaction Execution

FIGURE 23D

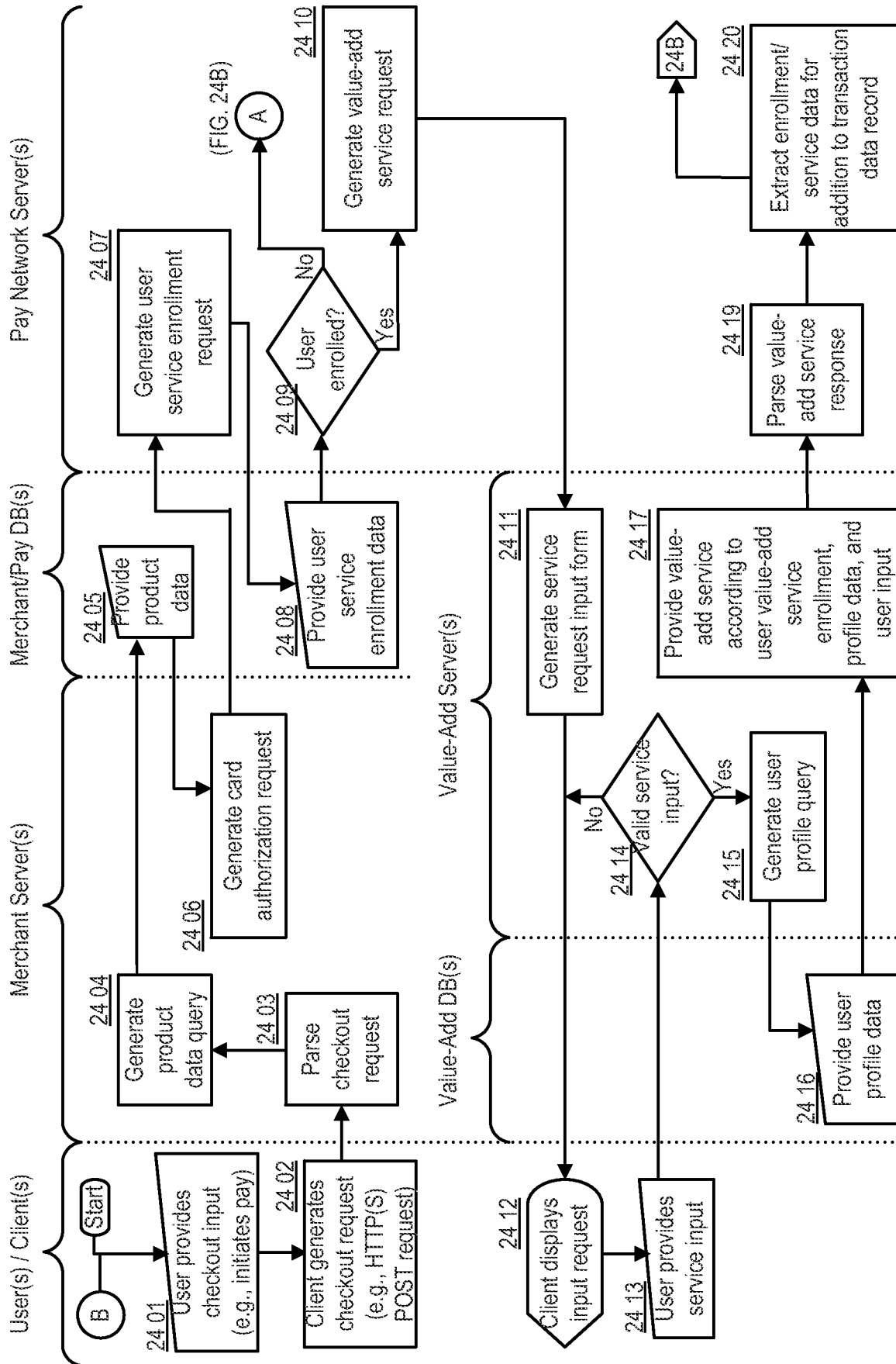


FIGURE 24A

Example: Card-Based Transaction Execution ("CTE") component 2400

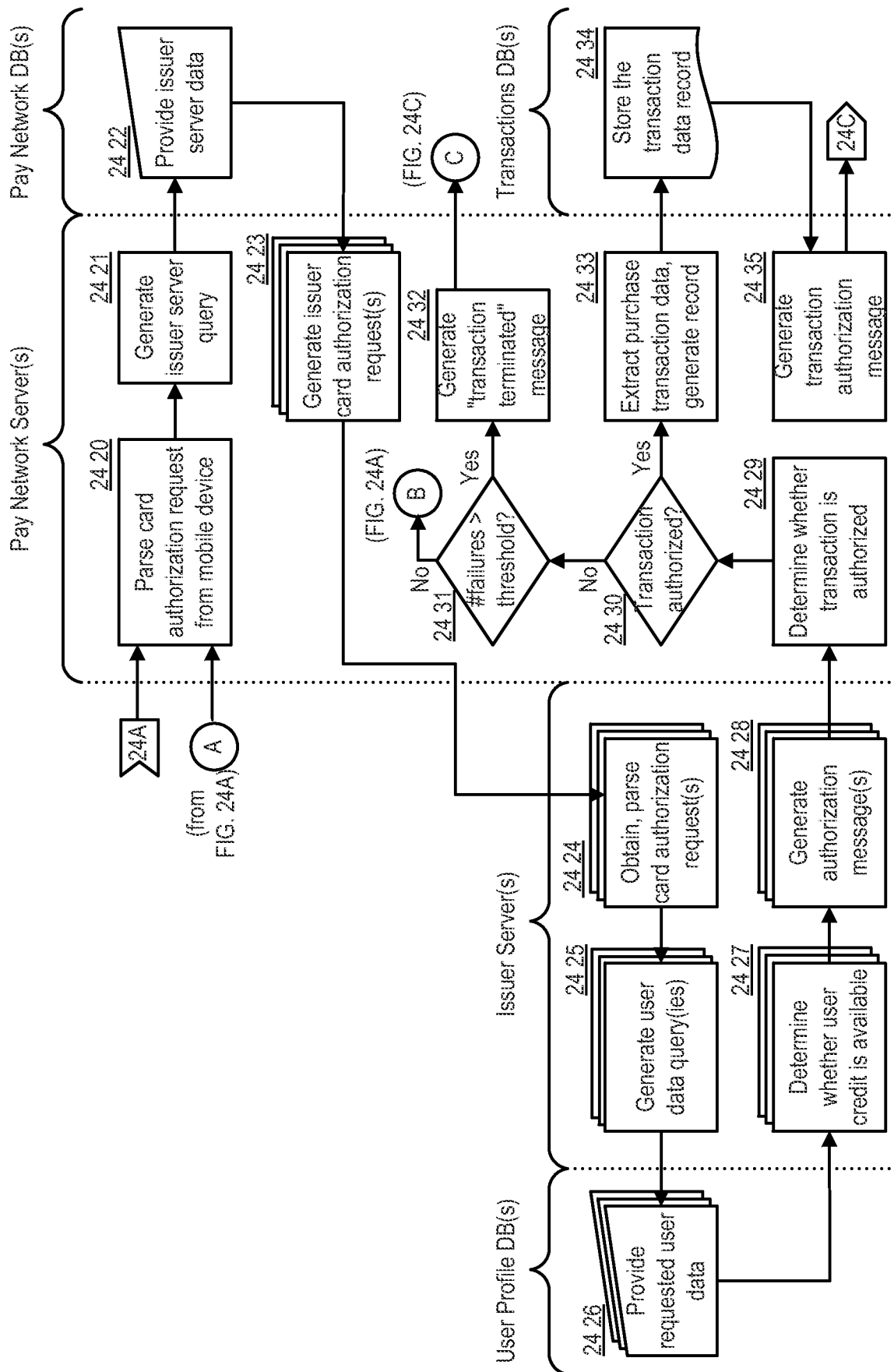


FIGURE 24B

Example: Card-Based Transaction Execution ("CTE") component 2400

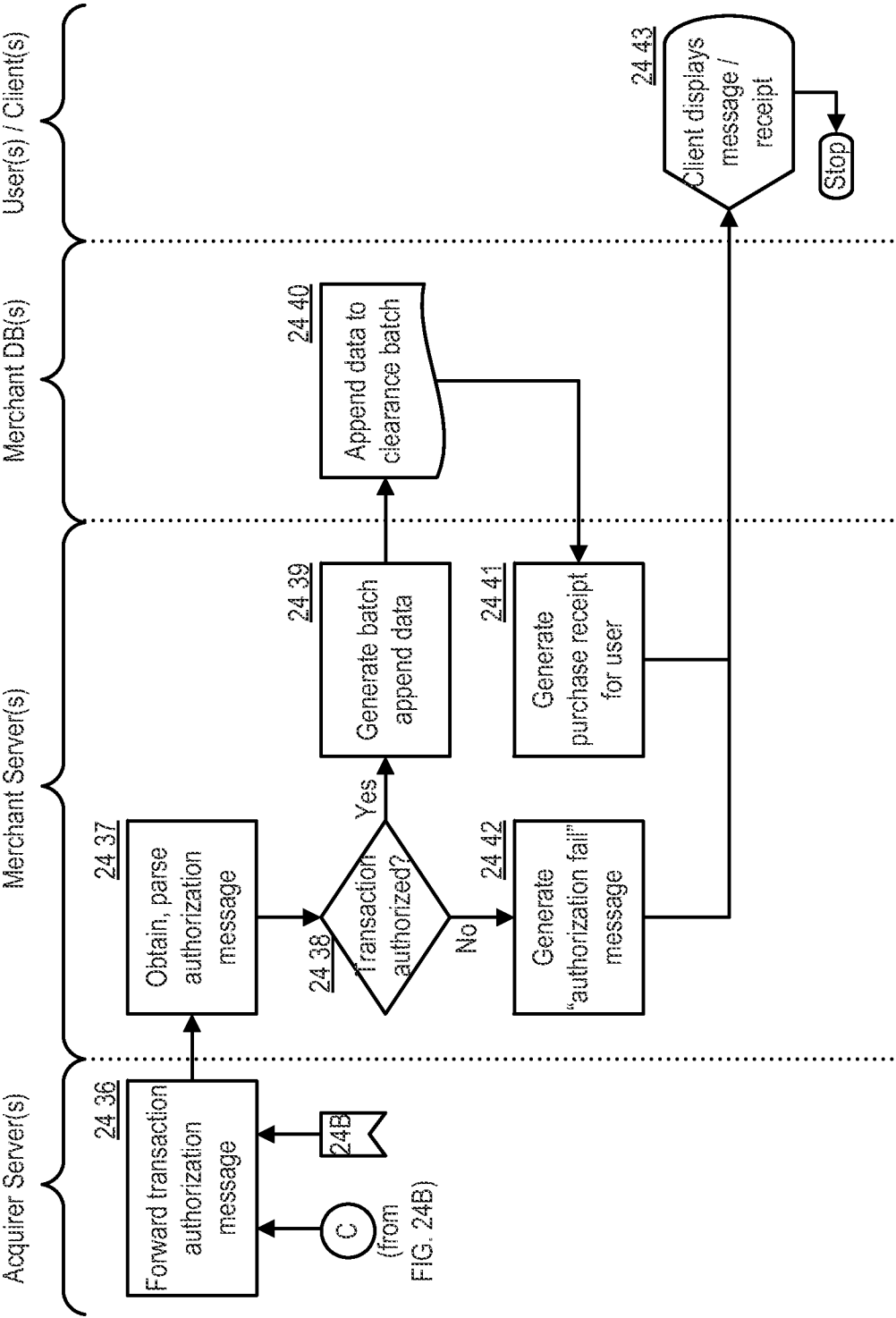


FIGURE 24C

Example: Card-Based Transaction Execution ("CTE") component 2400

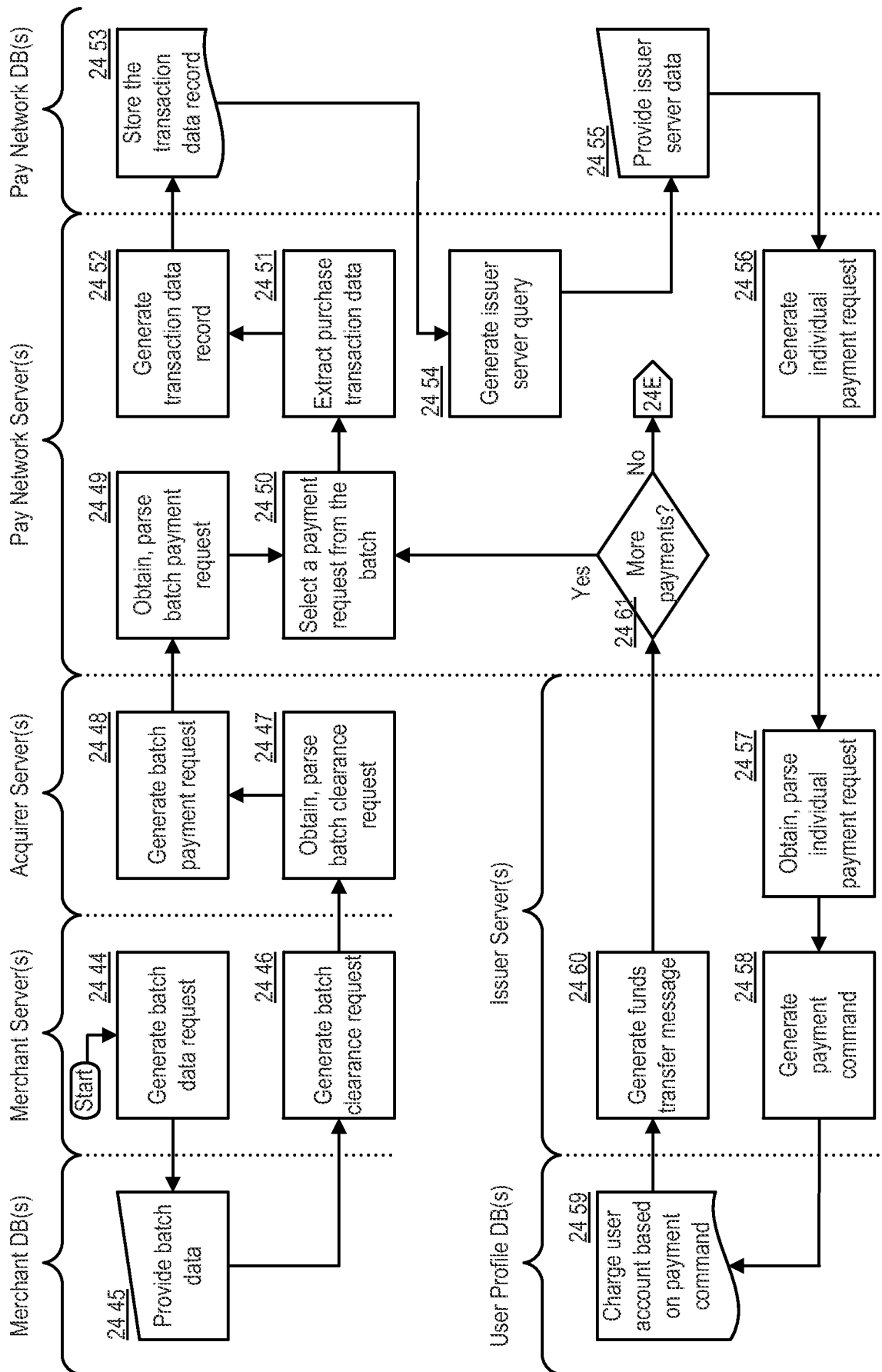


FIGURE 24D

Example: Card-Based Transaction Execution ("CTE") component 2400

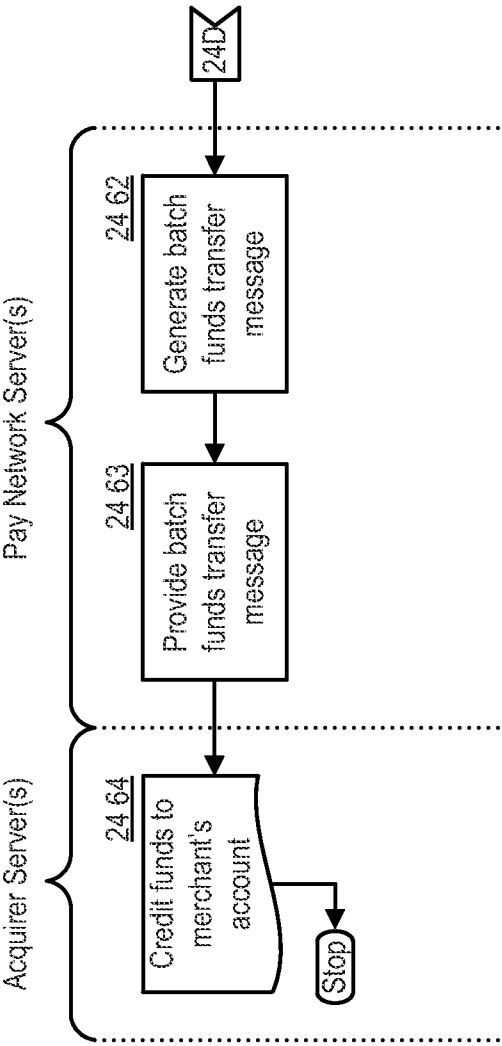
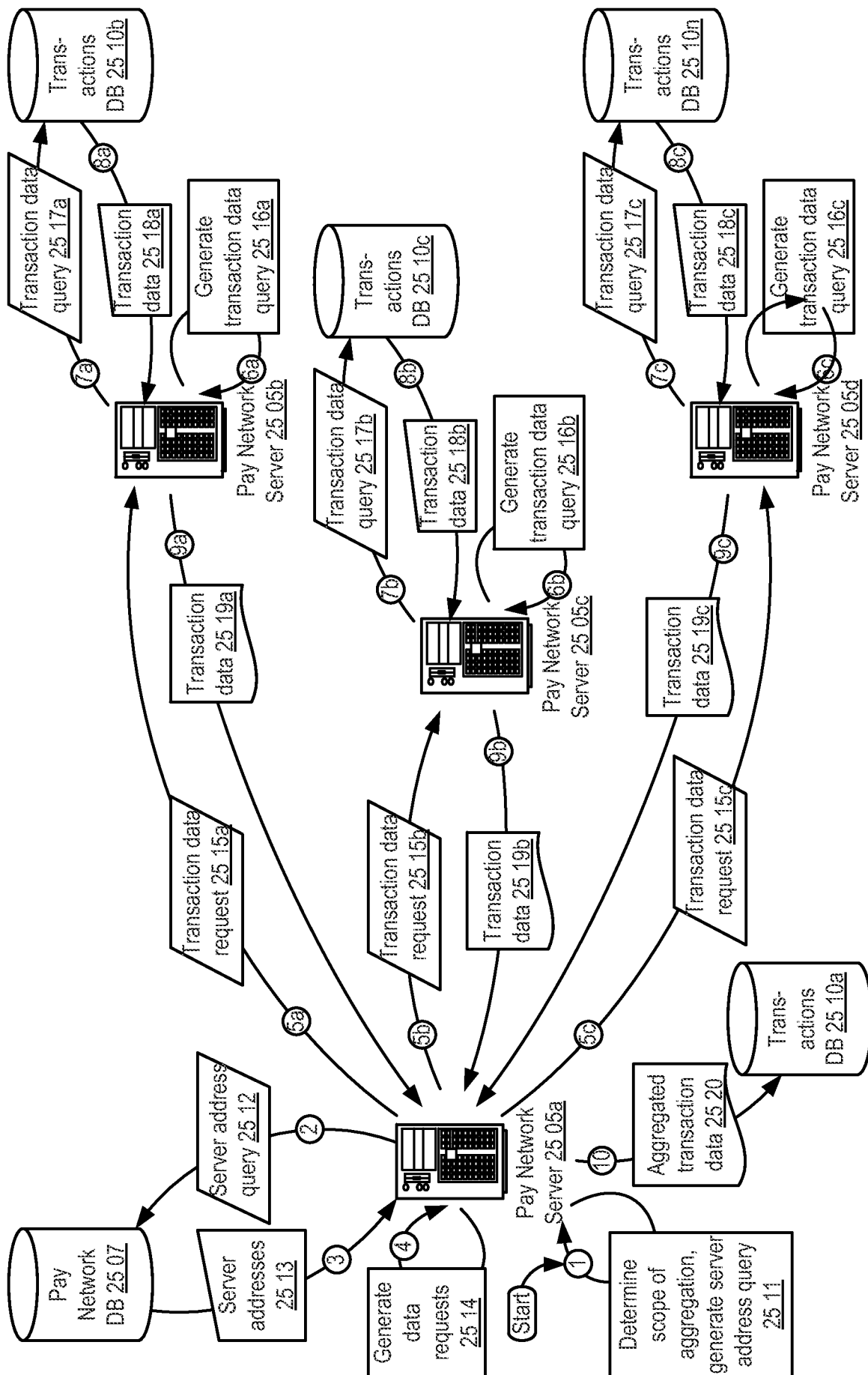


FIGURE 24E

Example: Card-Based Transaction Execution ("CTE") component 2400



Example: Transaction Data Aggregation

FIGURE 25

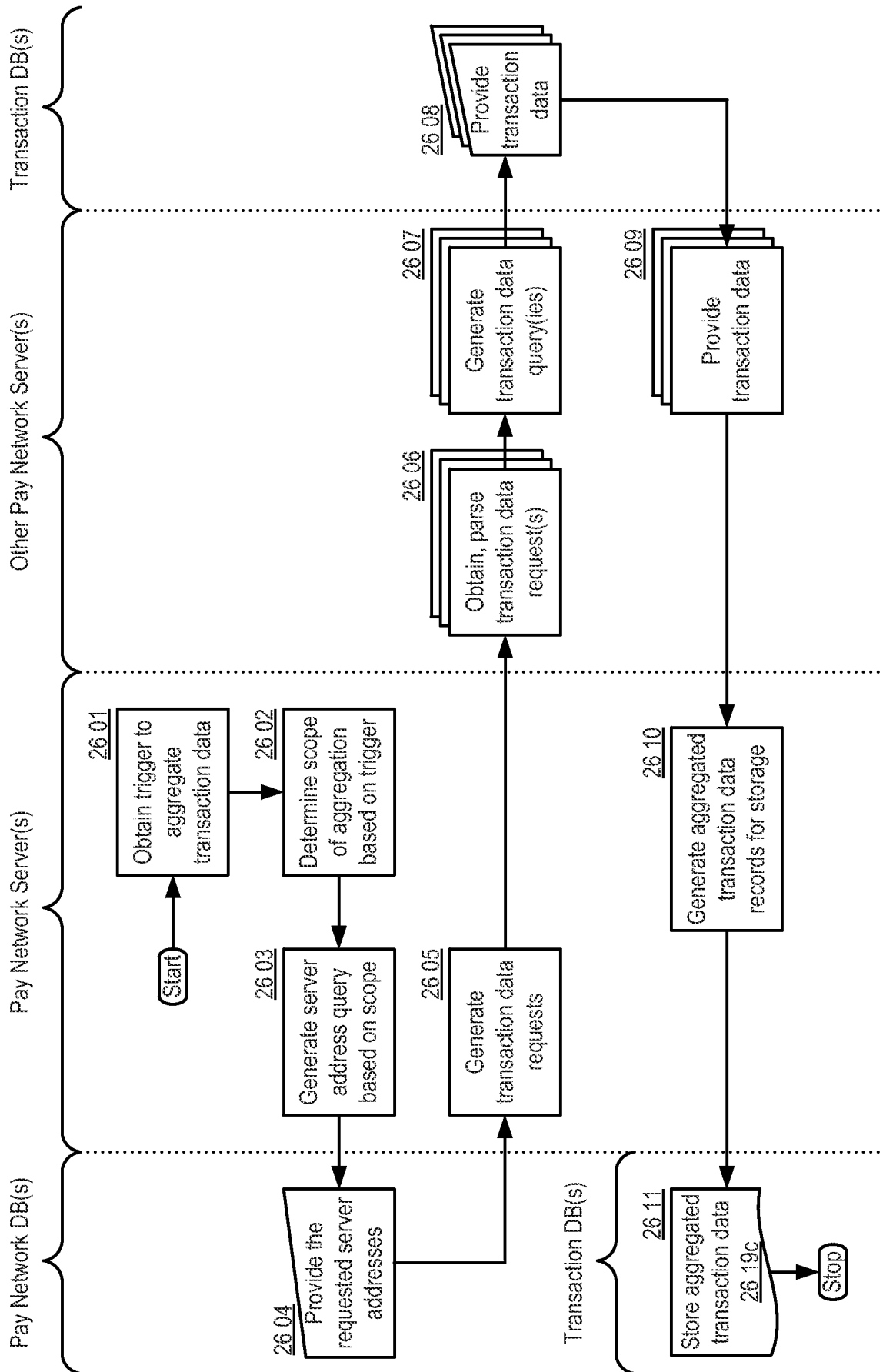


FIGURE 26 Example: Transaction Data Aggregation (TDA) component 600

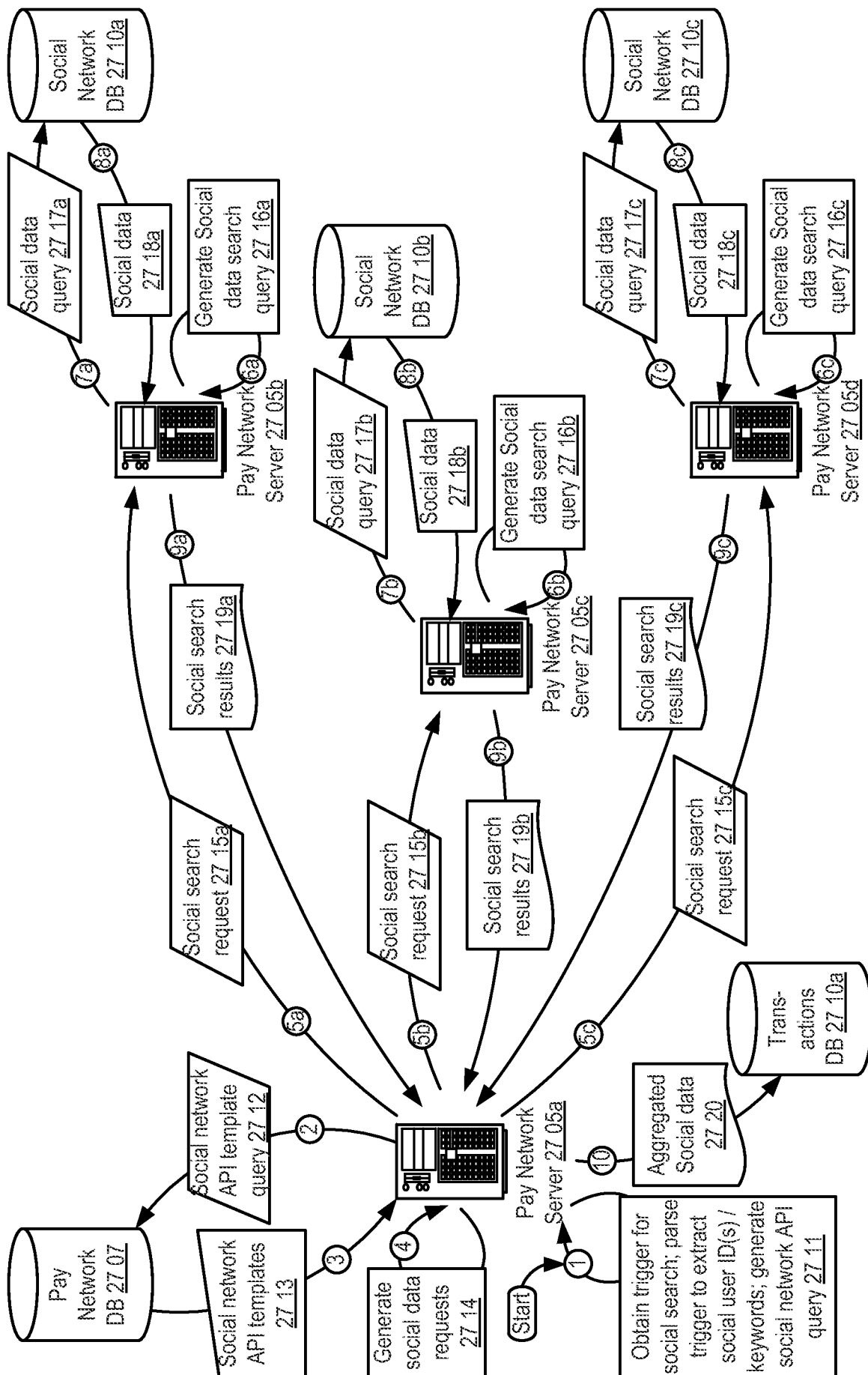


FIGURE 27

Example: Social Data Aggregation

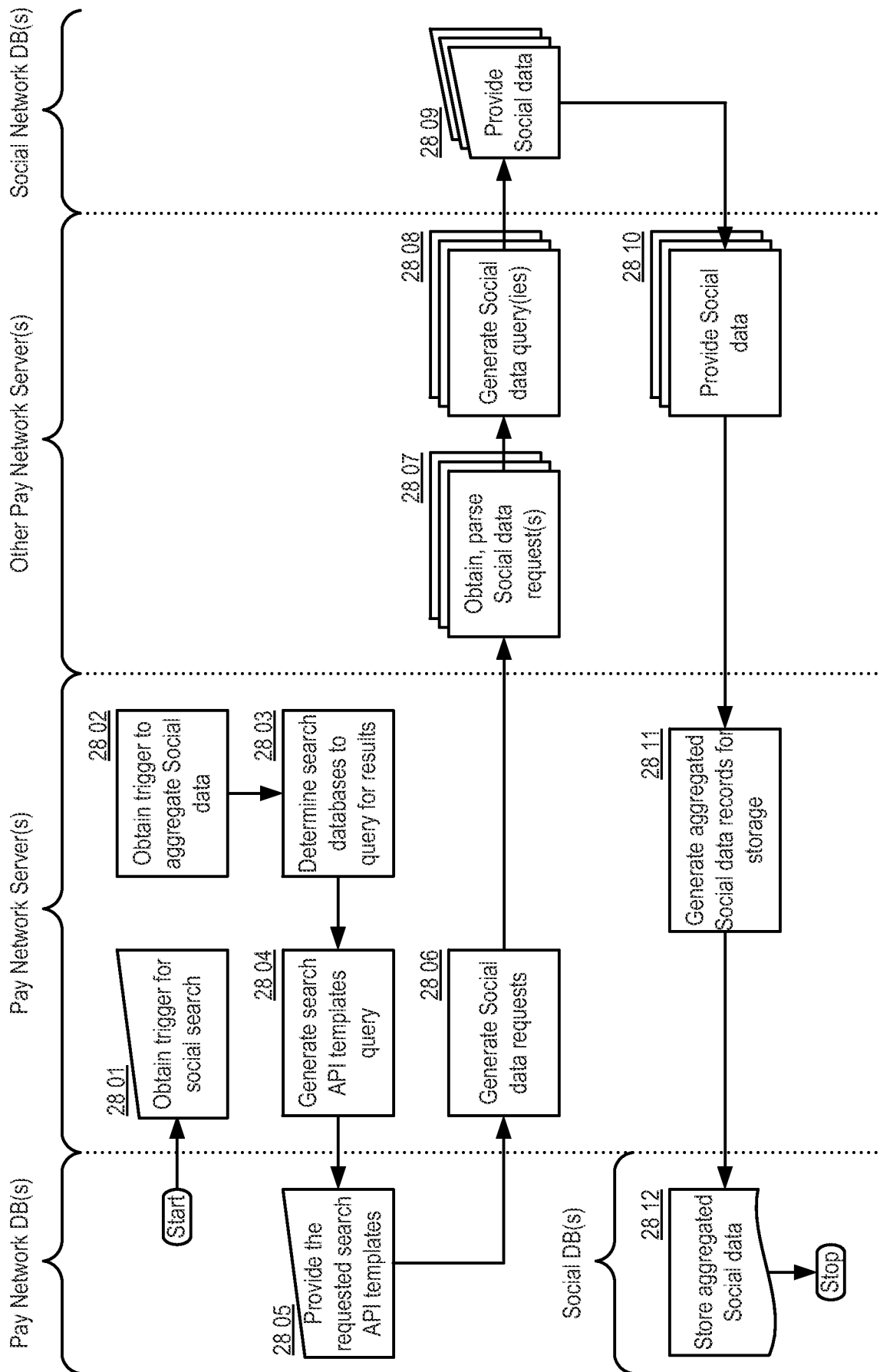


FIGURE 28

Example: Social Data Aggregation ("SDA") component 2800

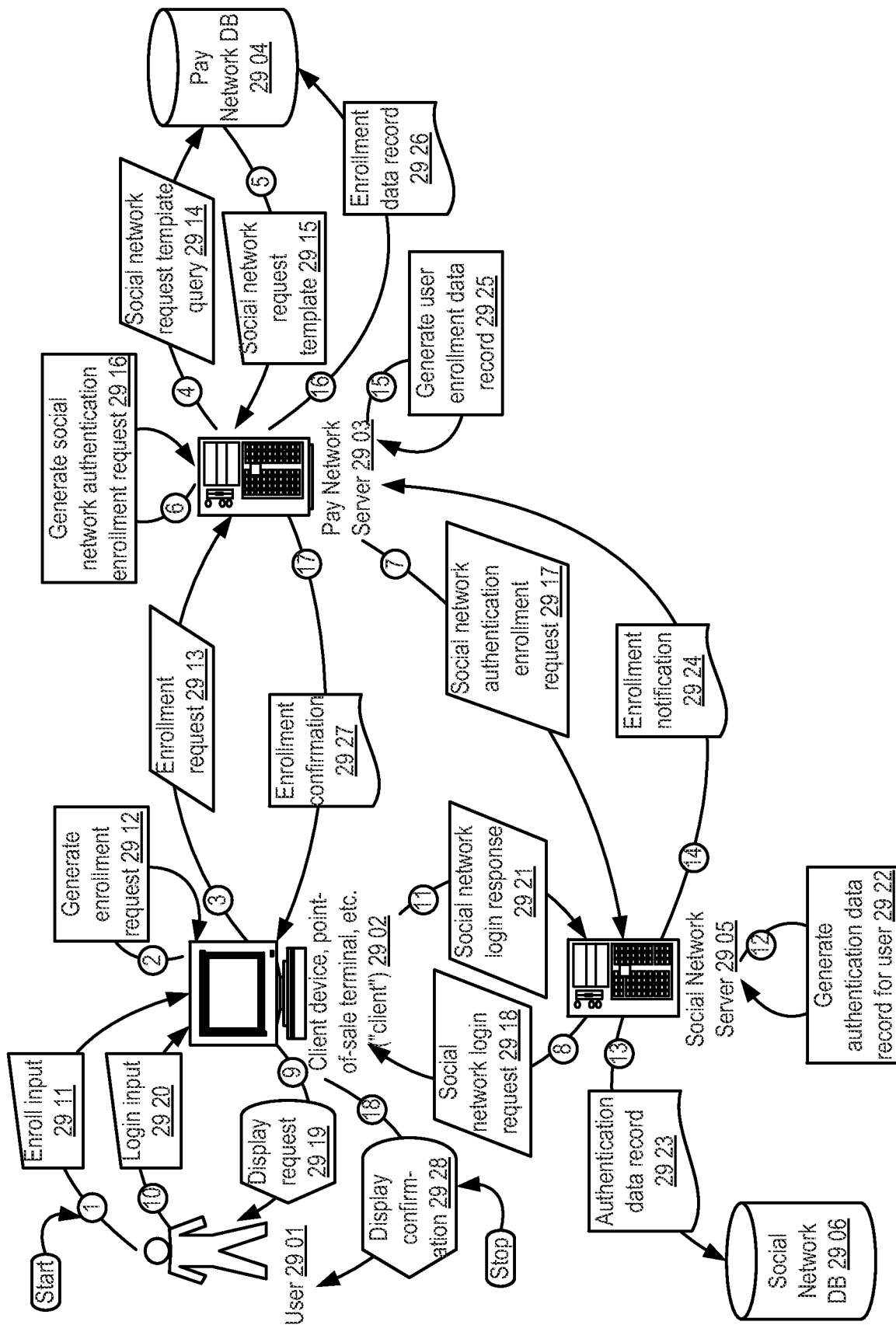


FIGURE 29

Example: Value-Add Service Enrollment

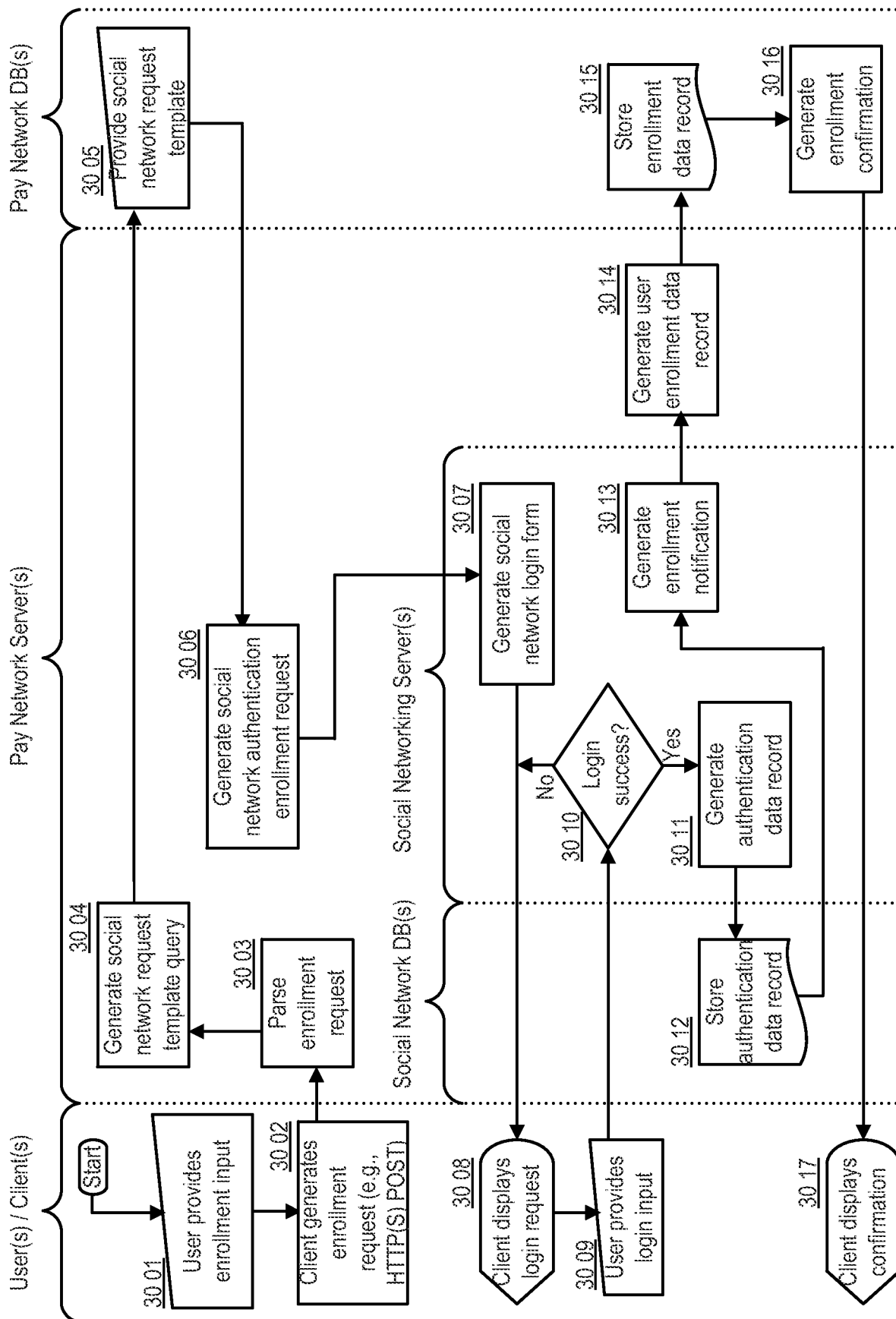


FIGURE 30

Example: Value-Add Service Enrollment ("VASE") component 3000

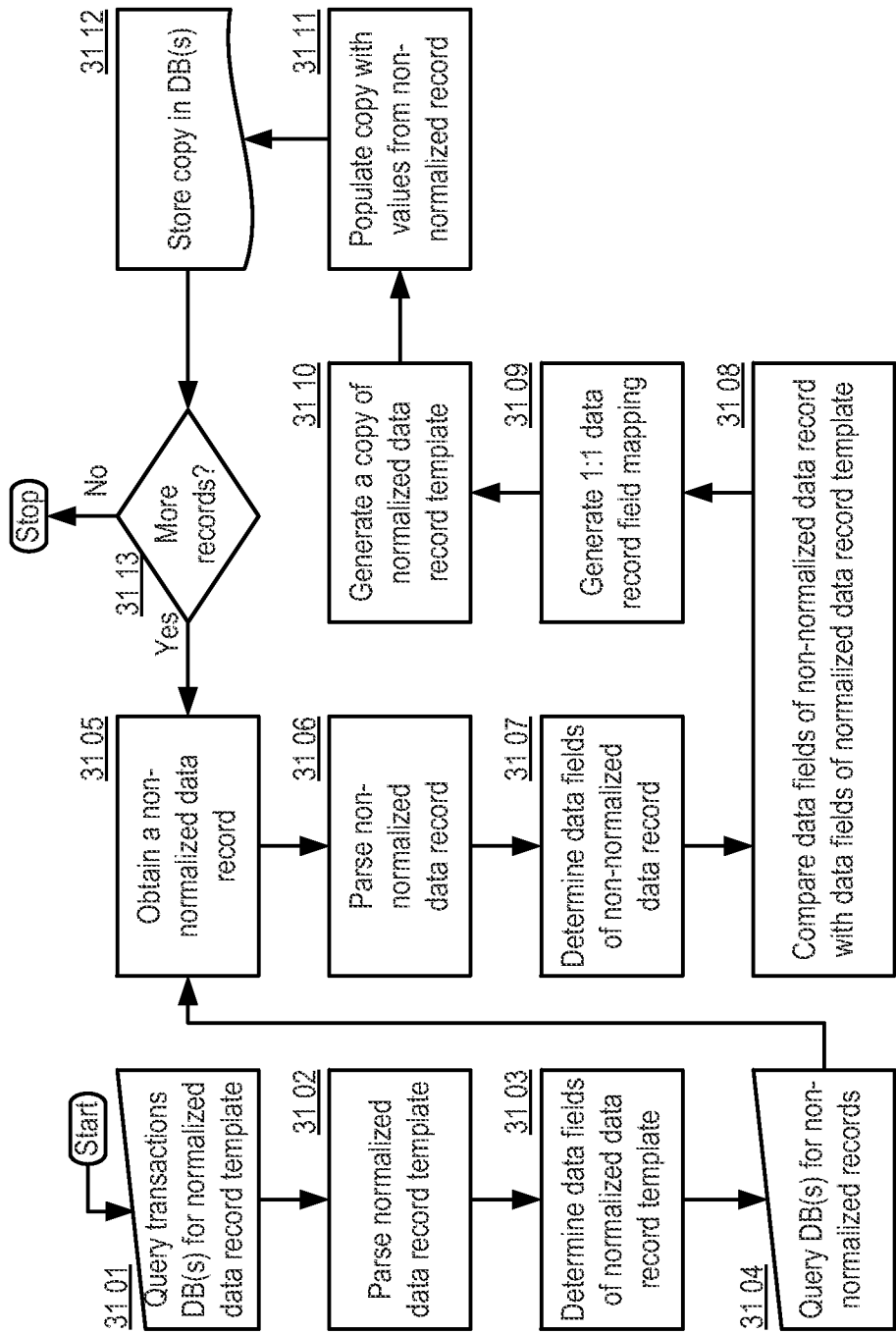


FIGURE 31A Example: Aggregated Data Record Normalization ("ADRN") component 3100

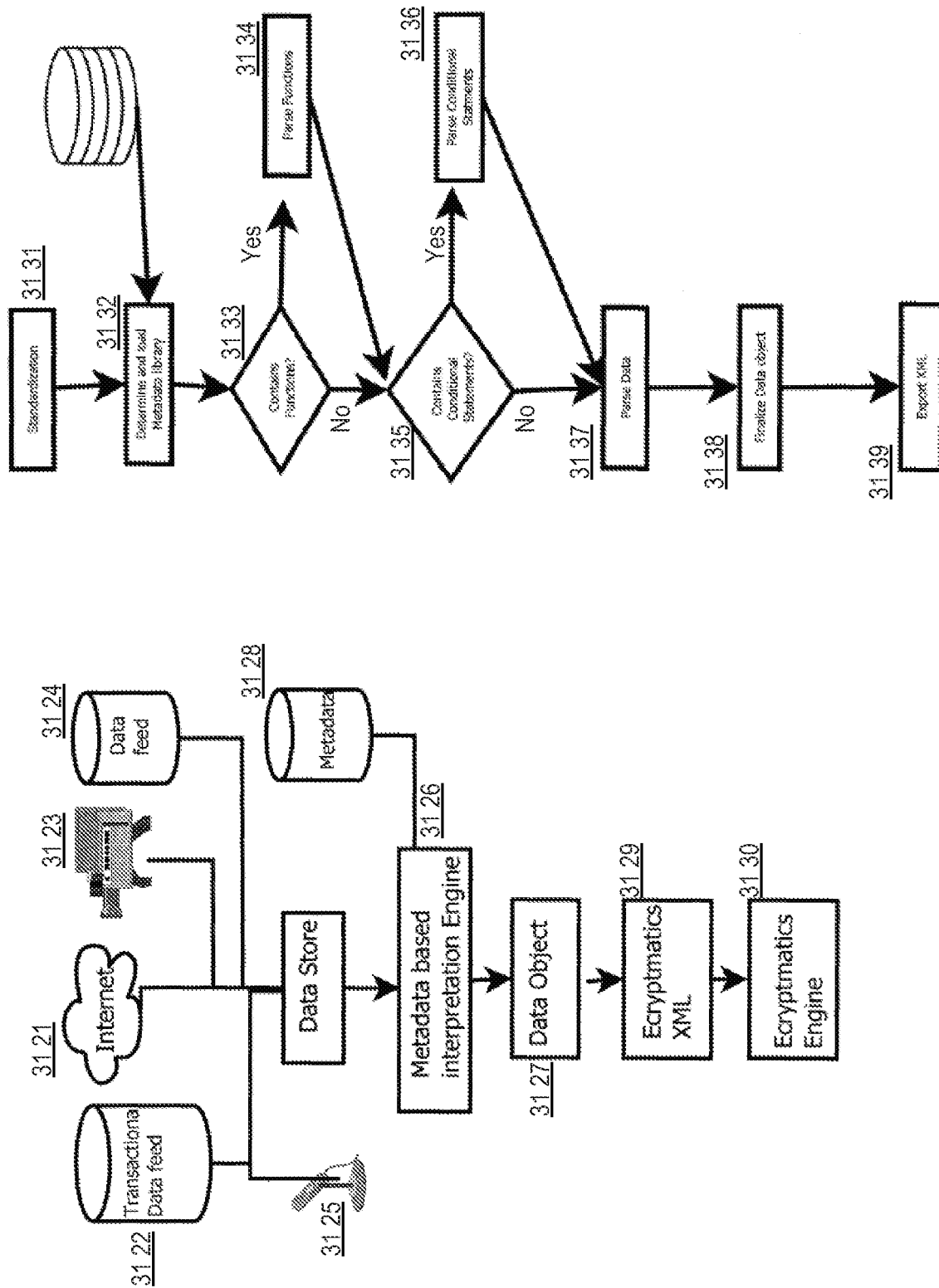


FIGURE 31B

Example: Aggregated Data Record Normalization ("ADRN") component 3100

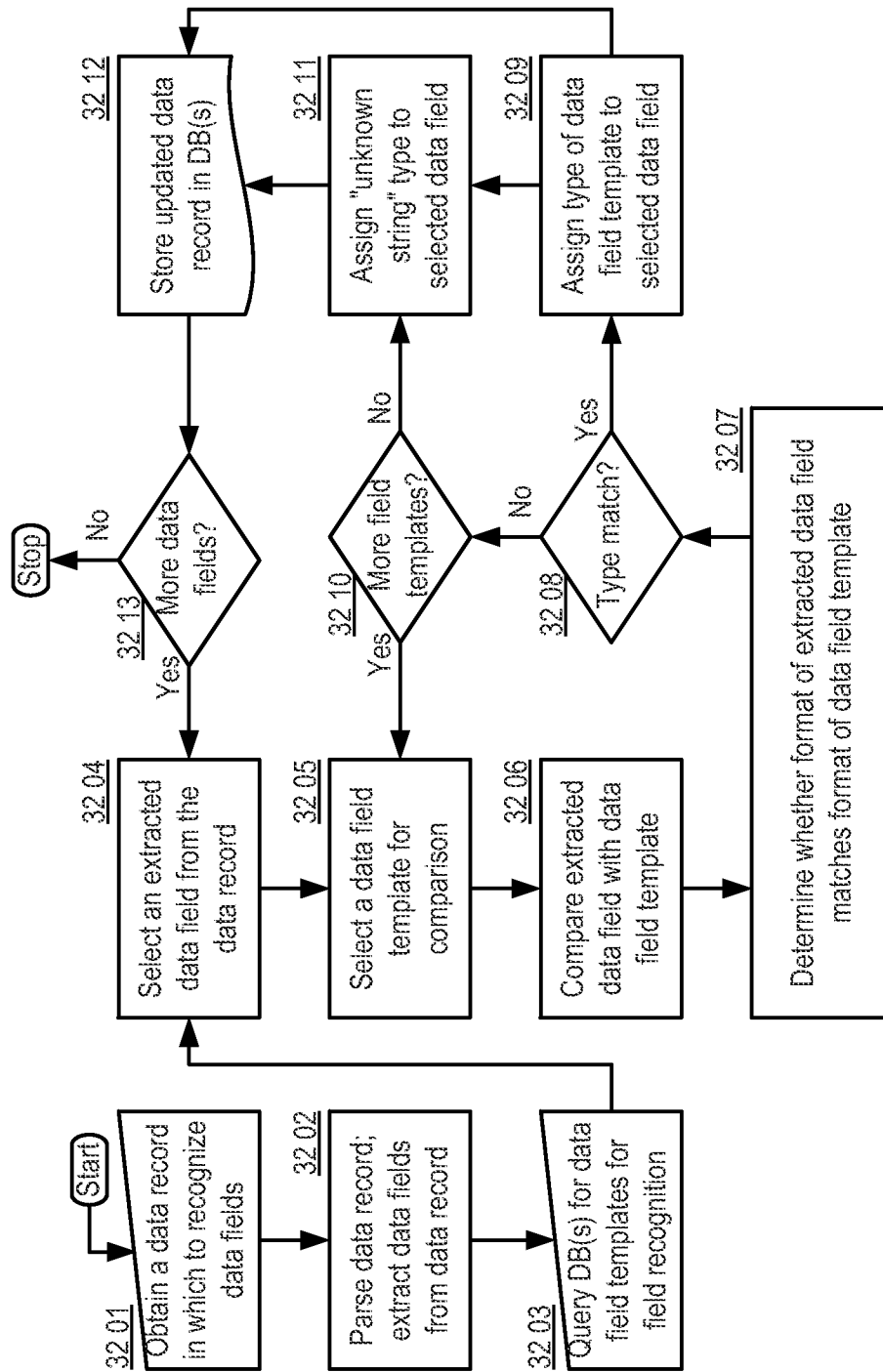


FIGURE 32

Example: Data Field Recognition ("DFR") component 3200

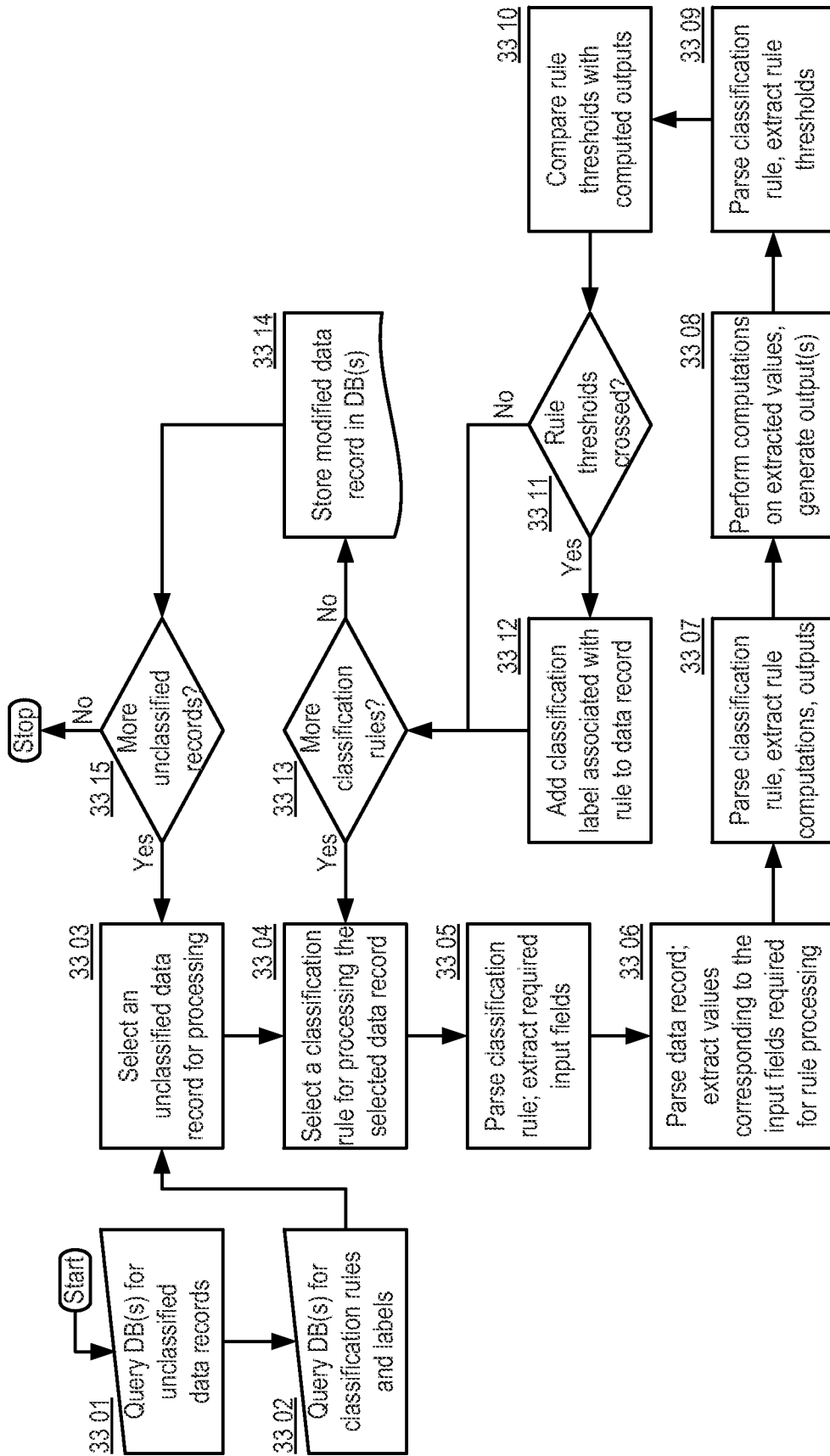


FIGURE 33

Example: Entity Type Classification ("ETC") component 3300

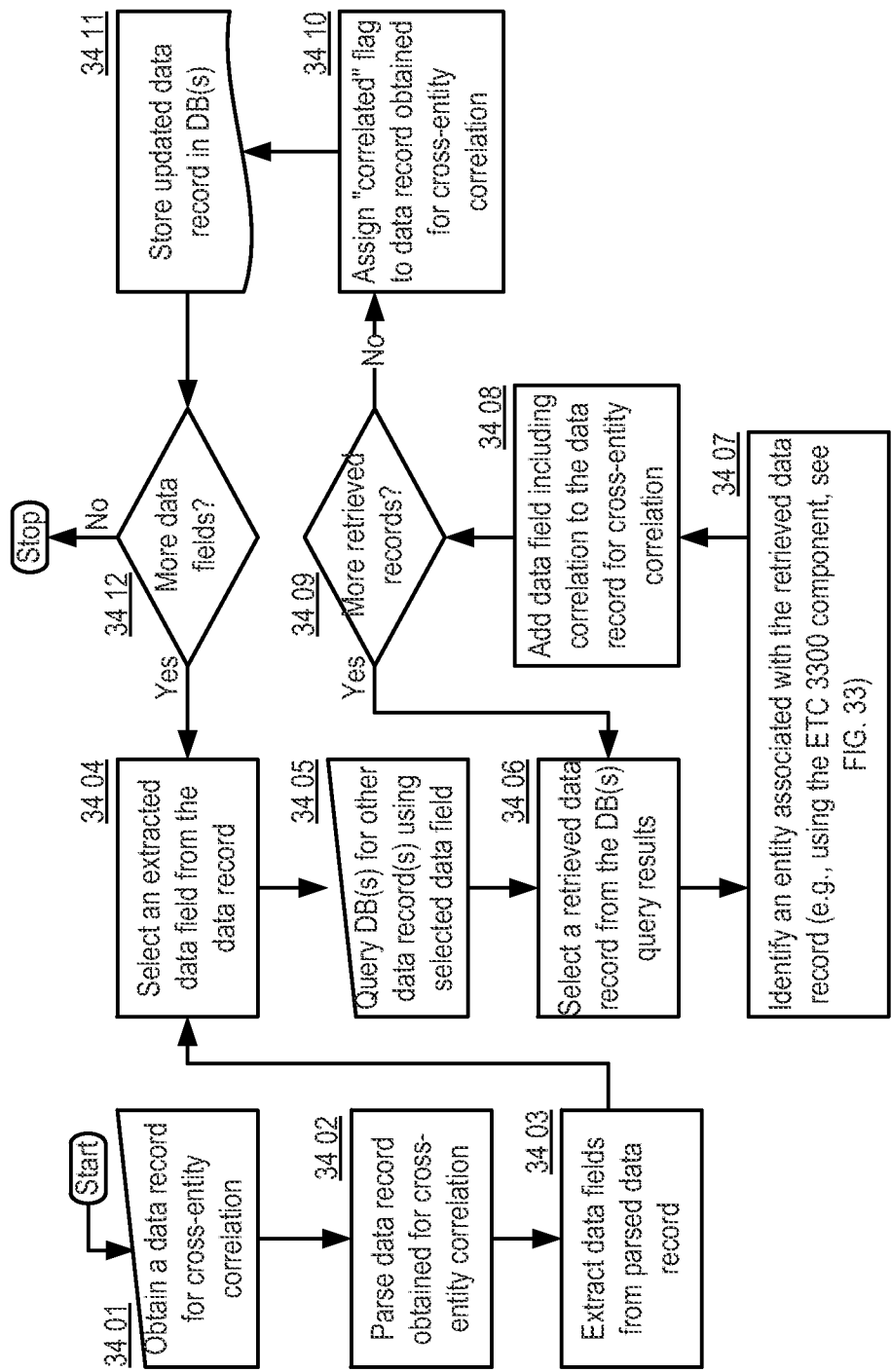


FIGURE 34

Example: Cross-Entity Correlation ("CEC") component 3400

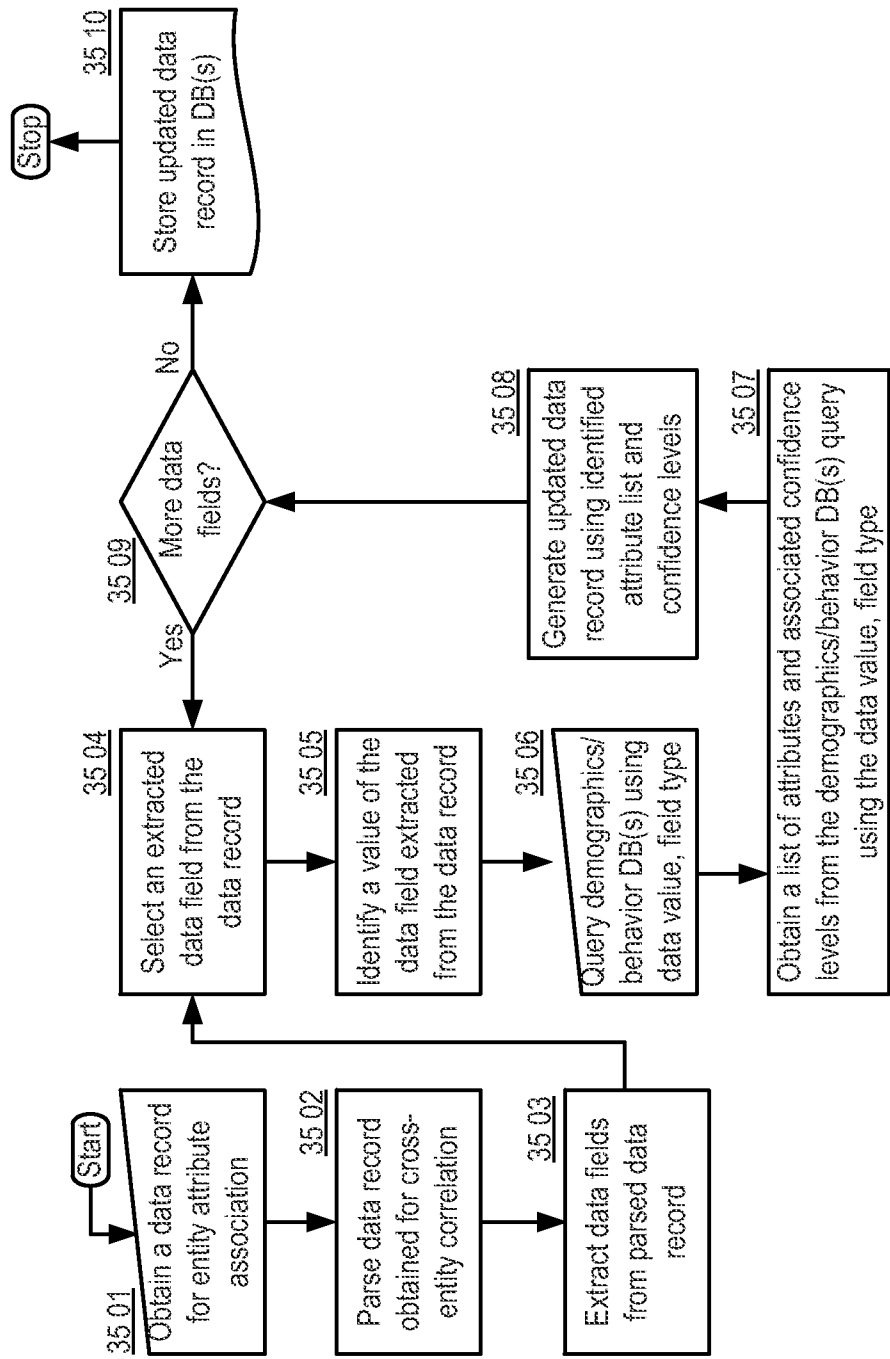


FIGURE 35

Example: Entity Attribute Association ("EAA") component 3500

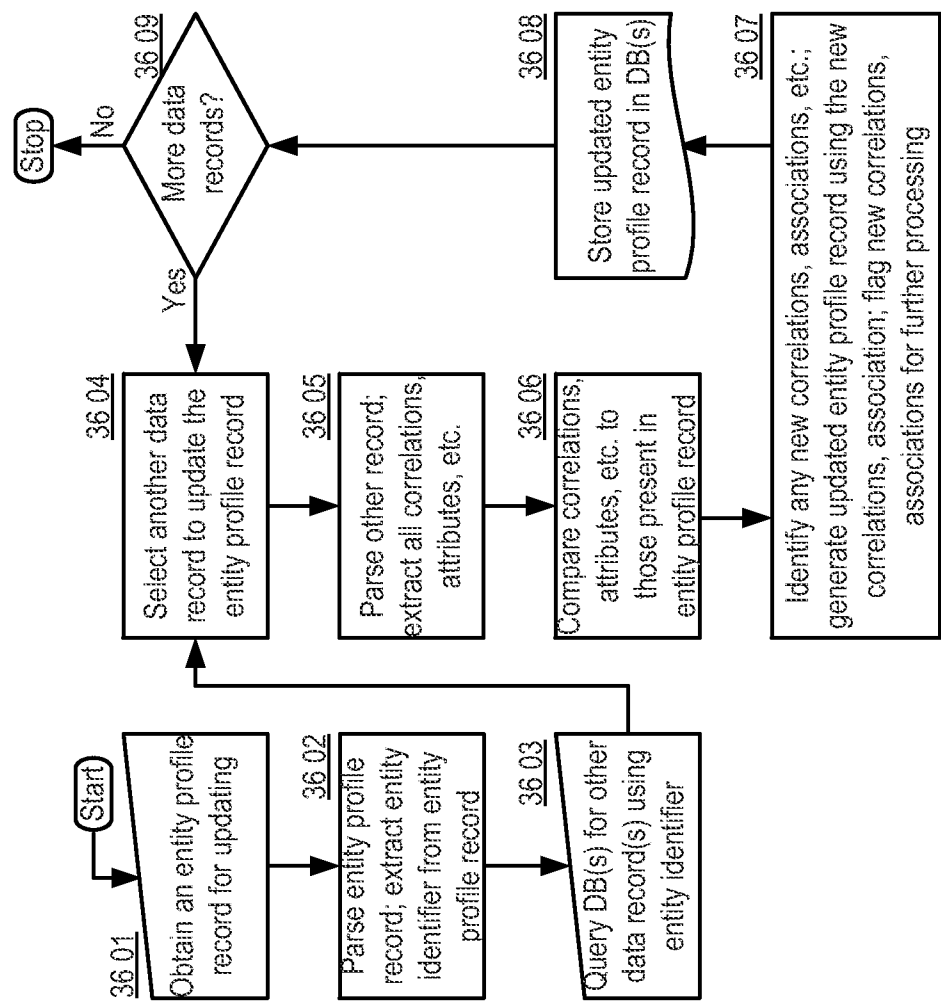


FIGURE 36

Example: Entity Profile-Graph Updating (“EPGU”) component 3600

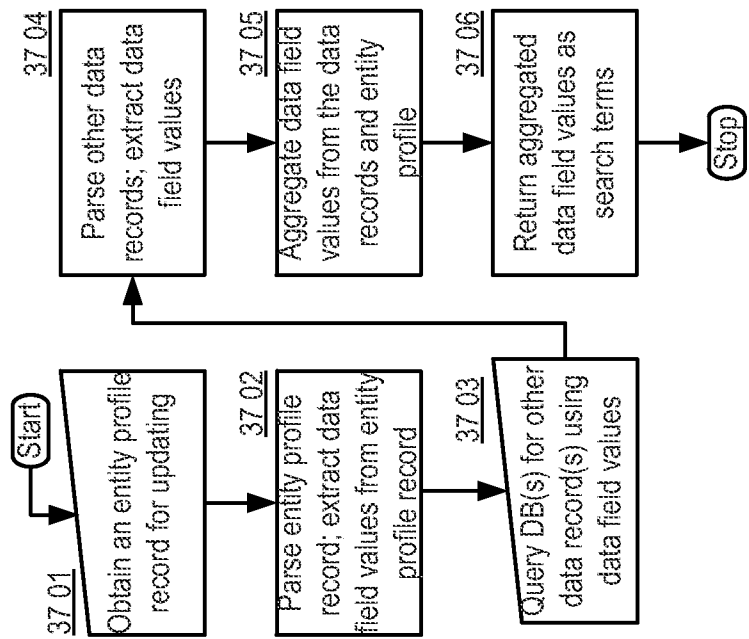


FIGURE 37

Example: Search Term Generation ("STG") component 3700

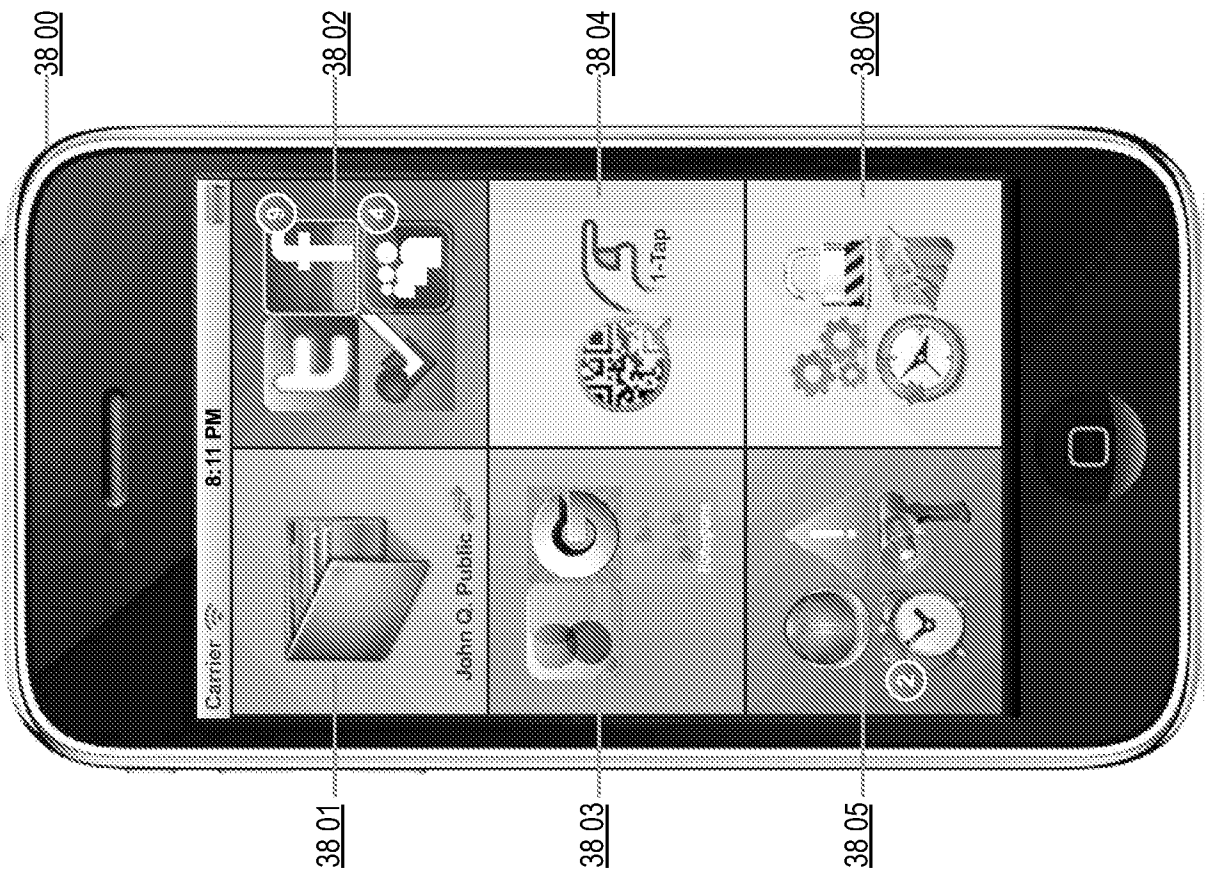


FIGURE 38

Example: Virtual Wallet Mobile App - Feature Overview

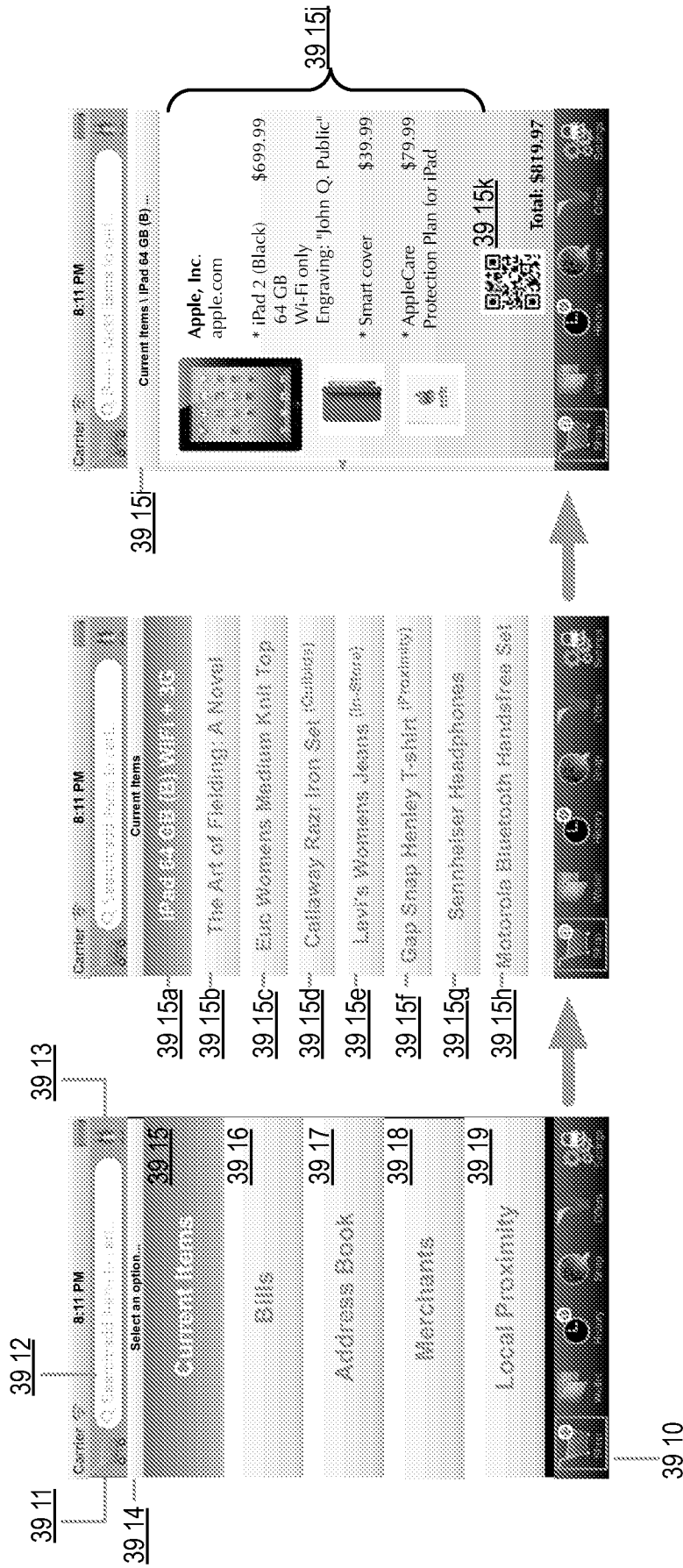


FIGURE 39A

Example: Virtual Wallet Mobile App - Shopping Mode

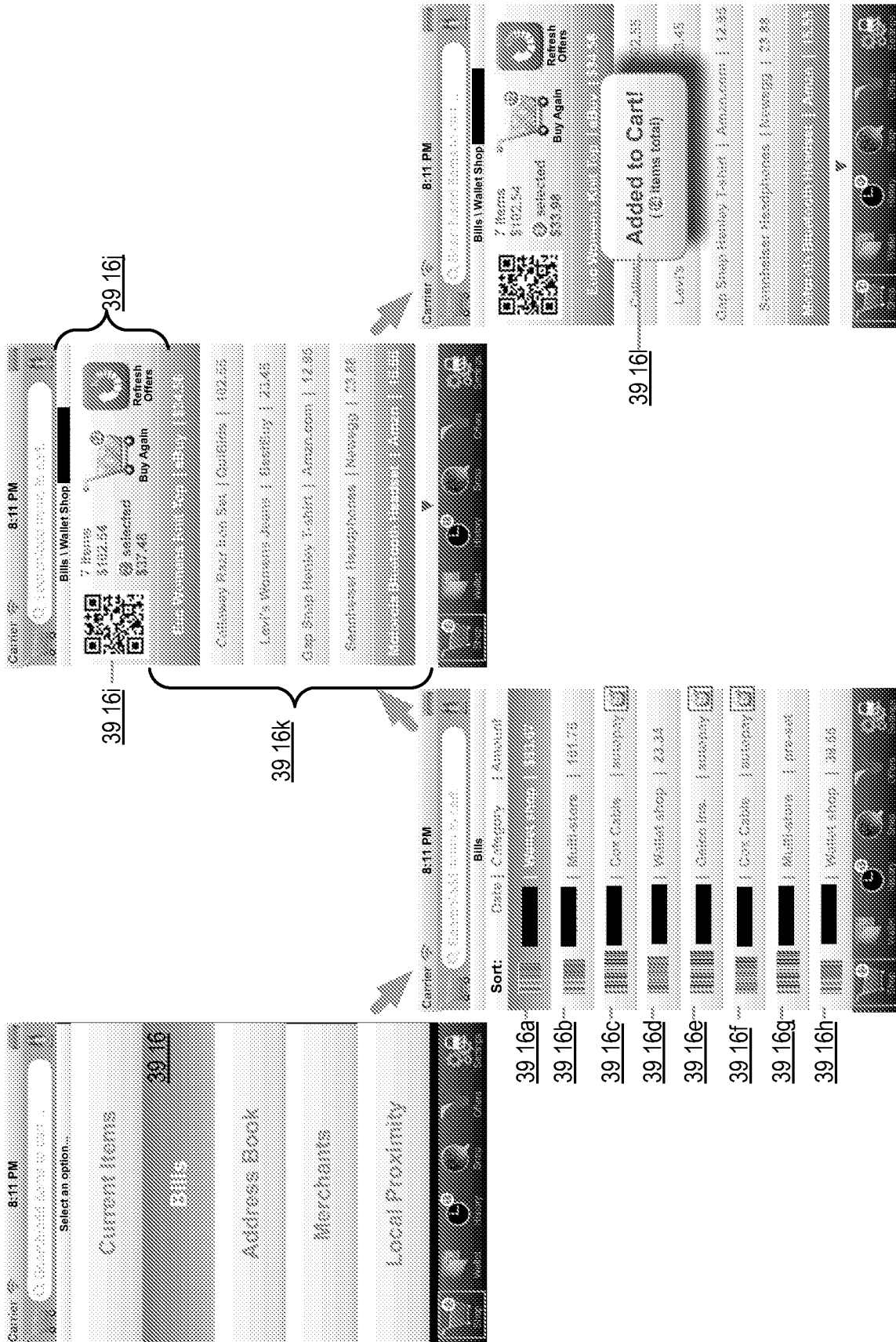


FIGURE 39B

Example: Virtual Wallet Mobile App - Shopping Mode

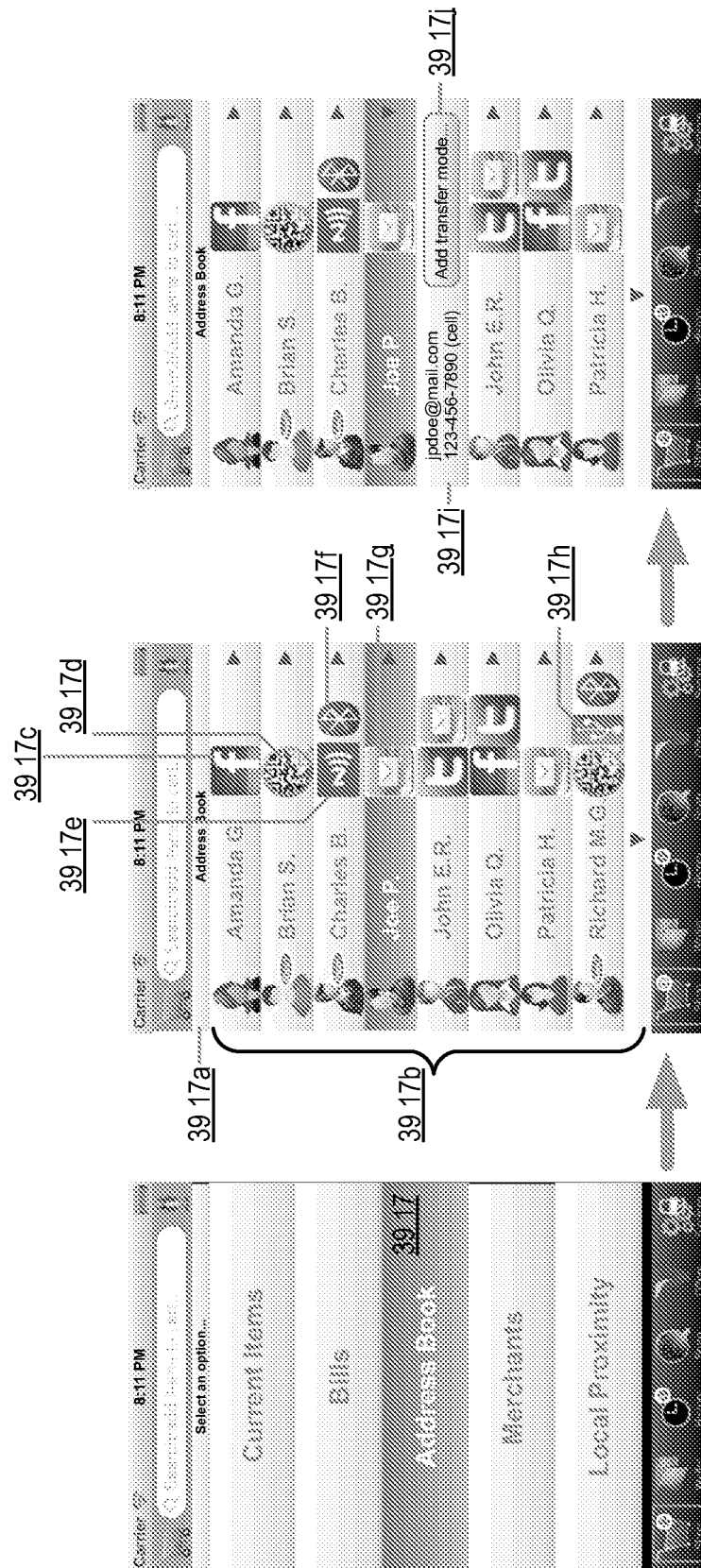


FIGURE 39C

Example: Virtual Wallet Mobile App - Shopping Mode

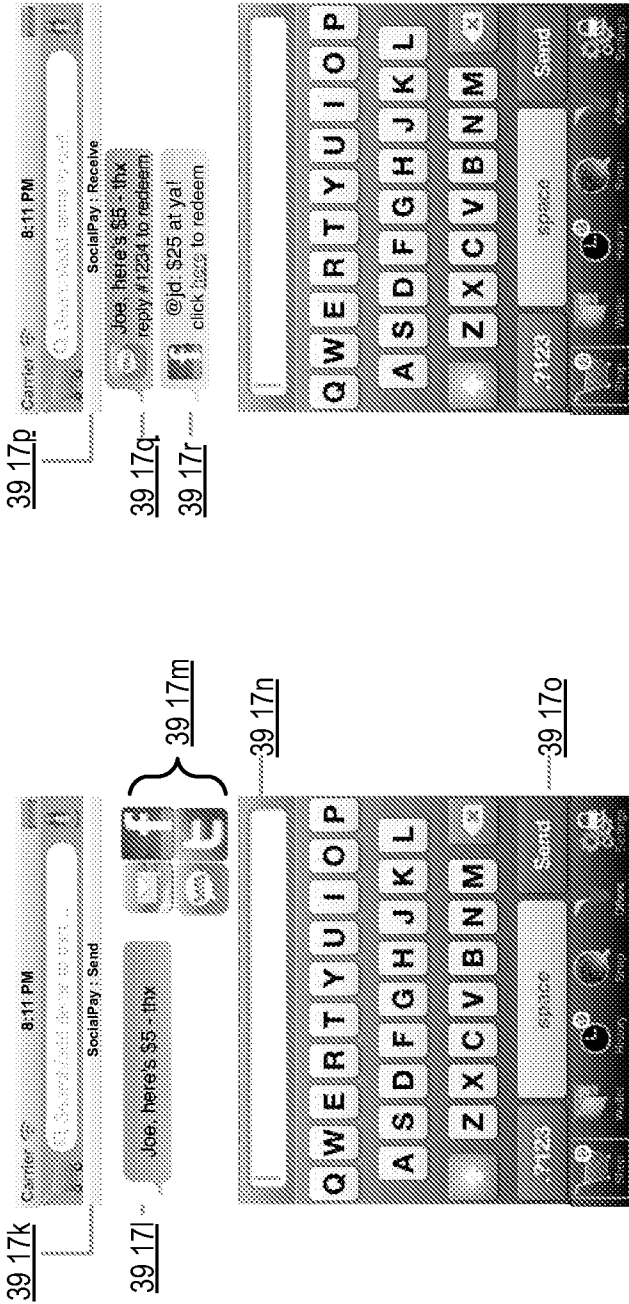


FIGURE 39D

Example: Virtual Wallet Mobile App - SocialPay Mode



FIGURE 39E

Example: Virtual Wallet Mobile App - Shopping Mode



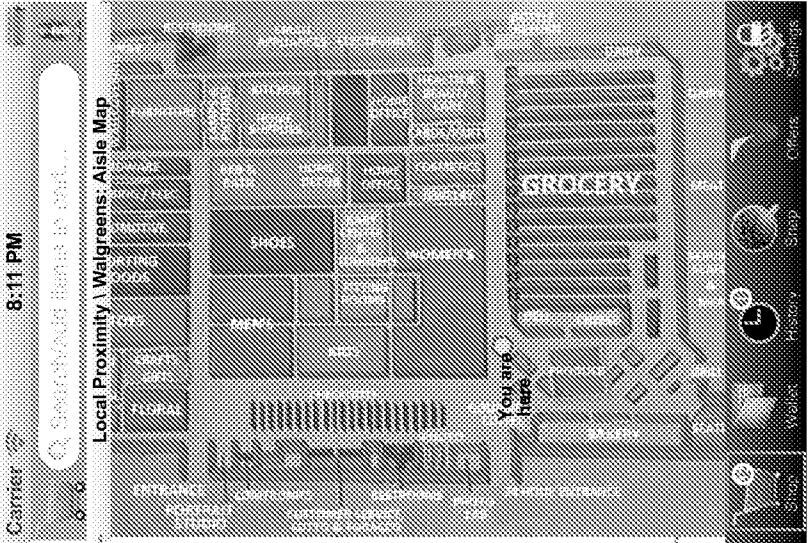
FIGURE 39F

Example: Virtual Wallet Mobile App - Shopping Mode



39 190

39 19n



39 19j

39 19m

Example: Virtual Wallet Mobile App - Shopping Mode

FIGURE 39G

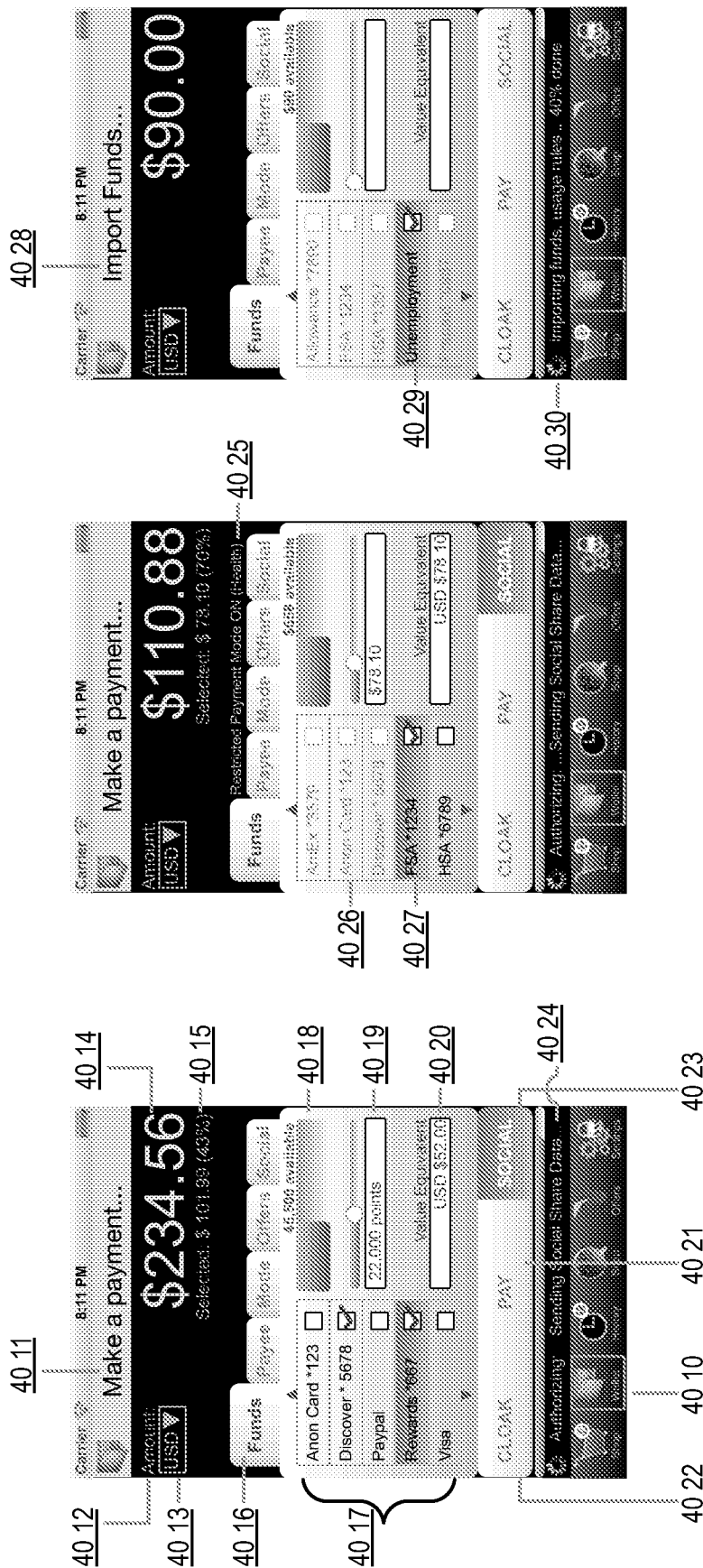


FIGURE 40A

Example: Virtual Wallet Mobile App - Payment Mode

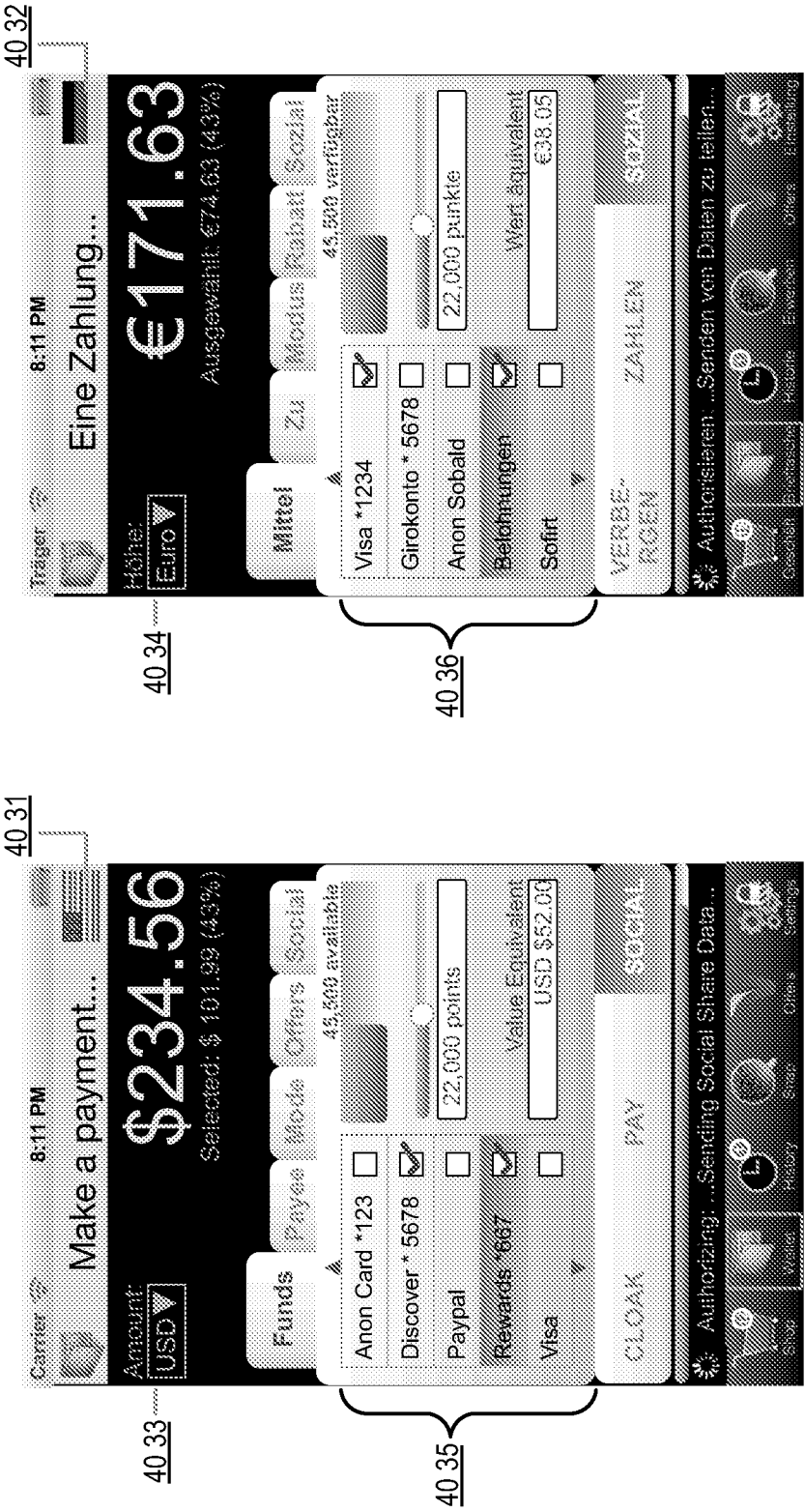
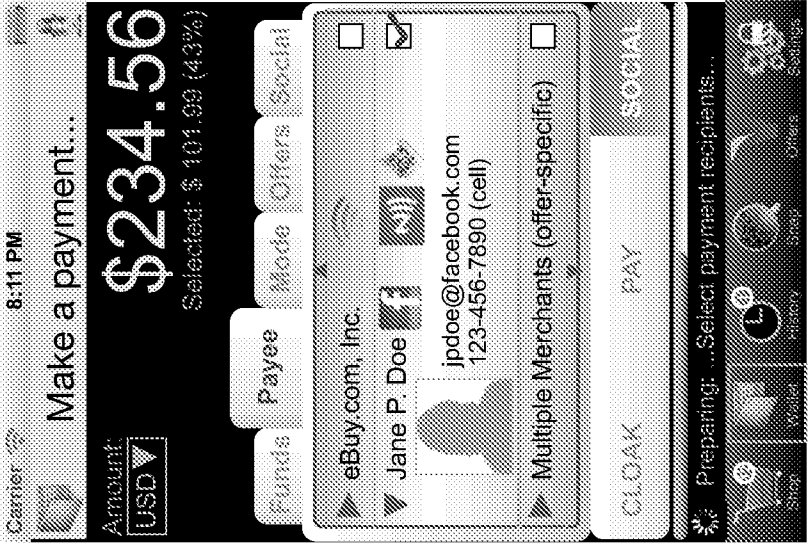
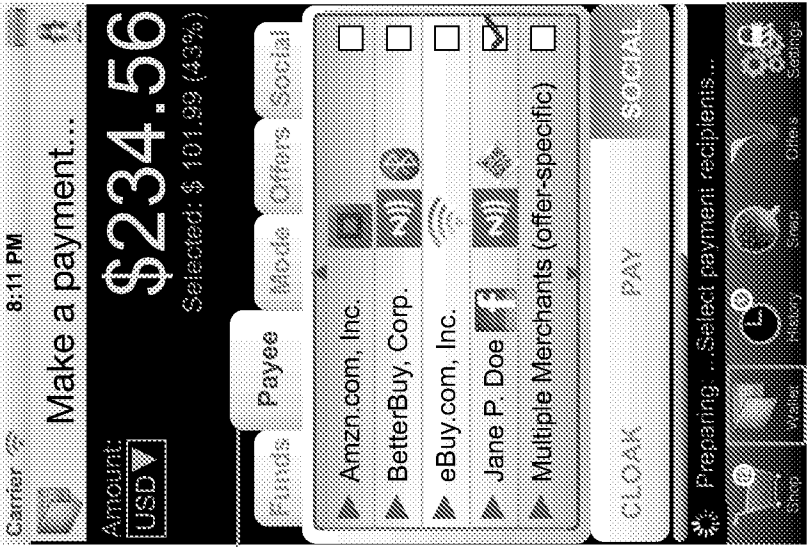


FIGURE 40B

Example: Virtual Wallet Mobile App - Dynamic Payment Optimization



40 39



40 37

40 38

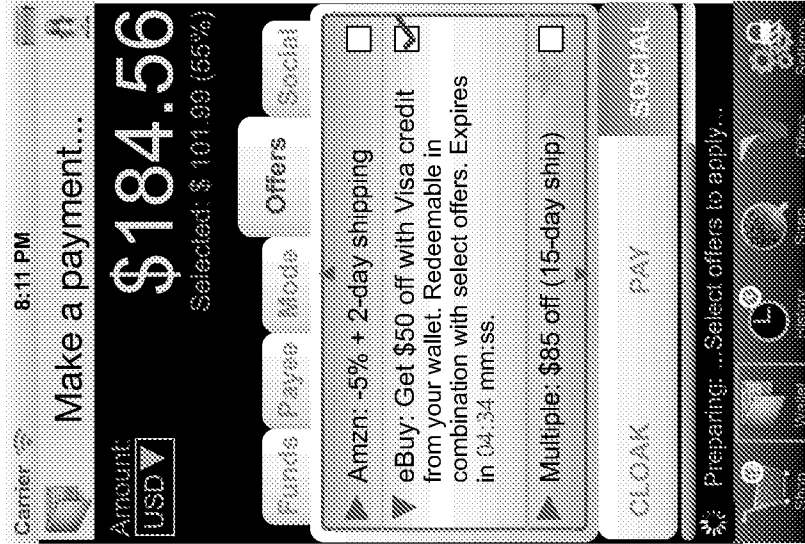
Example: Virtual Wallet Mobile App

FIGURE 40C

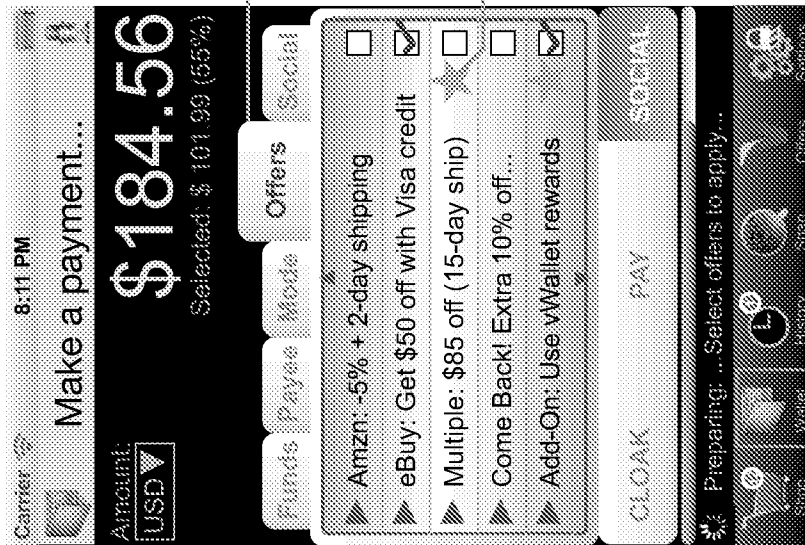


FIGURE 40D

Example: Virtual Wallet Mobile App



40 54



40 52

40 51

40 53

FIGURE 40E

Example: Virtual Wallet Mobile App

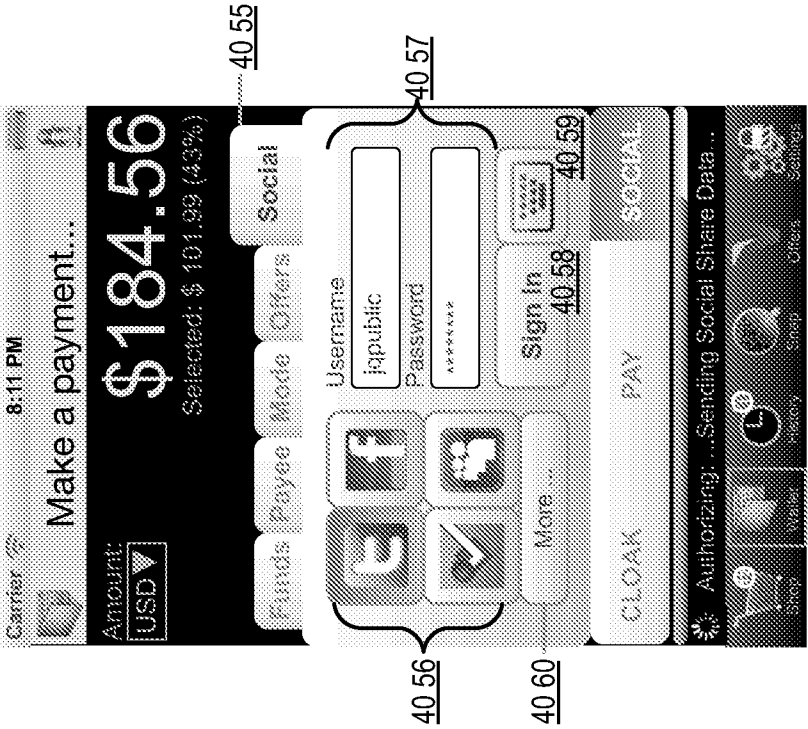


FIGURE 40F

Example: Virtual Wallet Mobile App

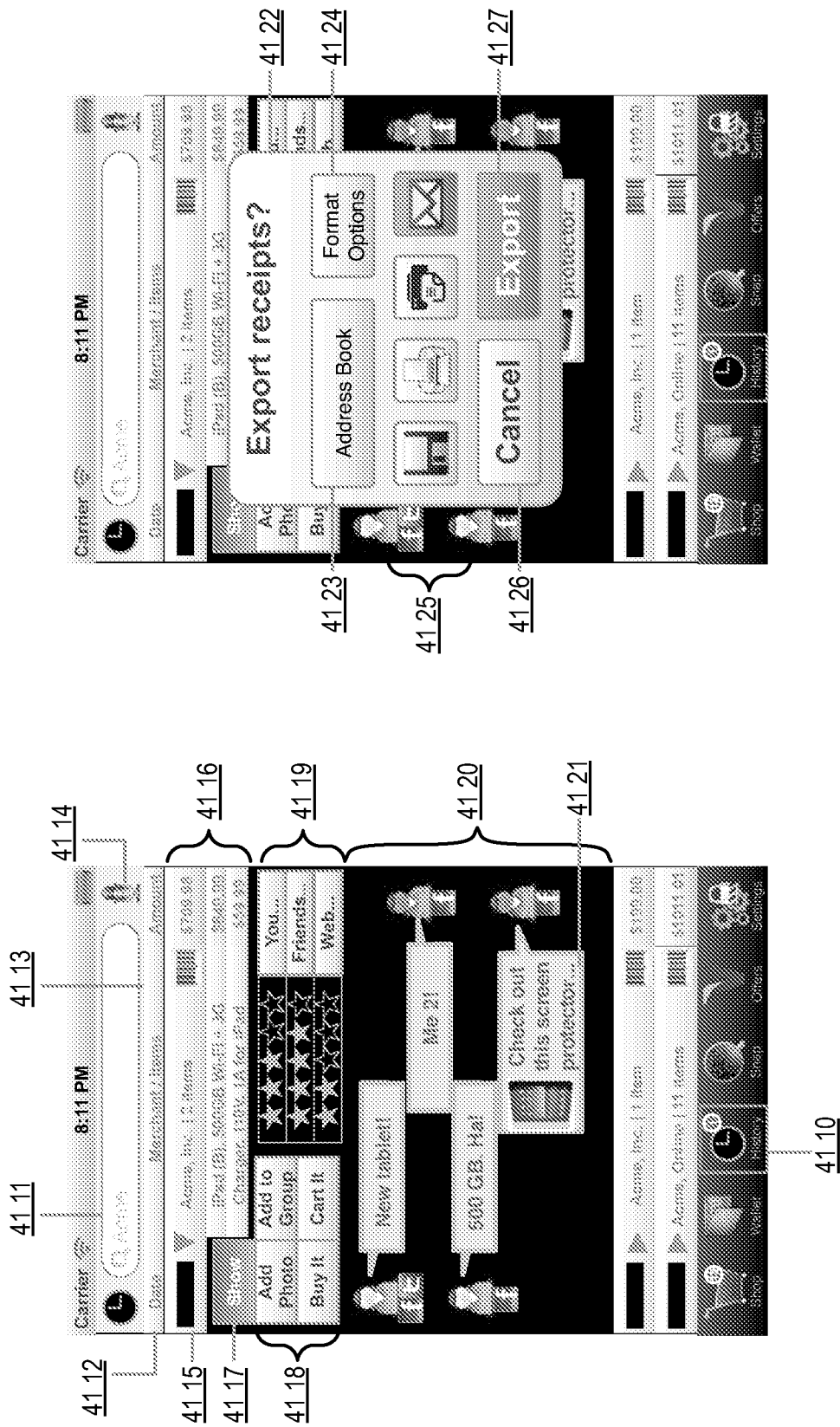


FIGURE 41

Example: Virtual Wallet Mobile App - History

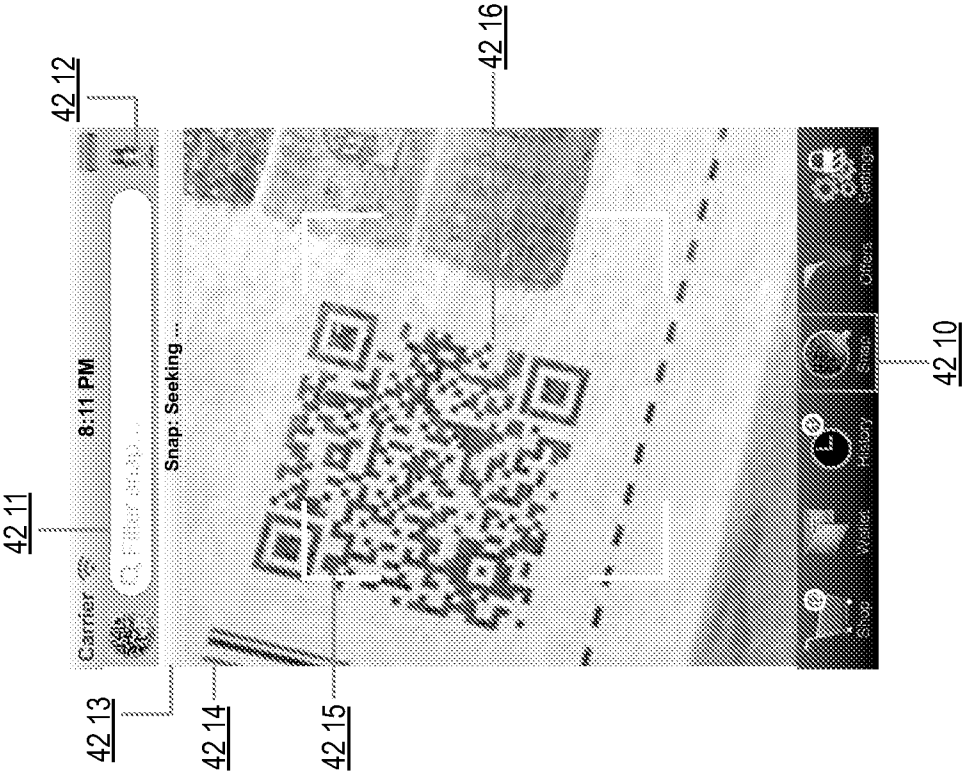


FIGURE 42A

Example: Virtual Wallet Mobile App - Snap Mode



FIGURE 42B

Example: Virtual Wallet Mobile App - Snap Mode

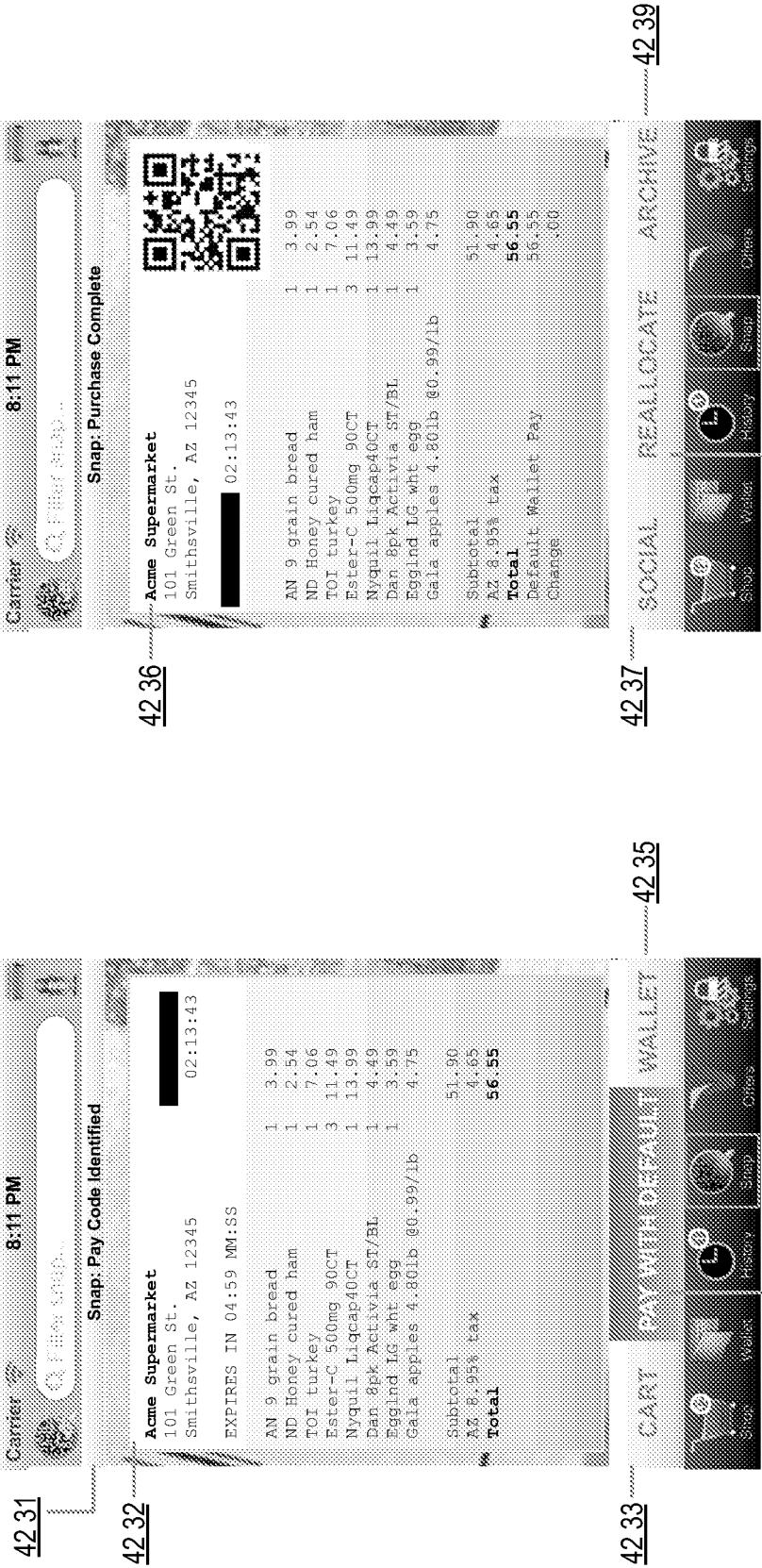


FIGURE 42C

Example: Virtual Wallet Mobile App - Snap Mode

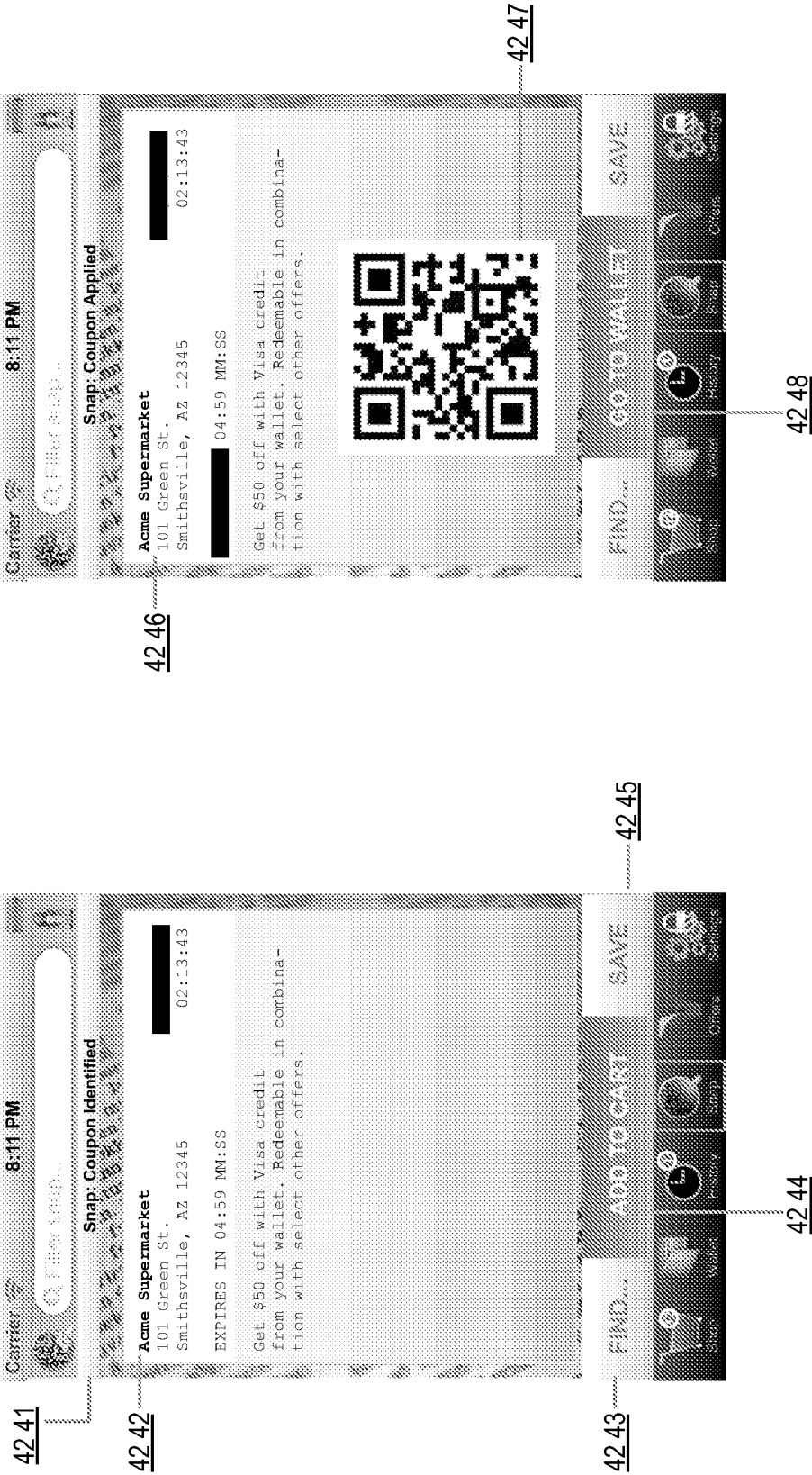


FIGURE 42D

Example: Virtual Wallet Mobile App - Snap Mode

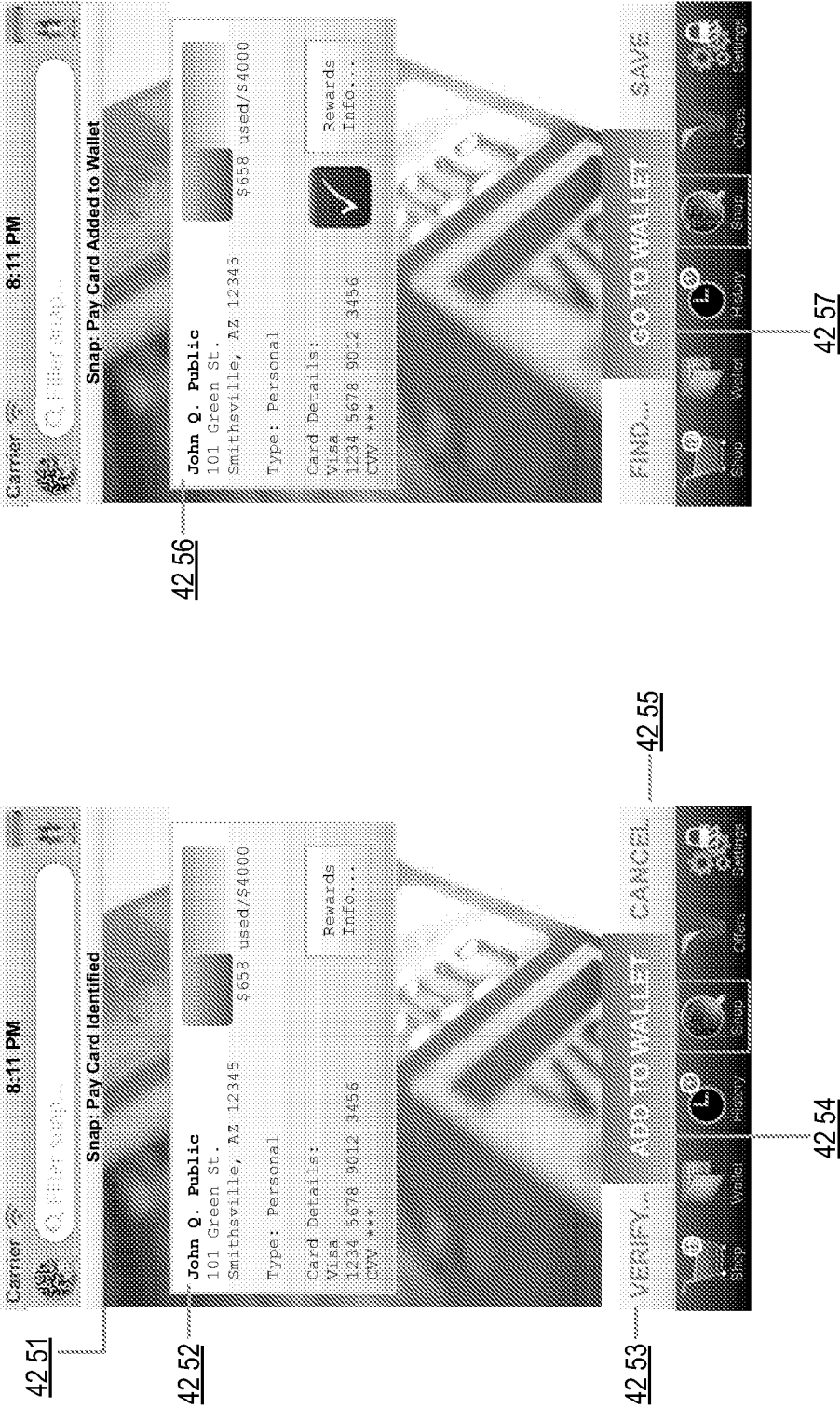


FIGURE 42E

Example: Virtual Wallet Mobile App - Snap Mode

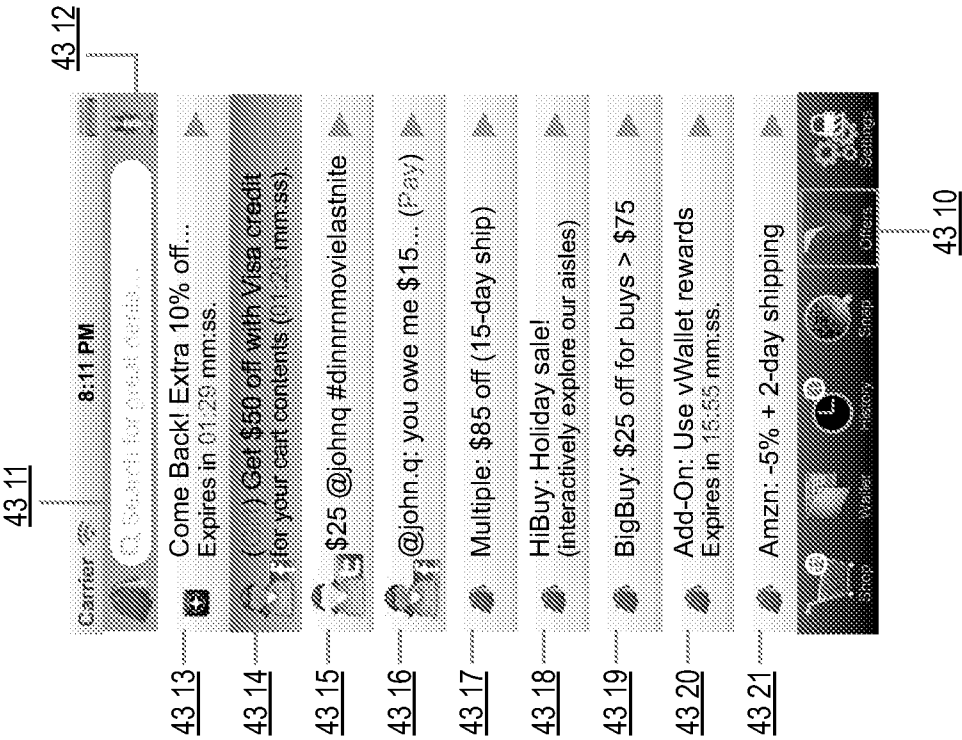


FIGURE 43

Example: Virtual Wallet Mobile App - Offers

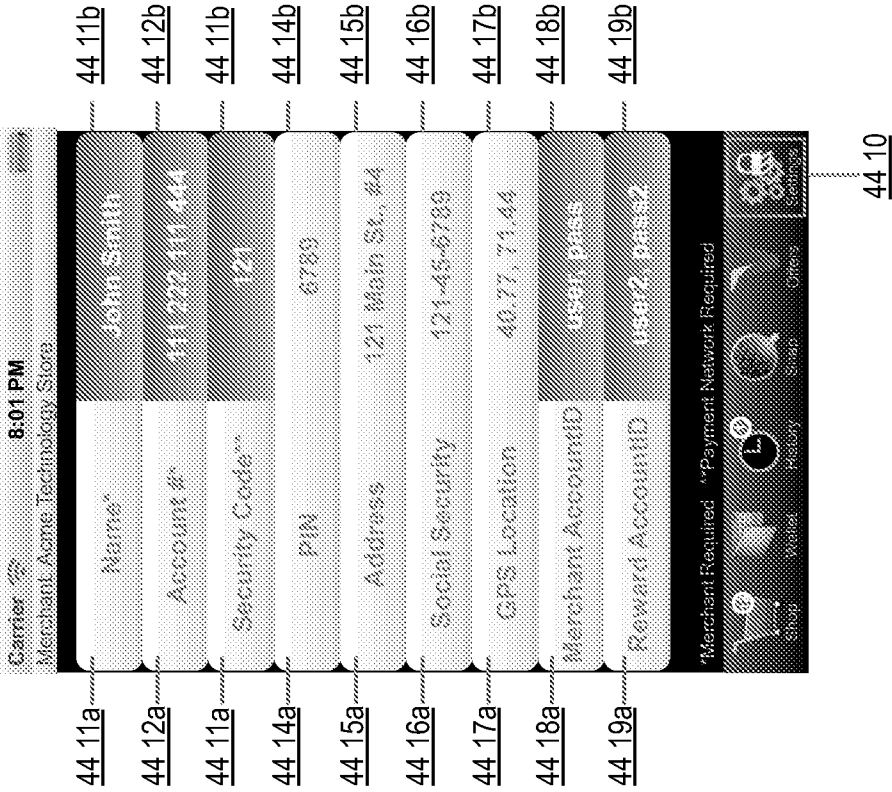


FIGURE 44A

Example: Virtual Wallet Mobile App

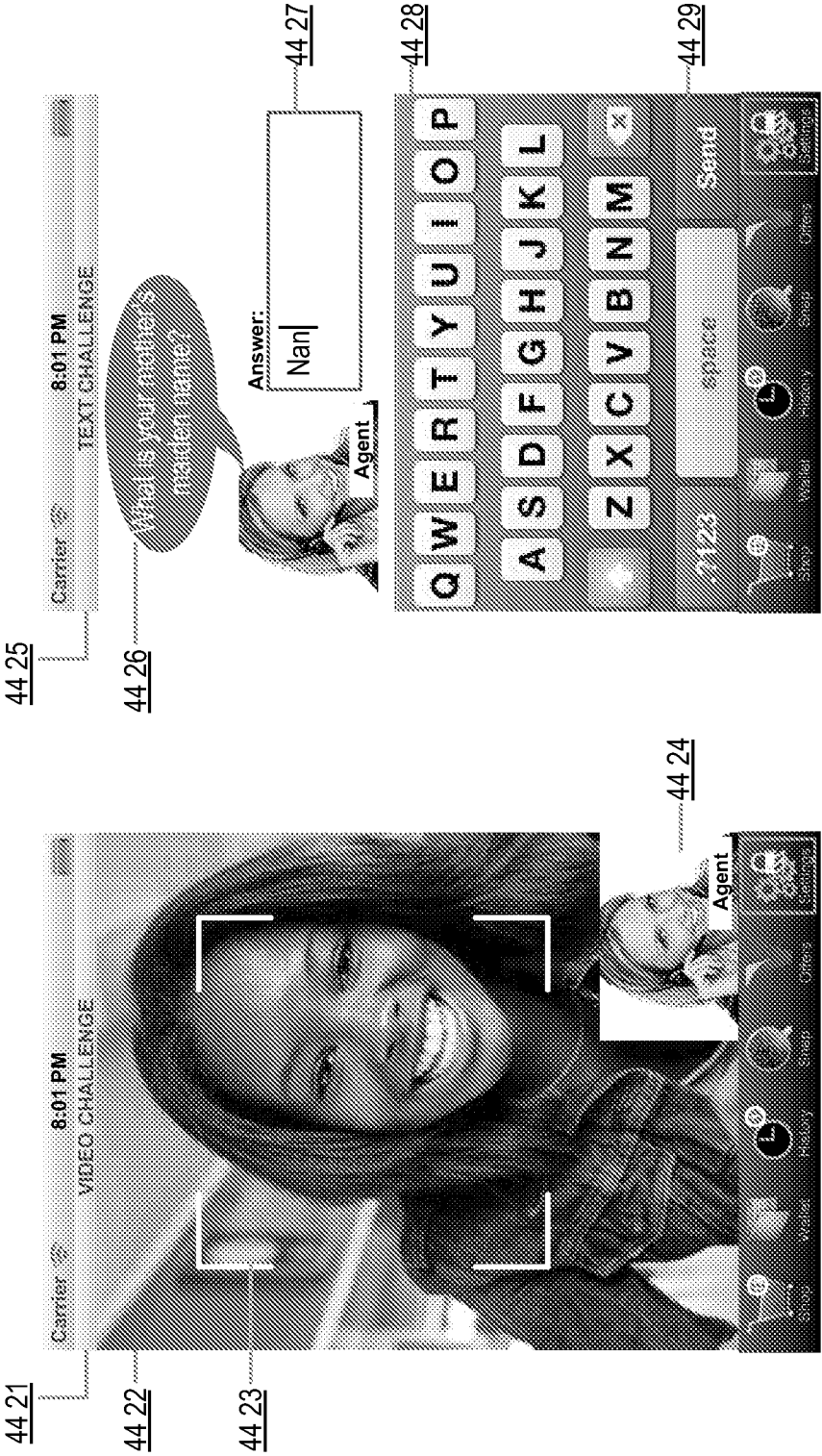
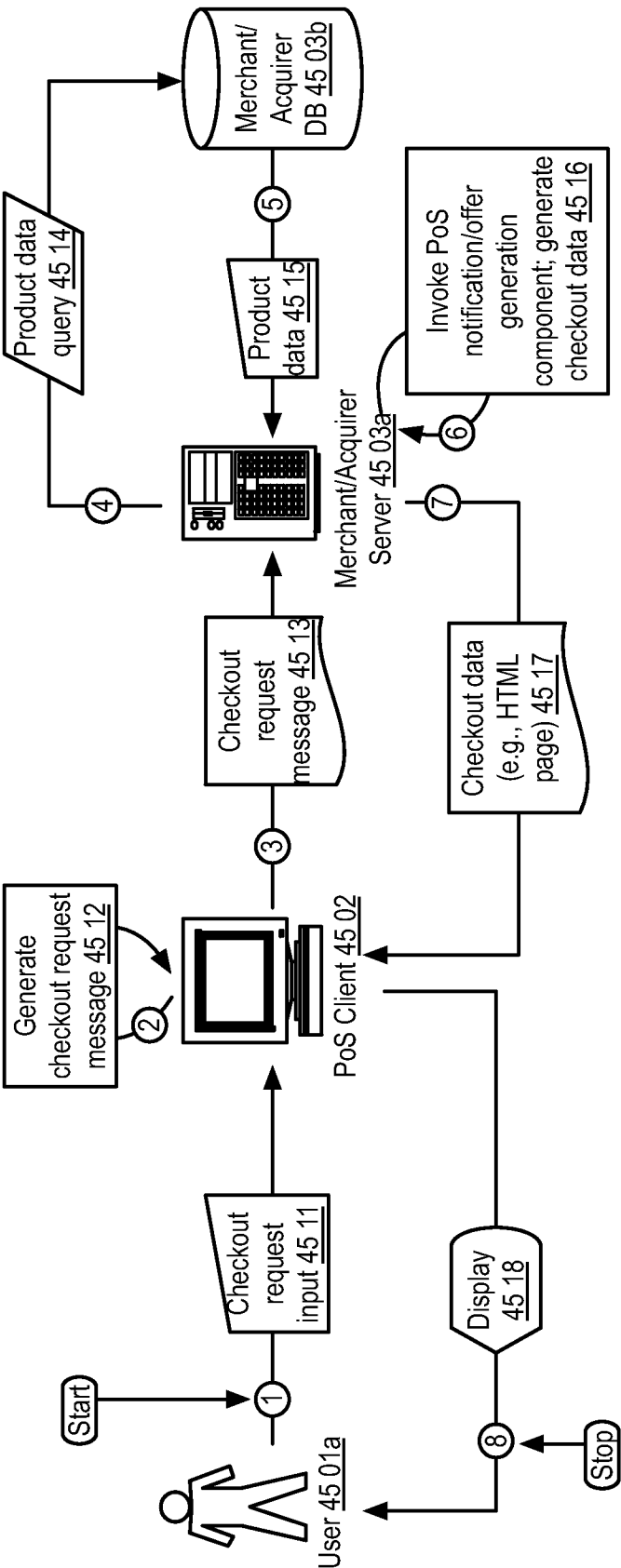


FIGURE 44B

Example: Virtual Wallet Mobile App



Example Data Flow: User Purchase Checkout

FIGURE 45

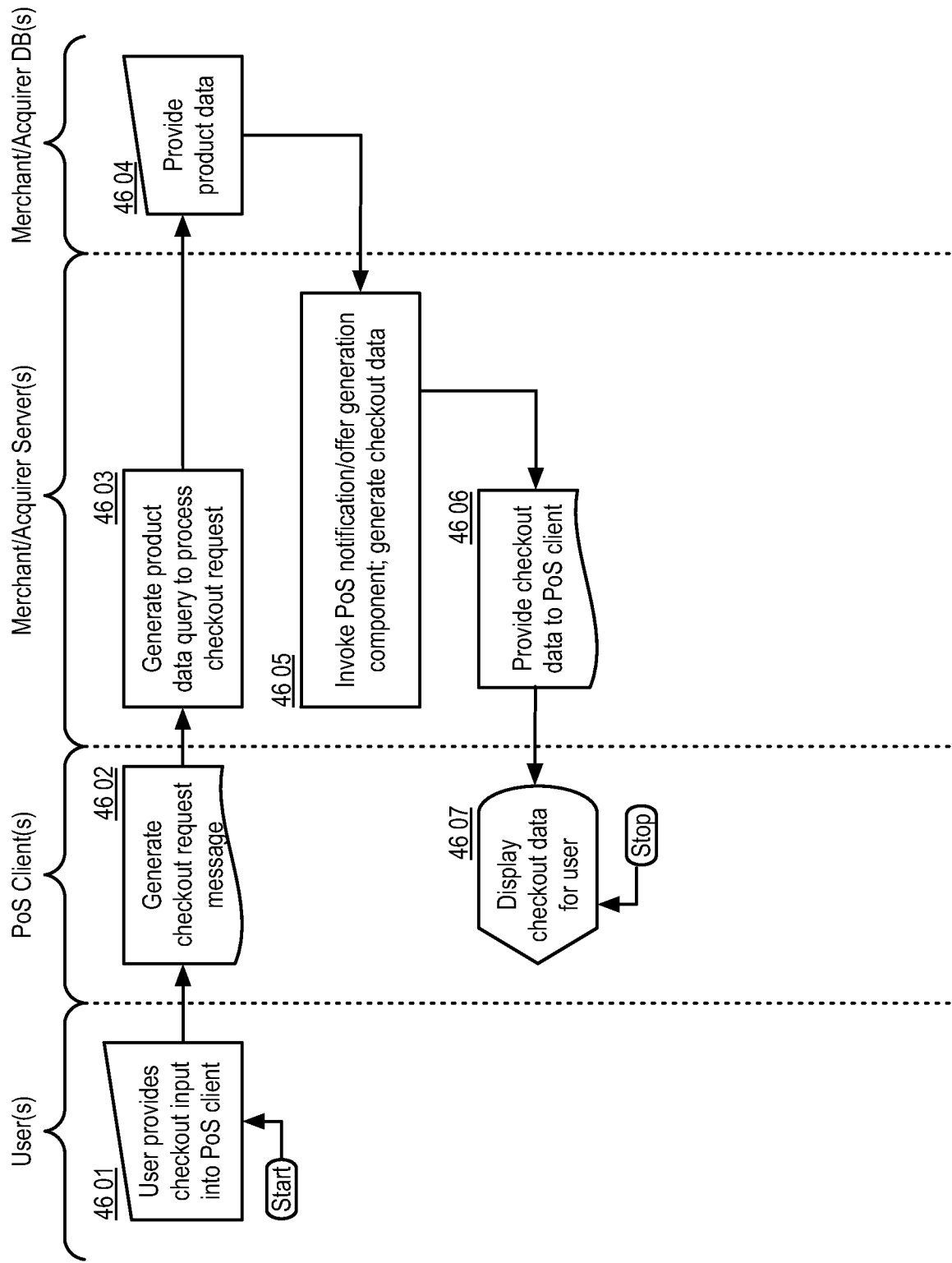


FIGURE 46 Example Logic Flow: User Purchase Checkout ("UPC") component 4600

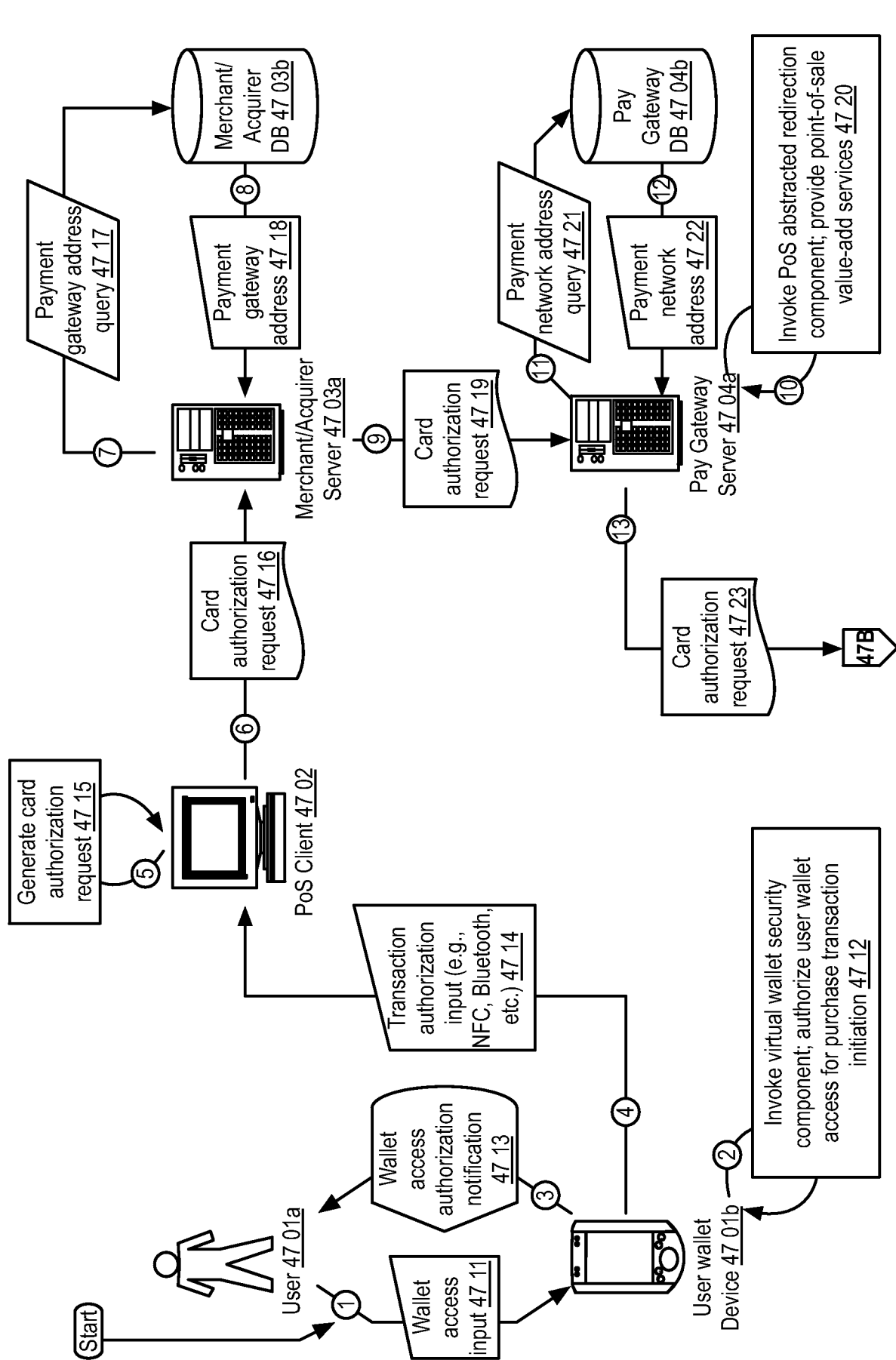
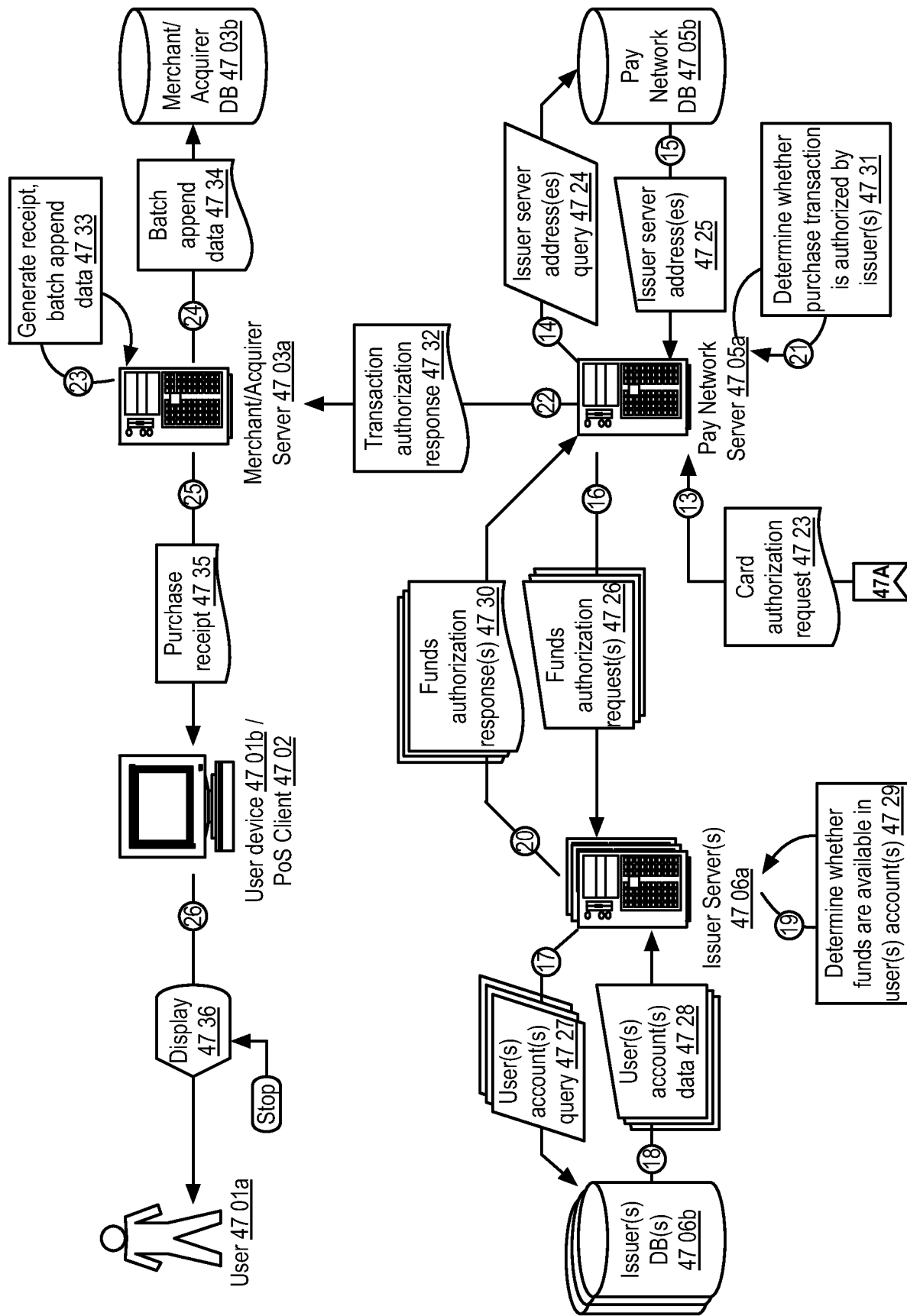


FIGURE 47A

Example Data Flow: Purchase Transaction Authorization



Example Data Flow: Purchase Transaction Authorization

FIGURE 47B

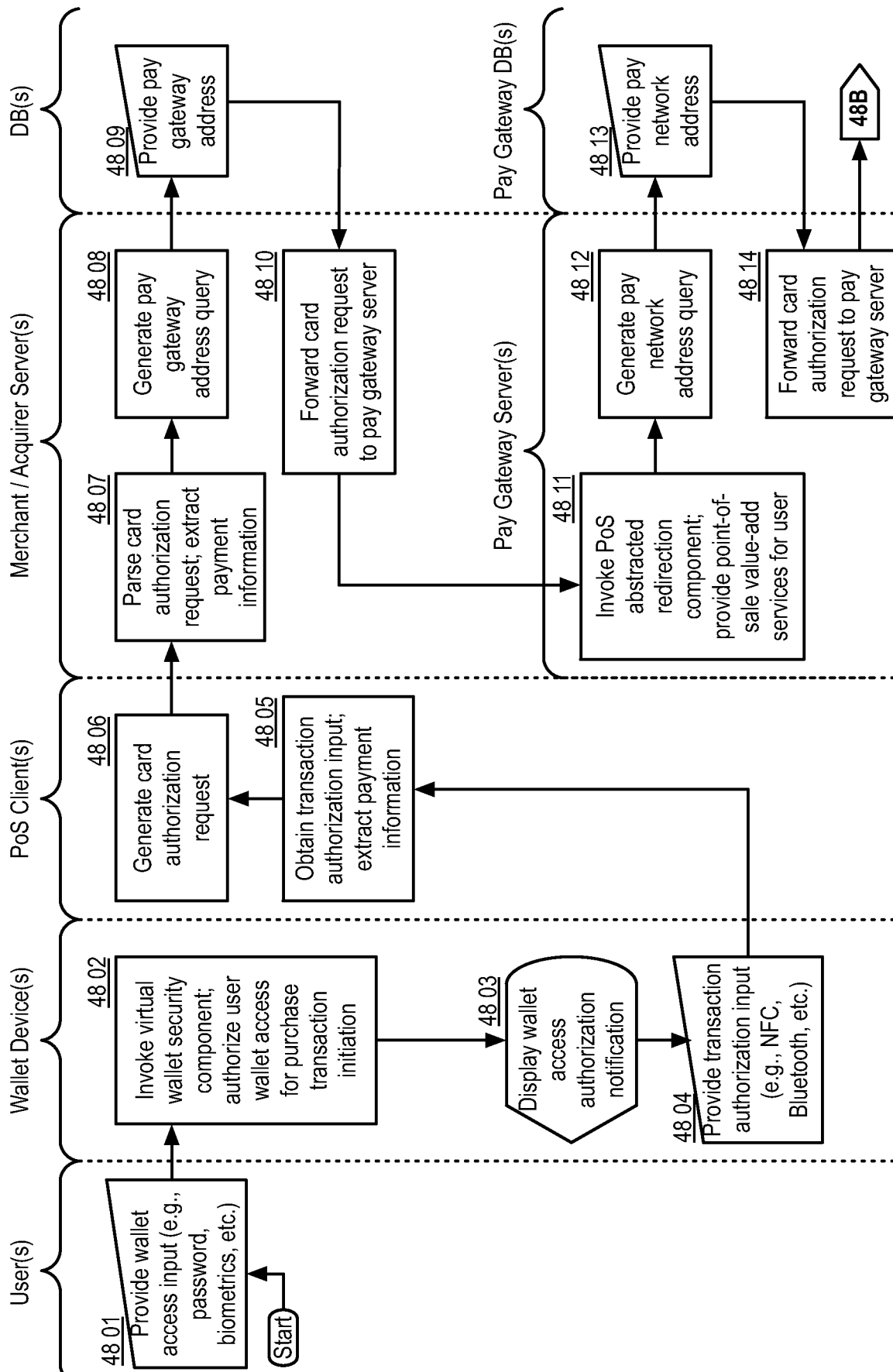
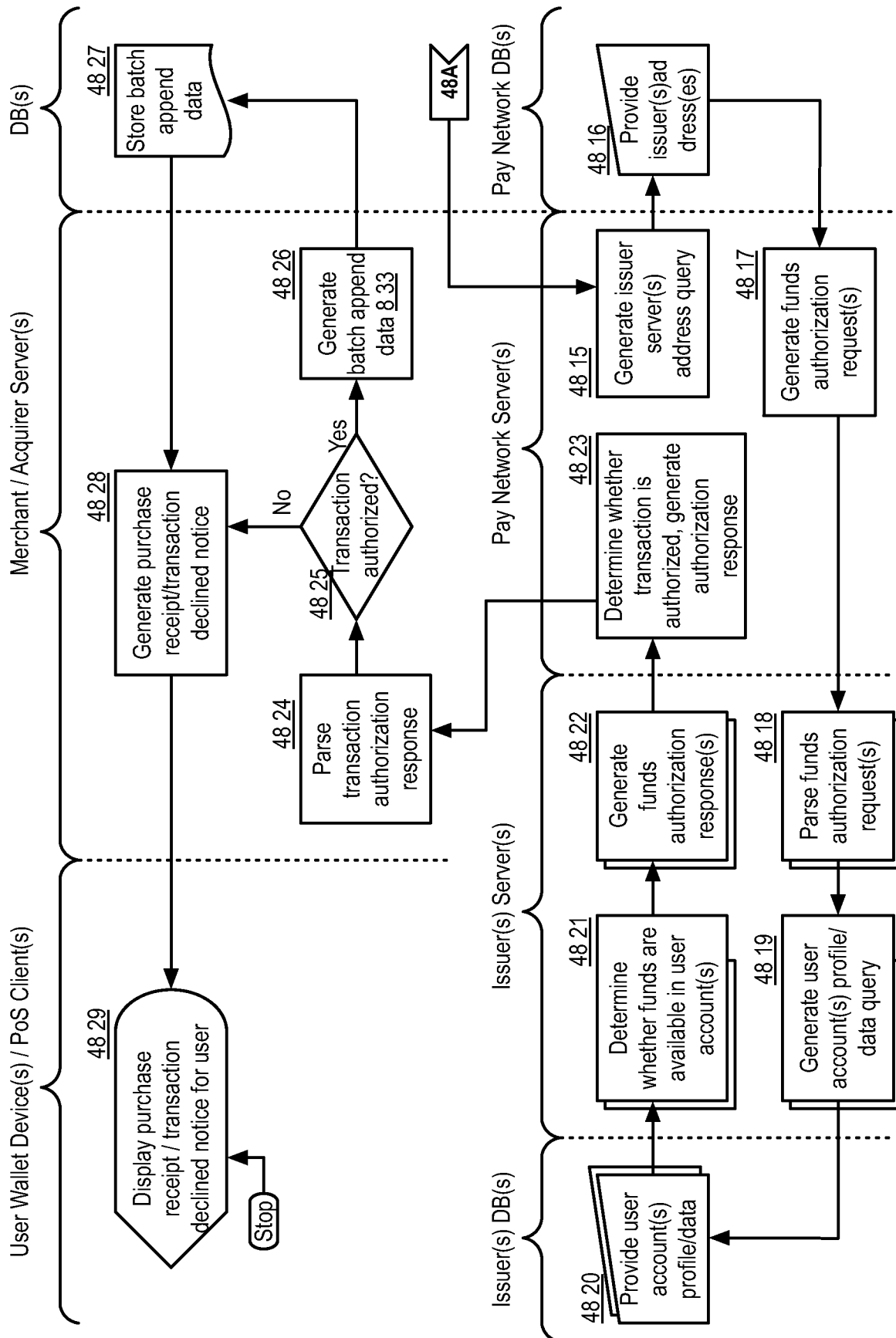


FIGURE 48A

Example: Purchase Transaction Authorization ("PTA") component 4800



Example: Purchase Transaction Authorization ("PTA") component 4800

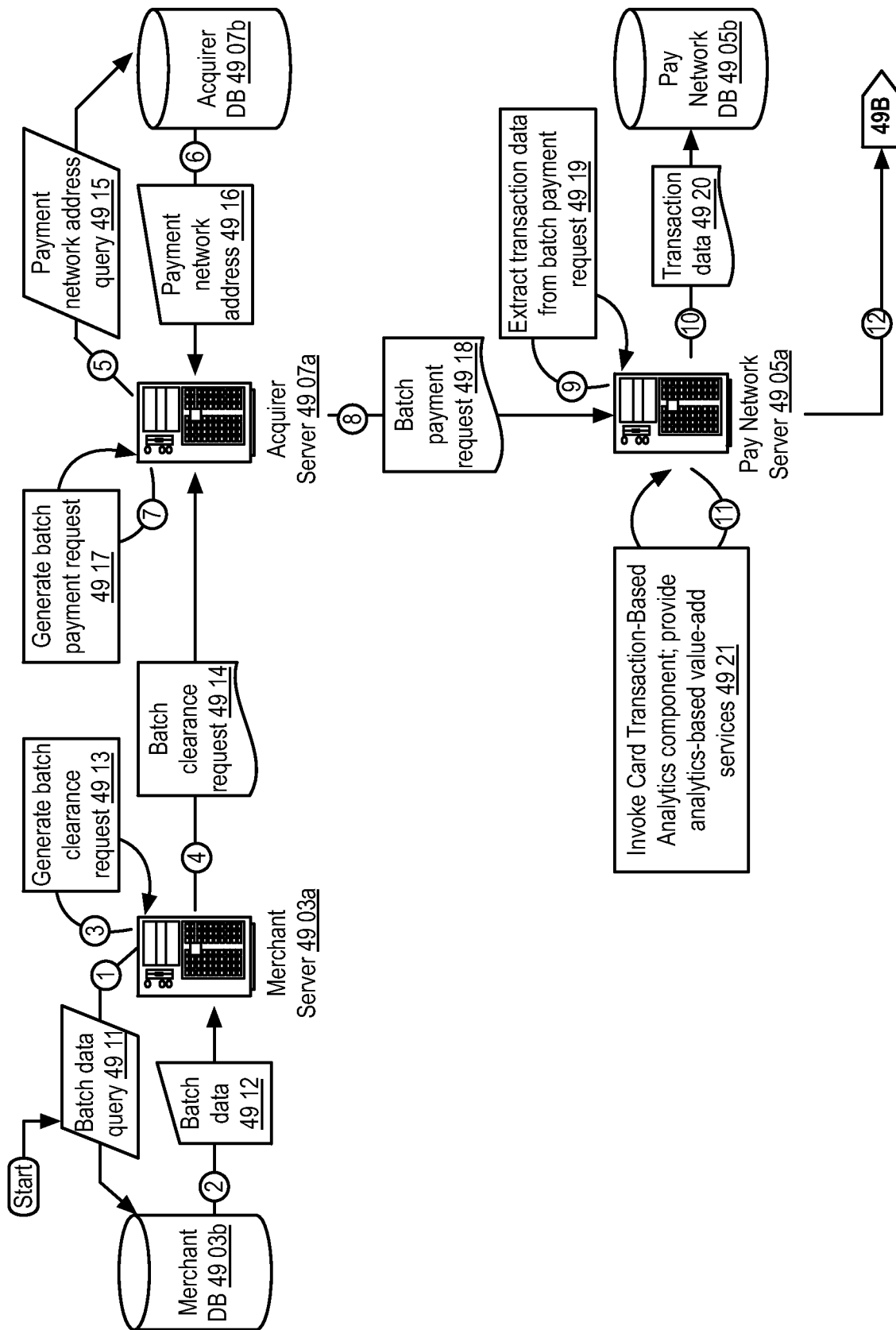
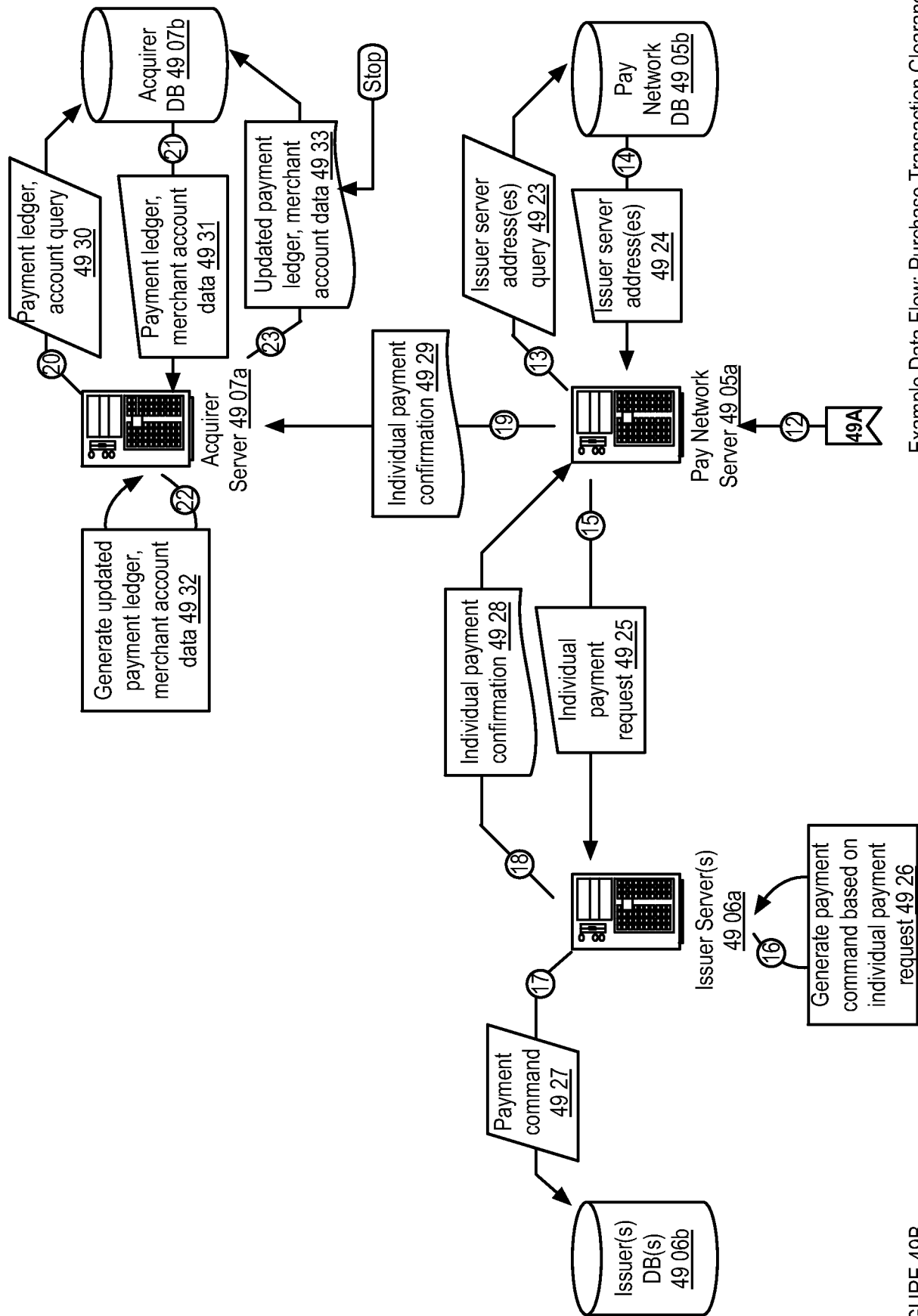


FIGURE 49A

Example Data Flow: Purchase Transaction Clearance



Example Data Flow: Purchase Transaction Clearance

FIGURE 49B

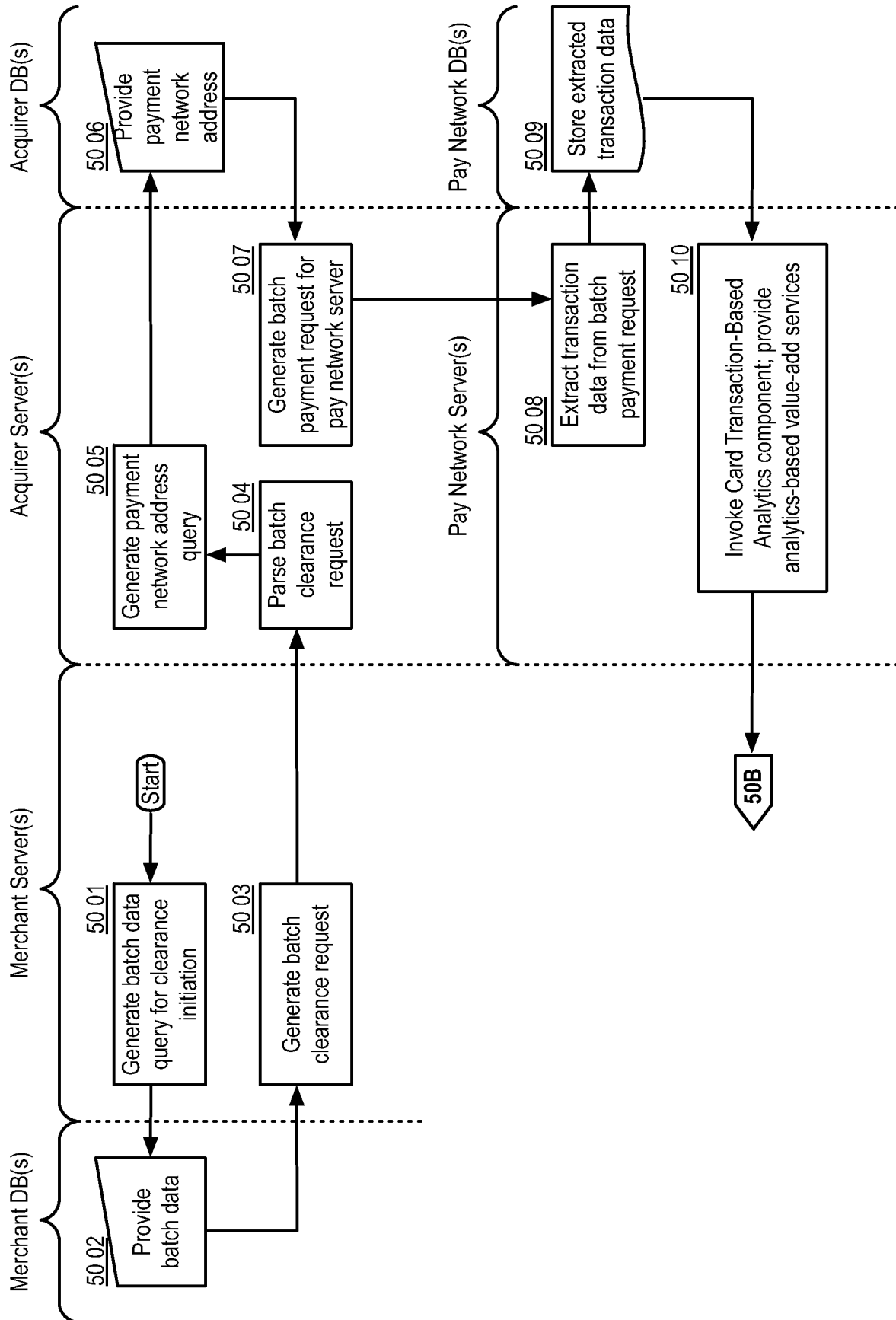


FIGURE 50A Example Logic Flow: Purchase Transaction Clearance ("PTC") component 5000

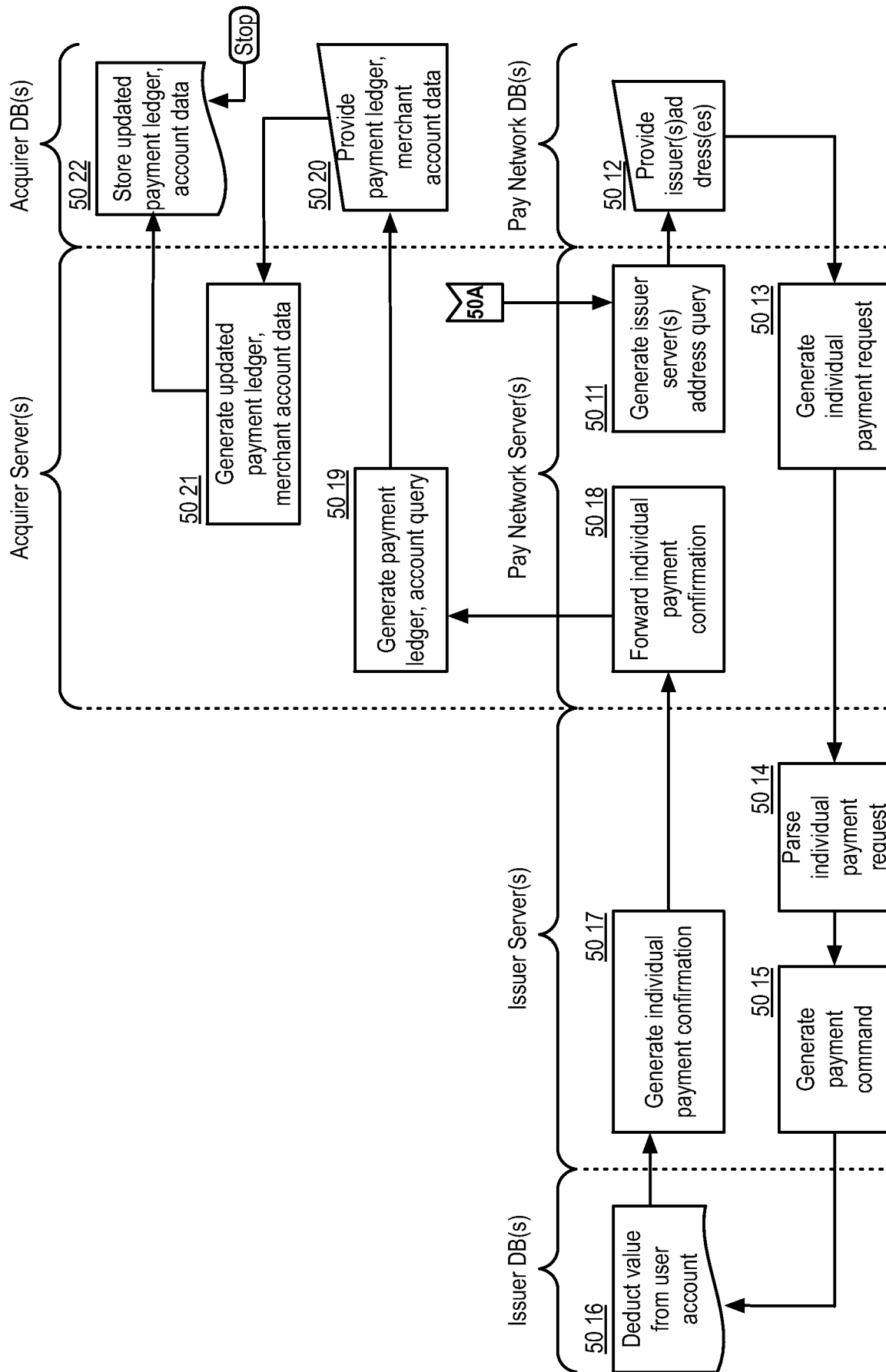


FIGURE 50B

Example Logic Flow: Purchase Transaction Clearance ("PTC") component 5000

