

(57) Abstract: The PREPAID ACCOUNT FUNDS TRANSFER APPARATUSES, METHODS AND SYSTEMS ("PAFT") transform prepaid account invitation requests via PAFT components into scheduled prepaid account transactions. In one embodiment, the PAFT obtains, from a transferor user holding a transferor prepaid account, a prepaid account invitation request specifying a transferee user and a prepaid funds transfer amount. The PAFT generates a customized prepaid account invitation for the transferee user based on the prepaid account invitation request, and provides the customized prepaid account invitation for the transferee user. Upon obtaining an invitation acceptance from the transferee user in response to the provided customized prepaid account invitation, the PAFT creates a transferee prepaid account for the transferee user, and transfers, upon creating the transferee prepaid account, the prepaid funds transfer amount of funds specified in the prepaid account invitation request from the transferor prepaid account to the created transferee prepaid account.

PREPAID ACCOUNT FUNDS TRANSFER APPARATUSES, METHODS AND SYSTEMS

[0001] This patent application disclosure document (hereinafter “description” and/or “descriptions”) describes inventive aspects directed at various novel innovations (hereinafter “innovation,” “innovations,” and/or “innovation(s)”) 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 patent disclosure document 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. 61/304,594 filed February 15, 2010, entitled “PREPAID ACCOUNT TRANSFEROR INVITATION FOR A TRANSFEREE LINKED PREPAID ACCOUNT,” attorney docket no. 20270-086PV. The entire contents of the aforementioned application are herein expressly incorporated by reference.

FIELD

[0003] The present inventions are directed generally to apparatuses, methods, and systems for account-based transactions, and more particularly, to PREPAID ACCOUNT FUNDS TRANSFER APPARATUSES, METHODS AND SYSTEMS (“PAFT”).

BACKGROUND

1

2 **[0004]** Many users prefer to engage in account-based transactions, such as credit
3 or debit transactions, because of the security and convenience they provide. Prepaid
4 cards are particularly attractive because they provide additional advantages such as
5 spending alerts and controls. The range of account-based transactions is currently
6 limited in terms of the types of transactions possible, and the persons with whom such
7 transactions may be conducted.

BRIEF DESCRIPTION OF THE DRAWINGS

8

9 **[0005]** The accompanying appendices and/or drawings illustrate various non-
10 limiting, example, inventive aspects in accordance with the present disclosure:

11 **[0006]** FIGURE 1 shows a block diagram illustrating example aspects of prepaid
12 account funds transfer in some embodiments of the PAFT;

13 **[0007]** FIGURE 2 shows a data flow diagram illustrating an example procedure to
14 generate a customized invitation for a user to open a new prepaid account in some
15 embodiments of the PAFT;

16 **[0008]** FIGURES 3A-B show logic flow diagrams illustrating example aspects of
17 generating a customized invitation for a user to open a new prepaid account in some
18 embodiments of the PAFT, e.g., a Prepaid Account Application Invitation (PAAI)
19 component 300;

1 **[0009]** FIGURES 4A-C show data flow diagrams illustrating an example
2 procedure to register a new prepaid account for a user in some embodiments of the
3 PAFT;

4 **[0010]** FIGURES 5A-D show logic flow diagrams illustrating example aspects of
5 registering a new prepaid account for a user in some embodiments of the PAFT, e.g., a
6 Prepaid Account Registration (PAR) component 500;

7 **[0011]** FIGURES 6A-B show data flow diagrams illustrating an example
8 procedure to transfer funds from a prepaid account of a user to a prepaid account of
9 another user in some embodiments of the PAFT;

10 **[0012]** FIGURE 7 shows a logic flow diagram illustrating example aspects of
11 implementing a transferor user-initiated trigger for prepaid account funds transfer in
12 some embodiments of the PAFT, e.g., a Transferor-Initiated Funds Transfer Trigger (Tr-
13 FTT) component 700;

14 **[0013]** FIGURE 8 shows a logic flow diagram illustrating example aspects of
15 implementing a transferee user-initiated trigger for prepaid account funds transfer in
16 some embodiments of the PAFT, e.g., a Transferee-Initiated Funds Transfer Trigger (Te-
17 FTT) component 800;

18 **[0014]** FIGURE 9 shows a logic flow diagram illustrating example aspects of
19 implementing a server-initiated trigger for prepaid account funds transfer in some
20 embodiments of the PAFT, e.g., a Server-Initiated Funds Transfer Trigger (S-FTT)
21 component 900;

1 **[0015]** FIGURES 10A-C show logic flow diagrams illustrating example aspects of
2 transferring funds from a prepaid account of a user to a prepaid account of another user
3 in some embodiments of the PAFT, e.g., a Prepaid Account Funds Transfer Processing
4 (PA-FTP) component 1000;

5 **[0016]** FIGURE 11 shows a user interface diagram illustrating example aspects of
6 managing prepaid account funds transfers in some embodiments of the PAFT; and

7 **[0017]** FIGURE 12 shows a block diagram illustrating embodiments of a PAFT
8 controller.

9 **[0018]** The leading number of each reference number within the drawings
10 indicates the figure in which that reference number is introduced and/or detailed. As
11 such, a detailed discussion of reference number 101 would be found and/or introduced
12 in Figure 1. Reference number 201 is introduced in Figure 2, etc.

DETAILED DESCRIPTION

PREPAID ACCOUNT FUNDS TRANSFER (PAFT)

[0019] The PREPAID ACCOUNT FUNDS TRANSFER APPARATUSES, METHODS AND SYSTEMS (hereinafter “PAFT”) transform prepaid account invitation requests, via various PAFT components, into scheduled prepaid account transactions.

[0020] FIGURE 1 shows a block diagram illustrating example aspects of prepaid account funds transfer in some embodiments of the PAFT. In some implementations, a prepaid account holder, e.g., 101, may owe funds to users, e.g., 102-104. For example, the prepaid account holder may owe funds for services rendered, e.g., in the example of users 103 and 103. In some scenarios, the user to whom funds are owe may not be able to accept the funds (see, e.g., 111) via account-based transactions such as credit card payment, for example, because they may not have the infrastructure to process such transactions as a merchant (e.g., gardener, repairman). The users may also be unwilling to accept the funds owed in other forms such as a check, because they may not even have a bank account, or because accepting such forms of payment may subject them to higher levels of risk (see, e.g., 112-113) that they find unacceptable, e.g., 116. In some scenarios, such users may be owed funds on, e.g., a regular, period, basis (see e.g., 114). For example, some services may require that performance be executed more than once, perhaps on a regular repeatable schedule, thus requiring payment on a regular repeatable schedule. In another example, a long schedule of payment of funds owed may be pre-determined, exacerbating the problems of transferring funds owed from the prepaid account holder to the user(s). In some scenarios, the prepaid account holder

1 may wish to transfer funds (see, e.g., 117) to a user, e.g., 104, but may wish to exercise
2 spending controls on the user for whom the prepaid account holder wishes to transfer
3 funds owed (see, e.g., 118).

4 **[0021]** In some implementations, the PAFT may allow the prepaid account holder
5 101 to invite the users 102-104 (to whom the prepaid account holder desires to transfer
6 funds owed), to open prepaid accounts via the PAFT. The PAFT may open prepaid
7 accounts for such users, and may provide links between the prepaid account of the
8 prepaid account holder 101 and the user 102-104. In some implementations, the PAFT
9 may facilitate fast, reliable, and secure transfer of funds from the prepaid account of the
10 prepaid account holder 101 to the prepaid accounts of the users 102-104. In some
11 implementations, the PAFT may initiate funds transfer immediately upon opening of
12 the prepaid accounts of the users 102-104. Thus, if the users accept the invitation of the
13 prepaid account holder, the PAFT may, in some implementations, conduct real-time
14 secure transfer of funds from the prepaid account holder to the users. In some
15 implementations, the PAFT may also allow the account holders and/or users to create
16 transfer schedules for future funds transfers. In some implementations, the PAFT may
17 enable the users who received new accounts to similarly invite others who do not have
18 prepaid accounts to establish new prepaid accounts with the PAFT to facilitate further
19 funds transfers. Such prepaid account funds transfers may also allow the transferors of
20 funds owed to exercise spending controls for themselves via their prepaid accounts.

21 **[0022]** FIGURE 2 shows a data flow diagram illustrating an example procedure to
22 generate a customized invitation for a user to open a new prepaid account in some
23 embodiments of the PAFT. In some implementations, a transferor user 201 may wish to

1 invite a transferee user 206 to open a prepaid account for funds transfer. The transferor
2 user may utilize a transferor client 202 communicating with a PAFT server 203
3 (“server”) to generate a prepaid account invitation customized for the transferee user.
4 For example, the transferor user may provide user input, e.g., invitation request input
5 211, into the client device. In various implementations, the user input may include, but
6 not be limited to: keyboard entry, 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 **[0023]** In some implementations, the user may provide invitation request input,
10 e.g., 211, to the transferor client indicating that the user desires to invite the transferee
11 user to apply for a prepaid account. For example, the transferor user may desire for the
12 transferee user to open the prepaid account so that the transferor user can transfer
13 funds to the transferee user’s newly opened prepaid account. As another example, the
14 transferor user may wish to obtain funds from the transferee user on a regular basis
15 (e.g., according to a predetermined schedule), and may wish for the transferee user to
16 utilize a prepaid account to provide the payments for funds transfer for risk
17 minimization and/or security purposes.

18 **[0024]** In response to the transferor providing the invitation request input, the
19 transferor client may generate a prepaid account invitation request, e.g., 212, and
20 provide, e.g., 213, the generated prepaid account invitation request to the server, e.g.,
21 203. For example, a browser application executing on the transferor client may provide,
22 on behalf of the user, a (Secure) Hypertext Transfer Protocol (“HTTP(S)”) POST
23 message including the prepaid account invitation request details for the server in the

1 form of data formatted according to the eXtensible Markup Language (“XML”). Below
 2 is an example HTTP(S) POST message that may be sent by a browser executing on the
 3 transferor client to provide a prepaid account invitation request to the server:

```

4  POST /inviterequests.php HTTP/1.1
5  Host: www.paft.com
6  Content-Type: Application/XML
7  Content-Length: 921
8  <?XML version = "1.0" encoding = "UTF-8"?>
9  <paft_invitation_request>
10     <request_id>FBSWEUHY48</request_id>
11     <timestamp>2011-02-09 09:27:13</timestamp>
12     <transferor_ID>transferor@paft.com</user_ID>
13     <client_IP>192.168.20.78</client_IP>
14     <transferee_details>
15         <transferee_type>individual</transferee_type>
16         <first_name>John</first_name>
17         <last_name>Public</last_name>
18         <contact_type>email</contact_type>
19         <contact_info>john.q.public@gmail.com</contact_info>
20         <alt_contact_type>SMS</alt_contact_type>
21         <alt_contact_info>415-927-4555</alt_contact_info>
22         <client_type>smartphone</client_type>
23     </transferee_details>
24     <num_invite_attempts>3</num_invite_attempts>
25     <account_params>
26         <same_bank>yes</same_bank>
27         <same_branch>no</same_branch>
28         <persistent_link>yes</persistent_link>
29     </account_params>
30     <transfer_params>
31         <on_schedule>no</on_schedule>
32         <one_time>yes</one_time>
33         <amount>$250.00</amount>
34         <instant_transfer>yes</instant_transfer>
35     </transfer_params>
36 </paft_invitation_request>
37

```

38 **[0025]** Upon receiving the prepaid account invitation request from the transferor
 39 client, the server 203 may parse the invitation request to extract the included invitation
 40 request data. Parsing procedures that the server may employ are outlined in the
 41 description below with reference to FIGURE 12. Upon determining the parameters of
 42 the invitation request, the server may generate a unique ID associated with the
 43 invitation to be generated based on the request. In some implementations, the server
 44 may extract, based on parsing the request message, data regarding the messaging mode
 45 (e.g., SMS, email, voicemail, automated/manual phone call, postal mail, etc.) by which

1 the transferor user desires the PAFT to provide a prepaid account invitation for the
 2 transferee user. Based on the user messaging mode preference(s), the server may
 3 generate a query, e.g., 214, for a template from a database which the server can utilize to
 4 generate a customized prepaid account invitation message for the transferee user in
 5 accordance with the messaging mode preferences of the transferor user. The server may
 6 issue the query, e.g., 215, to a database storing invitation templates, e.g., forms database
 7 204a. For example, the server may execute a PHP script including Structured Query
 8 Language ("SQL") commands to query a relational database storing invitation
 9 templates. An example listing, written substantially in the form of PHP/SQL
 10 commands, illustrating substantive aspects of querying the database for an invitation
 11 template is provided below:

```

12  <?PHP
13  header('Content-Type: text/plain');
14  function invitation_template_query($template_type, $DBserver, $password) {
15  mysql_connect("254.92.185.102", $DBserver, $password); // access database server
16  mysql_select_db("TEMPLATES.SQL"); // select database table to search
17  //create query for invitation template
18  $query = "SELECT invitation_template FROM TemplateStore WHERE type LIKE '%"
19  $template_type";
20  $result = mysql_query($query); // perform the search query
21  mysql_close("TEMPLATES.SQL"); // close database access
22  return $result; // return search results
23  ?>
24

```

25 **[0026]** In response to the invitation template query, the forms database may
 26 provide, e.g., 216, the requested invitation template to the server. In some
 27 implementations, the server may parse, e.g., 217, the invitation request to determine
 28 information of the transferee user who will be the recipient of the prepaid account
 29 invitation. Based on the recipient information of the transferee user, the server may
 30 generate a customized prepaid account invitation. For example, the customized prepaid
 31 account invitation may take the form of a text message, SMS, email, electronic
 32 communication, fax, voicemail, automated phone call, script print-out for a customer

1 service representative to make a manual phone call, automated/manual online chat
 2 script, webpage, and/or the like. In some implementations, the server may store, e.g.,
 3 218, the generated customized invitation, unique ID, and/or other invitation data to a
 4 database, eg., invitations database 204b. An example listing, written substantially in the
 5 form of PHP/SQL commands, to store the invitation data to a database is provided
 6 below:

```

7  <?PHP
8  header('Content-Type: text/plain');
9  mysql_connect("254.92.185.103",$DBserver,$password); // access database server
10 mysql_select("INVITATIONS.SQL"); // select database to append
11 mysql_query("INSERT INTO UserInvitationsTable (requestID, transferorID,
12     transferorIP, time, uniqueID, transferee_type, transfereename,
13     transfereecontacttype, transfereecontactinfo, altcontactinfo,
14     transfereeclienttype, numinvites, samebank, samebranch, persistentlink,
15     onschedule, onetime, amount, instanttransfer, custominvite)
16     VALUES ($request_ID, $transferor_ID, $transferor_IP, $timestamp, $unique_ID,
17     $transferee_type, $transferee_name, $transferee_contact_type,
18     $transferee_contact_info, $alt_contact_info, $transferee_client_type,
19     $num_invite_attempts, $same_bank, $same_branch, $persistent_link,
20     $on_schedule, $one_time, $amount, $instant_transfer, $custom_invitation)");
21     // add data to table in database
22 mysql_close("INVITATIONS.SQL"); // close connection to database
23 ?>
24

```

25 **[0027]** In some implementations, the server may provide the generated
 26 customized prepaid account invitation, e.g., 219, for a transferee client, e.g., 205, of a
 27 transferee user, e.g., 206, specified by the invitation request of the transferor user. For
 28 example, the server may send an electronic mail message to an electronic mail
 29 messaging account of the transferee user. An example listing, written substantially in
 30 the form of PHP/SQL commands, for the server to send an electronic mail message to
 31 an electronic mail messaging account of the transferee user is provided below:

```

32  <?PHP
33  header('Content-Type: text/plain');
34  // customized invitation is stored in data variable $message, and transferor
35  user name is stored in $transferor
36  $to = 'john.q.public@gmail.com';
37  $subject = 'Prepaid Account Invitation from ' . $transferor;
38  $headers = 'From: invitations@paft.com' . "\r\n" .
39  'Reply-To: invitations@paft.com' . "\r\n" .
40  'X-Mailer: PHP/' . phpversion();

```

```
1 mail($to, $subject, $message, $headers);  
2 ?>  
3
```

4 **[0028]** In some implementations, the transferee client may render, e.g., 220, the
5 customized prepaid account invitation for presentation to the transferee user. For
6 example, the transferee client may render a webpage, electronic message, text / SMS
7 message, buffer a voicemail, emit a ring tone for the transferee user to answer a phone
8 and/or play an audio message, etc. The transferee client may then present the
9 customized prepaid account invitation for the transferee user, e.g., 221. For example,
10 the transferee client may provide output including, but not limited to: sounds, music,
11 audio, video, images, tactile feedback, vibration alerts (e.g., on vibration-capable client
12 devices such as a smartphone etc.), and/or the like. In some implementations, the
13 server may also provide an invitation confirmation message, e.g., 222, for the transferor
14 client of the transferor user. For example, the server may send an electronic mail
15 message to an electronic mail messaging account of the transferor user. The transferor
16 client may render, e.g., 223, the customized prepaid account invitation for presentation
17 to the transferor user. For example, the transferee client may render a webpage,
18 electronic message, text / SMS message, buffer a voicemail, emit a ring tone for the
19 transferee user to answer a phone and/or play an audio message, etc. The transferor
20 client may then present the customized prepaid account invitation for the transferor
21 user, e.g., 224. For example, the transferor client may provide output including, but not
22 limited to: sounds, music, audio, video, images, tactile feedback, vibration alerts (e.g.,
23 on vibration-capable client devices such as a smartphone etc.), and/or the like.

24 **[0029]** FIGURES 3A-B show logic flow diagrams illustrating example aspects of
25 generating a customized invitation for a user to open a new prepaid account in some

1 embodiments of the PAFT, e.g., a Prepaid Account Application Invitation (PAAI)
2 component 300. In some implementations, a transferor user may invite a transferee
3 user to open a prepaid account for funds transfer. For example, the transferor user may
4 provide invitation request input, e.g. 301, into the transferor client. In some
5 implementations, the user provided invitation request input may indicate to the
6 transferor client that the user desires to invite the transferee user to apply for a prepaid
7 account. For example, the transferor user may desire for the transferee user to open the
8 prepaid account so that the transferor user can transfer funds to the transferee user's
9 newly opened prepaid account. As another example, the transferor user may wish to
10 obtain funds from the transferee user on a regular basis (e.g., according to a
11 predetermined schedule), and may wish for the transferee user to utilize a prepaid
12 account to provide the payments for funds transfer for risk minimization and/or
13 security purposes. In response to the transferor providing the invitation request input,
14 the transferor client may generate a prepaid account invitation request, e.g., 302, and
15 provide the generated prepaid account invitation request to the PAFT server.

16 **[0030]** Upon obtaining, e.g., 303, the prepaid account invitation request from the
17 transferor client, the server may parse the invitation request to extract the included
18 invitation request data, e.g., 304. Parsing procedures that the server may employ are
19 outlined in the description below with reference to FIGURE 12. Based on the parsing,
20 the server may extract data fields including, but not limited to: request_id, timestamp,
21 transferor_id, client_IP, transferee_details_list, transferee_type, first_name,
22 last_name, contact_type, contact_info, alt_contact_info, alt_contact_type,
23 alt_contact_info, client_type, num_invite_attempts, account_params_list,
24 same_bank_flag, same_branch_flag, persistent_link_flag, transfer_param_list,

1 on_schedule_flag, one_time_flag, amount, instant_transfer_flag, custom_invitation,
2 transferee_details_list, transferee_client_IP, and/or the like. Upon determining the
3 parameters of the invitation request, the server may generate a unique ID associated
4 with the invitation to be generated based on the request, e.g., 306. For example, the
5 server may execute a hypertext preprocess ("PHP") script which invokes the md5()
6 command to generate a hash of the invitation request message, and utilize the generated
7 message hash as the unique invitation ID. As another example, the server may utilize
8 the rand() command to generate a random number to be utilized as the unique
9 invitation ID. In some implementations, the server may extract, based on parsing the
10 request message, data regarding the messaging mode (e.g., SMS, email, voicemail,
11 automated/manual phone call, postal mail, etc.) by which the transferor user desires the
12 PAFT to provide a prepaid account invitation for the transferee user. Based on the user
13 messaging mode preference(s), the server may generate a query, e.g., 307, for a template
14 from a database which the server can utilize to generate a customized prepaid account
15 invitation message for the transferee user in accordance with the messaging mode
16 preferences of the transferor user. The server may issue the query to a database storing
17 invitation templates, e.g., a forms database. In response to the invitation template
18 query, the forms database may provide, e.g., 308, the invitation template to the server.

19 **[0031]** In some implementations, the server may parse, e.g., 309, the invitation
20 request to determine information of the transferee user who will be the recipient of the
21 prepaid account invitation. For example, the server may extract data field such as, but
22 not limited to: transferee_ID, transferee_type, first_name, last_name, contact_type,
23 contact_info, alt_contact_info, alt_contact_type, alt_contact_info, client_type, and/or
24 the like. Based on the recipient information of the transferee user, the server may

1 generate, e.g., 310, a customized prepaid account invitation. For example, the
2 customized prepaid account invitation may take the form of a text message, SMS, email,
3 electronic communication, fax, voicemail, automated phone call, script print-out for a
4 customer service representative to make a manual phone call, automated/manual online
5 chat script, webpage, and/or the like. In some implementations, the server may
6 generate, e.g., 311, a customized invitation data record including the data fields, unique
7 ID, customized invitation, and/or other invitation data, and provide the invitation data
8 record to a database, e.g., invitations database for storing the invitation data, e.g., 312.
9 In some implementations, the server may provide the generated customized prepaid
10 account invitation, e.g., 313, for a transferee client of a transferee user specified by the
11 invitation request of the transferor user. In some implementations, the transferee client
12 may render, e.g., 314, the customized prepaid account invitation for presentation to the
13 transferee user. The transferee client may then present the customized prepaid account
14 invitation for the transferee user, e.g., 315. In some implementations, the server may
15 also provide an invitation confirmation message, e.g., 316, for the transferor client of the
16 transferor user. The transferor client may render, e.g., 317, the customized prepaid
17 account invitation for presentation to the transferor user. The transferor client may
18 then present the customized prepaid account invitation for the transferor user, e.g., 318.

19 **[0032]** FIGURES 4A-C show data flow diagrams illustrating an example
20 procedure to register a new prepaid account for a user in some embodiments of the
21 PAFT. In some implementations, the transferee user may desire to accept the invitation
22 offer by the server to open a prepaid account with the PAFT. For example, the
23 transferee user 406 may provide invitation acceptance input, e.g., 411, to the transferee
24 client to convey acceptance of the invitation. In various implementations, the user input

1 may include, but not be limited to: keyboard entry, mouse clicks, depressing buttons on
2 a joystick/game console, voice commands, single/multi-touch gestures on a touch-
3 sensitive interface, touching user interface elements on a touch-sensitive display, and/or
4 the like. For example, the transferee user may activate a hyperlink in an email message
5 sent by the server to the transferee user including a customized prepaid account
6 invitation. Activating the hyperlink may serve as the signal for the transferee client that
7 the transferee user wishes to accept the customized prepaid account invitation. In
8 response to the transferee user input, the transferee client may generate a prepaid
9 account invitation acceptance message, e.g., 412 for the server. For example, a browser
10 executing on the transferee client may generate a HTTP(S) GET message including the
11 prepaid account invitation acceptance message. An example HTTP(S) GET message
12 that may be sent by a browser executing on the transferee client to provide a prepaid
13 account invitation acceptance message to the server is provided below:

```
14 GET /inviteaccept.php HTTP/1.1
15 Host: www.paft.com
16 Content-Type: Application/XML
17 Content-Length: 894
18 <?XML version = "1.0" encoding = "UTF-8"?>
19 <paft_invitation_accept>
20   <invitation_id>VBUQQW9M,</invitation_id>
21   <accept_timestamp>2011-02-10 08:27:13</accept_timestamp>
22   <transferor_ID>transferor@paft.com</user_ID>
23   <transferee_details>
24     <transferee_type>individual</transferee_type>
25     <first_name>John</first_name>
26     <last_name>Public</last_name>
27     <contact_type>email</contact_type>
28     <contact_info>john.q.public@gmail.com</contact_info>
29     <alt_contact_type>SMS</alt_contact_type>
30     <alt_contact_info>415-927-4555</alt_contact_info>
31     <client_type>smartphone</client_type>
32   </transferee_details>
33   <client_IP>192.168.20.67</client_IP>
34   <account_params>
35     <same_bank>yes</same_bank>
36     <same_branch>no</same_branch>
37     <persistent_link>yes</persistent_link>
38   </account_params>
39   <transfer_params>
40     <on_schedule>no</on_schedule>
41     <one_time>yes</one_time>
```

```
1         <amount>$250.00</amount>
2         <instant_transfer>yes</instant_transfer>
3     </transfer_params>
4 </paft_invitation_accept>
5
```

6 **[0033]** Upon receiving the prepaid account invitation acceptance message from
7 the transferee client, the server 403 may parse the invitation acceptance to extract the
8 included invitation acceptance data, e.g., 414. For example, the server may extract the
9 unique invitation ID of the invitation which the transferee user has accepted. Based on
10 the extracted invitation data, the server may, in some implementations, determine the
11 type of invitation to which the transferee user has response with acceptance and a client
12 type of the transferee user. In alternate implementations, the server may use the
13 extracted invitation data, e.g., the unique invitation ID, to query, e.g., 415, a database,
14 e.g., invitations database 404b, for the type of the invitation. For example, the server
15 may obtain the invitation type from the database by issuing PHP/SQL commands
16 similar to those described above with reference to FIGURE 2. In response, the database
17 may provide the invitation type, e.g., 416, for the server.

18 **[0034]** In some implementations, the server may generate a query for a prepaid
19 account application form using the invitation type and client type of the transferee user.
20 For example, various transferee users and/or client may require and/or prefer different
21 types of prepaid account application forms. For example, a prepaid account application
22 form may be provided as a (Dynamic) HyperText Markup Language (“HTML”) page,
23 interactive Adobe® Flash object, iPhone® app, Android™ app, interactive voice
24 application, etc. Also, the size, resolution, fidelity and/or like attributes of the prepaid
25 account application form may be customized depending on the screen size, pixel count,
26 audio/video processing capabilities and/or like attributes and/or preferences of the
27 transferee client and/or transferee user. Thus, the server may generate a query for a

1 prepaid application form that is customized to the transferee client and/or transferee
2 user, and issue the query, e.g., 418, to a database, e.g., forms database 404a. In
3 response to the prepaid account application form query, the database may provide a
4 prepaid account application form, e.g., 419, that is customized to the attributes and/or
5 preferences of the transferee client and/or transferee user. The server may provide the
6 obtained prepaid account application form, e.g., 420, to the transferee client. For
7 example, the server may provide the transferee client with a HTML page including a
8 reference to an Adobe® Flash object stored on the server including the prepaid account
9 application. An example HTML code listing including JavaScript™ commands
10 referencing an Adobe® Flash object within the HTML page is provided below:

```
11 <html>
12 <div id="PAFT">
13   If you're seeing this, you don't have Flash Player installed.
14 </div>
15
16 <script type="text/javascript">
17   var app = new SWFObject("www.paft.com/prepaid_app.swf", "Media", "640",
18     "480", "8", "#000000");
19   app.addParam("quality", "high");
20   app.write("PAFT");
21 </script>
22 </html>
23
```

24 **[0035]** Upon obtaining the prepaid account application, the transferee client may
25 execute the prepaid account application for presentation to the user, e.g., 422. For
26 example, with reference to the examples above, a web browser executing on the
27 transferee client device may render, e.g., 421, the HTML web page and may
28 communicate with the server to download the Adobe® Flash object. An Adobe® Flash
29 browser plug-in installed on the transferee client and operating in conjunction with the
30 browser may play/execute the downloaded Flash object for presentation, e.g., 422, to
31 the transferee user.

1 **[0036]** In some implementations, the transferee user may provide application
2 form input, e.g., 423, to the transferee client. For example, with reference to the
3 example above, the Adobe® Flash object including the prepaid account application may
4 provide interactive features, and may allow the user to enter user input/feedback, e.g.,
5 423, via a variety of mechanisms (e.g., keyboard entry into a command-line interface,
6 mouse input in a graphical user interface, gestures on a touch-sensitive interface, voice
7 commands, etc.). Using the user application form input, the client may generate, e.g.,
8 424, a completed prepaid account application. For example, the transferee client
9 executing the Adobe® Flash object may generate, maintain, update and/or store data
10 pertaining to the user's interaction with the Adobe® Flash object (e.g., an application
11 state, an application data structure, a block of memory with data variables, etc.). For
12 example, the Adobe® Flash object may store a prepaid account application data
13 structure encoded according to the JavaScript Object Notation ("JSON") format. An
14 example JSON-encoded prepaid account application data structure is provided below:

```
15  "prepaid_account_application_data"
16  {
17    "timestamp": "2010-06-30 09:23:47",
18    "application_id": "A236269",
19    "md5_auth": "f585e3efede0c3b400b25908f8fa3f6d",
20    "applicant_details": {
21      "applicant_id": "jqpublic",
22      "applicant_name": "John Q. Public",
23      "dob": "1976-02-15",
24      "ssn": "123-45-6789",
25      "credit_check_ok": "yes",
26      "address_firstline": "123 Green Blvd.",
27      "address_secondline": "Apt 58",
28      "zipcode": "98765",
29      "City": "Terrace Heights",
30      "State": "CA"
31    },
32    "account_params": {
33      "account_mode": "prepaid",
34      "account_type": "debit",
35      "account_expiry": "none",
36      "bank_name": "Chasey Bank",
37      "bank_branch": "auto-assign"
38    }
39  }
```

1
2 **[0037]** In some implementations, the server may generate a secure
3 communications session with the transferee client in order to facilitate communication
4 during the prepaid account application procedure between the transferee client and
5 server. As an example, the prepaid account application may provide data stored on the
6 transferee client for the server as prepaid account application data, e.g., completed
7 prepaid account application 425. For example, an Adobe® Flash object running on the
8 transferee client may include ActionScript™ 3.0 commands to create a Secure Sockets
9 Layer (“SSL”) connection with a server, generate a message including a JSON-encoded
10 data structure such as illustrated in the example above, and send the message via the
11 secure SSL connection to the server. Example commands, written substantially in the
12 form of ActionScript™ 3.0, to create a secure SSL connection to a server, load prepaid
13 account application data from a locally stored JSON-encoded data file, and send a
14 message including the JSON-encoded data via the SSL connection to the server, are
15 provided below:

```
16 // import required packages
17 import flash.events.*;
18 import flash.net.socket;
19 import flash.net.URLLoader;
20 import flash.net.URLRequest;
21 import com.adobe.serialization.json.*;
22
23 // obtain server socket policy file, create socket connection to server port
24 system.security.loadPolicyFile("xmlsocket://www.aliserver.com:208");
25 msg = new socket();
26 msg.connect("https://www.paft.com", 255);
27
28 // load data as text string from .json file
29 var loader:URLLoader = new URLLoader();
30 var request:URLRequest = new URLRequest();
31 request.URL = "application_data.json";
32 loader.dataformat = "text"
33 loader.load(request)
34
35 // transmit data to server via secure SSL connection, then close socket
36 msg.writeMultiByte(loader.data, "UTF-8");
37 msg.close();
38
```

1 **[0038]** In some implementations, the server may be executing a PHP script
2 implementing a SSL socket server which listens to incoming communications on a
3 server port to which the transferee client may send data, e.g., prepaid account
4 application 425, encoded according to the JSON format. Upon identifying an incoming
5 communication, the PHP script may read the incoming data from the transferee client
6 into a memory variable which can then be operated upon by the transferee client. An
7 example listing, written substantially in the form of PHP/SQL commands, to accept
8 JSON-encoded prepaid account application data from a transferee client via a SSL
9 connection is provided below:

```
10 <?PHP
11 header('Content-Type: text/plain');
12 // set ip address and port to listen to for incoming data
13 $address = '192.168.0.100';
14 $port = 255;
15
16 // create a server-side SSL socket, listen for/accept incoming communication
17 $sock = socket_create(AF_INET, SOCK_STREAM, 0);
18 socket_bind($sock, $address, $port) or die('Could not bind to address');
19 socket_listen($sock);
20 $client = socket_accept($sock);
21
22 // read input data from client in 1024 byte blocks until end of message
23 do {
24     $input = "";
25     $input = socket_read($client, 1024);
26     $data .= $input;
27 } while($input != "");
28 ?>
29
```

30 **[0039]** In some implementations, the server may parse the obtained completed
31 prepaid account application form from the transferee client, and extract the application
32 data from the completed prepaid account application form. Based on the application
33 data, the server may generate an application screening request, e.g., 426. In some
34 implementations, the server may generate a request to perform a security and/or credit
35 check on the applicant. For example, the server may generate an applicant screening
36 request, e.g., 427, for a credit check service such as Equifax®. In such an example, the

1 server may provide a HTTP(S) POST message to a screening server, e.g., 407, including
 2 applicant details extracted from the completed prepaid account application. An
 3 example HTTP(S) POST message including an applicant screening request that may be
 4 sent by the PAFT server to a screening server is provided below:

```

5 POST /applicantscreen.php HTTP/1.1
6 Host: www.screeningserver.com
7 Content-Type: Application/XML
8 Content-Length: 1151
9 <?XML version = "1.0" encoding = "UTF-8"?>
10 <request>
11   <request_header interface_version_no="1.0" dtd_version_no="1.0">
12     <client_reference>A236269</client_reference>
13   </request_header>
14   <service_request id="201102091121">
15     <consumer_bureau_service>
16       <consumer_bureau_request domain="****" usercode="****" password="****"
17       operator_id="0000-0044">
18         <consumer_search search_type="SE" opt_in_out_flag="I"
19         associate_jointapp_flag="">
20           <scoring qcb_option="1" same_company_insight="">
21             <score_product code="RNISF02"/>
22           </scoring >
23         <data_types>
24           <data_type id="ADO" maximum="00" />
25           <data_type maximum="00" />
26           <data_type id="CJR" maximum="00" />
27           <data_type maximum="00" />
28           <data_type maximum="00" />
29           <data_type maximum="00" />
30           <data_type maximum="00" />
31           <data_type maximum="00" />
32         </data_types>
33         <display_requirements applicant_data="Y" associate_data="Y"
34         attributable_data="N" potential_associate_data="N" family_data="N" />
35         <applicant_details title="" forename="John" second_name="Q."
36         surname="Public">
37           <address_details time_at_address="" address_match_flag="" address="123 Green
38           Blvd. Apt 58, Terrace Heights CA 98765"></address_details>
39         </applicant_details>
40       </consumer_search>
41     </consumer_bureau_request>
42   </consumer_bureau_service>
43 </service_request>
44 </request>
45

```

46 **[0040]** Upon receiving the applicant screening request, the screening server may
 47 process the applicant screening request, and may generate an applicant screening
 48 report. For example, the screening server may determine that the applicant fails the
 49 screening tests for obtaining a prepaid account with the PAFT. In such an example, the

1 screening server may provide an applicant screening failure report, e.g., 428a. If the
 2 screening server determines that the applicant passed the screening tests for obtaining a
 3 prepaid account with the PAFT, the screening server may respond to the applicant
 4 screening request with an applicant screening success report, e.g., 428b. For example,
 5 the screening server may provide a HTTP(S) POST message to a PAFT server including
 6 an applicant screening report indicating success or failure on the part of the applicant to
 7 pass the screening tests. An example HTTP(S) POST message including an applicant
 8 screening report is provided below:

```

9  POST /screenresults.php HTTP/1.1
10  Host: www.paft.com
11  Content-Type: Application/XML
12  Content-Length: 1151
13  <?XML version = "1.0" encoding = "UTF-8"?>
14  <response>
15    <response_header>
16      <client_reference> A236269</client_reference>
17    </response_header>
18    <service_response id="201102091121" success_flag="1">
19      <consumer_bureau_service>
20        <consumer_bureau_response>
21          <header error_code="00" error_message="" reference=""/>
22          <non_address>
23            <supplied_address id="ADS" sequence_number="1" noc_indicator="N"/>
24            <score id="SCO" label="RNISF02" sign="+" score="528"/>
25          </non_address>
26          <address_details sequence_number="1" match_indicator="L">
27            <address_matched id="ADO" address_key="58150004076" house_name=""
28              house_number="123" street_1="GREEN BLVD" street_2="Apt 58" district=""
29              posttown="TERRACE HEIGHTS" county="LOS LOBOS" postcode="98765"
30              address_type="HOME"/>
31            <electoral_roll id="ELR" name_match_indicator="A" title="" forename="John"
32              second_name="Q" surname="Public" date_of_birth="" period="07-10"
33              junior_senior=""/>
34            <insight id="INR" name_match_indicator="C" title="MR." forename="John"
35              second_name="Q" surname="PUBLIC" date_of_birth="YYYY-MM-DD"
36              company_class="FN" account_type="03" account_number="" start_date="2005-03-
37              01" end_date="2006-11-12" update_date="2011-02-09" credit_limit="0"
38              current_balance_sign="+" current_balance="0" start_balance="0"
39              default_balance="0" credit_terms="275" repayment_period="300"
40              payment_frequency="M" payment_history="S" delinquent_date=""
41              client_number="" quality_flags="" quality_indicator_1=""
42              quality_indicator_2="" credit_card_payment_amount_sign=""
43              credit_card_payment_amount="" credit_card_previous_statement_balance_sign=""
44              credit_card_previous_statement_balance="" credit_card_cash_advance_value=""
45              credit_card_cash_advance_count="" credit_card_limit_change_flag=""
46              credit_card_minimum_payment_flag="" credit_card_promotional_rate_flag=""/>
47          </address_details>
48        </consumer_bureau_response>
49      </consumer_bureau_service>

```

1 </service_response>
2 </response>
3

4 **[0041]** Upon receiving the applicant screening report, the PAFT server may
5 determine whether the transferee user applicant passed the screening tests. If the
6 applicant failed the screening tests, the server may generate an application denial
7 notification, e.g., 429a. For example, the server may generate a HTTP(S) POST message
8 similar to the example described above. The server may provide the application denial
9 notification to the transferee client, and the transferee client may render, e.g., 430a, and
10 display, e.g., 431a, the application denial notification for the transferee user.

11 **[0042]** In some implementations, the server may determine that the transferee
12 user applicant passed the screening tests performed by the screening server based on
13 parsing the applicant screening report. In such implementations, the server may
14 proceed with creation of a new prepaid account for the transferee user. For example,
15 the server may determine whether the new prepaid account can be hosted locally, or if
16 the prepaid account should be hosted at a different bank/branch/server. For example,
17 the server may obtain the prepaid account preferences from the transferor user's
18 prepaid account application invitation request and/or from the transferee user's
19 completed prepaid application. If the new prepaid account can be hosted locally, the
20 server may generate a new user profile and/or user account database record(s)
21 reflecting a new prepaid account for the transferee user, e.g., 429b. If the prepaid
22 account is required to be hosted on another server, e.g., bank/branch server 408, the
23 PAFT server may generate a prepaid account opening request message, e.g., 430b, for
24 the bank/branch server. For example, the PAFT server may generate a HTTP(S) GET

1 message including a prepaid account opening request message, similar to the example
2 provided below:

```

3 GET /newaccountrequest.php HTTP/1.1
4 Host: www.bankbranch.com
5 Content-Type: Application/XML
6 Content-Length: 894
7 <?XML version = "1.0" encoding = "UTF-8"?>
8 <new_account_request>
9   <request_id>VBUQQW9M,</request_id>
10  <request_timestamp>2011-02-10 08:27:13</request_timestamp>
11  <requestor_ID>manager@paft.com for transferee@paft.com</requestor_ID>
12  <account_details>
13    <account_holder>individual</account_holder>
14    <account_type>prepaid debit</account_type>
15    <first_name>John</first_name>
16    <last_name>Public</last_name>
17    <contact_type>email</contact_type>
18    <contact_info>john.q.public@gmail.com</contact_info>
19    <opening_balance>$250.00</opening_balance>
20  </account_details>
21 </new_account_request>
22

```

23 **[0043]** In some implementations, the bank/branch server may require additional
24 information on the transferee user, invitation request, and/or the like, and may request
25 additional user information, e.g., additional user information request 432. For example,
26 the bank/branch server may provide to the PAFT server a HTTP(S) POST message
27 similar to the examples above. The PAFT server may parse any obtained additional user
28 information request(s), and obtain from local storage, database(s) and/or any PAFT-
29 affiliated entities and/or components the additional information requested, e.g., 433.
30 The PAFT server may in response generate, e.g., 434, an additional user information
31 message, for example, in the form of a HTTP(S) POST message similar to the examples
32 above, and provide the message to the bank/branch server. In some implementations,
33 the bank/branch server may create a new prepaid account for the transferee user based
34 on the information provided by the PAFT server, e.g., 435. The bank/branch server may
35 provide an account issue message, e.g., 436, to notify the PAFT server and/or other

1 PAFT entites and/or components that a prepaid account has been created at the
2 bank/branch server for the transferee user.

3 **[0044]** In some implementations, the PAFT server may, upon creating the prepaid
4 account for the transferee server (e.g., locally, or at another server such as the
5 bank/branch server), generate a prepaid account link data record that provides a link
6 between the prepaid accounts of the transferor and transferee user. The server may also
7 generate, e.g., 437 a transfer schedule that may specify funds transfer attributes between
8 the transferor and transferee prepaid accounts, such attributes including, but not
9 limited to: number of transfers, transfer dates, transfer frequency, transfer intervals,
10 transfer amounts, transfer notification settings, and/or the like. An example transfer
11 schedule encoded according to the XML format is provided below:

```
12 <?XML version = "1.0" encoding = "UTF-8"?>
13 <transfer_schedule>
14   <timestamp>2011-02-10 08:27:15</timestamp>
15   <transferor_ID>transferor@paft.com</transferor_ID>
16   <transferee_ID>transferee@paft.com</transferee_ID>
17   <link_ID>45UIVBNYT9</link_ID>
18   <num_transfers>6</num_transfers>
19   <transfer_date>01 of mo</transfer_date>
20   <transfer_frequency>1 per 2 mo</transfer_frequency>
21   <transfer_interval>2 mo</transfer_interval>
22   <notify_settings>transferor transferee</notify_settings>
23 </transfer_schedule>
24
```

25 **[0045]** In some implementations, the server may store a user profile record, e.g.,
26 438, prepaid account link data record, e.g., 439, and prepaid account funds transfer
27 schedule record, e.g., 440, to database(s), e.g., users database 404c, links database
28 404d, and schedules database 404e. For example, the server may issue PHP/SQL
29 commands similar to the example below to store the user profile, prepaid account link
30 data, and prepaid account funds transfer schedule records in database(s):

```
31 <?PHP
32 header('Content-Type: text/plain');
33 mysql_connect("254.92.185.103",$DBserver,$password); // access database server
```

```

1
2 // write user profile data into users database
3 mysql_select("USERS.SQL"); // select database to append
4 mysql_query("INSERT INTO UserTable (timestamp, userID, clientIP, usertype,
5     firstname, lastname, contacttype, contactinfo, altcontactinfo,
6     clienttype,ssn, dob, creditreport, accountnumbers)
7 VALUES (time(), $user_ID, $client_IP, $user_type, $first_name, $last_name,
8     $contact_type, $contact_info, $alt_contact_info, $client_type,$ssn, $dob,
9     $credit_report, $account_num_list)"); // add data to table in database
10 mysql_close("USERS.SQL"); // close connection to database
11
12 // write prepaid account link data into link database
13 mysql_select("LINKS.SQL"); // select database to append
14 mysql_query("INSERT INTO LinkTable (timestamp, link_ID, bidirectional_flag,
15     transferor_ID, transferor_account_num, transferor_bank, transferor_branch,
16     transferor_ABA, transferee_ID, transferee_account_num, transferee_bank,
17     transferee_branch, transferee_ABA)
18 VALUES (time(), $link_ID, $bi_flag, $transferor_ID, $transferor_account_num,
19     $transferor_bank, $transferor_branch, $transferor_ABA, $transferee_ID,
20     $transferee_account_num, $transferee_bank, $transferee_branch,
21     $transferee_ABA"); // add data to table in database
22 mysql_close("LINKS.SQL"); // close connection to database
23
24 // write prepaid account funds transfer schedule data into schedules database
25 mysql_select("SCHEDULES.SQL"); // select database to append
26 mysql_query("INSERT INTO ScheduleTable (timestamp, userID, schedule_data)
27 VALUES (time(), $transferor_ID, $transfer_schedule_xml)"); // add data to table
28     in database
29 mysql_close("SCHEDULES.SQL"); // close connection to database
30 ?>
31

```

32 **[0046]** In some implementations, the server may provide an account issue
33 notification, e.g., 441, for the transferee (and/or transferor) user(s). For example, the
34 server may provide a HTTP(S) POST message similar to the examples above. The
35 client(s) may render, e.g., 442, the account issue notification for presentation to the
36 user(s). For example, the client(s) may render a webpage, electronic message, text/SMS
37 message, buffer a voicemail, emit a ring tone for the user(s) to answer a phone and/or
38 play an audio message, etc. The client(s) may then present the account issue
39 notification for the user(s), e.g., 443. For example, the client(s) may provide output
40 including, but not limited to: sounds, music, audio, video, images, tactile feedback,
41 vibration alerts (e.g., on vibration-capable client devices such as a smartphone etc.),
42 and/or the like.

1 **[0047]** FIGURES 5A-D show logic flow diagrams illustrating example aspects of
2 registering a new prepaid account for a user in some embodiments of the PAFT, e.g., a
3 Prepaid Account Registration (PAR) component 500. In some implementations, the
4 transferee user may desire to accept the invitation offer by the server to open a prepaid
5 account with the PAFT. For example, the transferee user may provide invitation
6 acceptance input, e.g., 501, to the transferee client to convey acceptance of the
7 invitation. In response to the transferee user input, the transferee client may generate a
8 prepaid account invitation acceptance message, e.g., 502, for the PAFT server. Upon
9 obtaining, e.g., 503, the prepaid account invitation acceptance message from the
10 transferee client, the server may parse the invitation acceptance to extract the included
11 invitation acceptance data, e.g., 504. For example, the server may extract data fields
12 including, but not limited to: request_id, timestamp, transferor_id, client_IP,
13 transferee_type, first_name, last_name, contact_type, contact_info, client_type,
14 account_params_list, same_bank_flag, same_branch_flag, persistent_link_flag,
15 transfer_param_list, on_schedule_flag, one_time_flag, amount, instant_transfer_flag,
16 custom_invitation, transferee_details_list, transferee_client_IP, client_type,
17 client_MAC, client_IP, presentation_format, pixel_count, resolution, screen_size,
18 audio_fidelity, hardware_settings_list, software_compatibilities_list, installed_apps_
19 list, and/or the like. Based on the extracted data fields, the server may determine the
20 type of transferee client, e.g., 505. For example, the server may determine whether the
21 transferee client is a desktop computer, tablet computer, mobile phone accessing email,
22 text voice messages, etc. For example, the server may determine the attributes of the
23 client device such as the screen resolution, screen size, audio/video capabilities,
24 hardware configuration, software configuration, applications installed on the client,

1 and/or the like. The server may also extract the unique ID of the invitation from the
2 invitation acceptance, e.g., 506. Based on the extracted invitation data, the server may,
3 in some implementations, generate a query for the type of the invitation, e.g., 507, and
4 provide the query to a database, e.g., an invitations database. For example, the server
5 may query for invitation attributes such as, but not limited to: transferee_type,
6 account_params_list, account_mode, account_type, account_expiry, preferred_bank,
7 preferred_branch, and/or the like. For example, using such data fields server may
8 determine such details as whether the invitation is for an individual/merchant for
9 opening of a credit/debit account, whether the account is
10 prepaid/savings/checking/money market account etc. In response to the server's query,
11 the database may provide the requested information, e.g., 508.

12 **[0048]** In some implementations, the server may generate a query for a prepaid
13 account application form using the invitation type and client type of the transferee user.
14 The server may generate a query for a prepaid application form, e.g., 509, that is
15 customized to the transferee client and/or transferee user, and issue the query to a
16 database, e.g., a forms database. In response to the prepaid account application form
17 query, the database may provide a prepaid account application form, e.g., 510, that is
18 customized to the attributes and/or preferences of the transferee client and/or
19 transferee user. The server may provide the obtained prepaid account application form,
20 e.g., 511, to the transferee client. Upon obtaining the prepaid account application, the
21 transferee client may render/execute, e.g., 512, the prepaid account application, and
22 present the prepaid account application for the transferee user, e.g., 513.

1 **[0049]** In some implementations, the transferee user may provide application
2 form input, e.g., 514, to the transferee client. Using the user application form input, the
3 client may generate, e.g., 515, a completed prepaid account application. The transferee
4 client may provide the completed prepaid account application for the server. In some
5 implementations, the server may obtain the completed prepaid account application
6 form from the transferee client, e.g., 516, and parse, e.g., 517, the completed application
7 to extract the application data from the completed prepaid account application form.
8 For example, the server may obtain fields such as, but not limited to: applicant_ID,
9 applicane_firstname, applicant_lastname, dob, ssn, credit_check_ok_flag, address_
10 line1, address_line2, zipcode, city, state, account_params_list, account_mode,
11 account_type, account_expiry, bank_name, branch_name, and/or the like. Based on
12 the application data, the server may generate an application screening request, e.g., 518.
13 In some implementations, the server may generate a request to perform a security
14 and/or credit check on the applicant. Upon receiving the applicant screening request,
15 e.g., 519, the screening server may process, e.g., 520, the applicant screening request,
16 and may generate an applicant screening report, e.g., 521. For example, the screening
17 server may determine whether the applicant passes or fails the screening tests for
18 obtaining a prepaid account with the PAFT. Upon obtaining and parsing, e.g., 522, the
19 applicant screening report, the PAFT server may determine whether the transferee user
20 applicant passed the screening tests, e.g., 523. If the applicant failed the screening tests,
21 e.g., 523, option “No,” the server may generate an application denial notification, e.g.,
22 543. The server may provide the application denial notification to the transferee client,
23 and the transferee client may render, e.g., 545, and display, e.g., 546, the application
24 denial notification for the transferee user.

1 **[0050]** In some implementations, the server may determine that the transferee
2 user applicant passed the screening tests performed by the screening server based on
3 parsing the applicant screening report, e.g., 523, option “Yes.” In such
4 implementations, the server may proceed with creation of a new prepaid account for the
5 transferee user. The server may generate a user profile for the transferee user applicant,
6 e.g., 524. The server may also determine whether the new prepaid account can be
7 hosted locally, or if the prepaid account should be hosted at a different
8 bank/branch/server, e.g., 525. For example, the server may obtain the prepaid account
9 preferences from the transferor user’s prepaid account application invitation request
10 and/or from the transferee user’s completed prepaid application. If the new prepaid
11 account can be hosted locally, e.g., 525, option “No,” the server may generate a new user
12 profile and/or user account database record(s) reflecting a new prepaid account for the
13 transferee user, e.g., 526. If the prepaid account is required to be hosted on another
14 server, e.g., 525, option “Yes,” the PAFT server may generate a prepaid account opening
15 request message, e.g., 527, for the bank/branch server. The PAFT server may provide
16 the prepaid account opening request message to the bank/branch server where the
17 account should be opened, based on the applicant data extracted from the applicant’s
18 completed prepaid account application form. In some implementations, the
19 bank/branch server may, upon obtaining the prepaid account opening request, e.g., 528,
20 parse the prepaid account opening request, e.g., 529, and extract request data. For
21 example, the bank/branch server may extract fields including, but not limited to:
22 applicant_firstname, applicant_lastname, applicant_address_line1, applicant_
23 address_line2, applicant_clear_flag, and/or the like. If the bank/branch server
24 determines that it requires additional information, e.g., 530, for example on the

1 transferee user, invitation request, and/or the like, the bank/branch server may
2 generate an additional user information request, e.g., 531, and provide the request for
3 additional user information to the PAFT server. The PAFT server may parse, e.g., 532,
4 the additional user information request from the bank/branch server, and determine the
5 additional user information requested, e.g., 533. The PAFT server may generate an
6 additional user information message, e.g., 534, using, for example, information obtained
7 from local storage, database(s) and/or any PAFT-affiliated entities and/or components.
8 The PAFT server may provide the message to the bank/branch server. Upon obtaining
9 the additional user information message, e.g., 535, the bank/branch server may, in some
10 implementations, create a new prepaid account for the transferee user based on the
11 information provided by the PAFT server, e.g., 536. The bank/branch server may
12 generate an account issue message, e.g., 537, and provide the account issue message to
13 notify the PAFT server and/or other PAFT entities and/or components that a prepaid
14 account has been created at the bank/branch server for the transferee user.

15 **[0051]** In some implementations, the PAFT server may, upon obtaining
16 notification (and/or creating) the prepaid account for the transferee server (e.g., locally,
17 or at another server such as the bank/branch server), generate a prepaid account link
18 data record that provides a link between the prepaid accounts of the transferor and
19 transferee user, e.g., 538. The server may also generate, e.g., 539, a prepaid account
20 funds transfer schedule that may specify funds transfer attributes between the
21 transferor and transferee prepaid accounts, such attributes including, but not limited to:
22 number of transfers, transfer dates, transfer frequency, transfer intervals, transfer
23 amounts, transfer notification settings, and/or the like. In some implementations, the
24 server may also generate a user profile record, e.g., 540. The server may provide the

1 user account data record, prepaid account link data record, and/or prepaid account
2 funds transfer schedule record to database(s), e.g., users database, links database,
3 and/or schedules database. In some implementations, the server may provide an
4 account issue notification, e.g., 543, for the transferee (and/or transferor) user(s). The
5 client(s) may render, e.g., 545, the account issue notification for presentation to the
6 user(s). The client(s) may then present the account issue notification for the user(s),
7 e.g., 546.

8 **[0052]** FIGURES 6A-B show data flow diagrams illustrating an example
9 procedure to transfer funds from a prepaid account of a user to a prepaid account of
10 another user in some embodiments of the PAFT. In some implementations, various
11 inputs into a server, e.g. PAFT server 603, may trigger a prepaid account funds transfer
12 from a transferor user's prepaid account to a transferee user's prepaid account.

13 **[0053]** As an example, the server may spontaneously initiate a prepaid accounts
14 funds transfer upon creation of a new prepaid account (e.g., a transferor and/or
15 transferee user account) by the PAFT and/or affiliated entities and/or components.

16 **[0054]** As another example, a transferor user, e.g., 601, may provide a payment
17 order input, e.g., 611a, into a transferor client, e.g., 602, requesting that funds be
18 transferred from the transferor user's prepaid account into the transferee user's prepaid
19 account. The transferor client, may utilize the transferor user's payment order input to
20 generate a prepaid account payment order, e.g., 612a. The transferor client may provide
21 the prepaid account payment order, e.g., 613a, to the server, e.g., 603, which may trigger
22 the server to execute transfer of funds as requested by the transferor user.

1 **[0055]** As another example, a transferee user, e.g., 606, may provide a receivables
2 request input, e.g., 611b, into a transferee client, e.g., 605, requesting that funds owed be
3 transferred from the transferor user's prepaid account into the transferee user's prepaid
4 account. The transferee client may utilize the transferee user's receivables request input
5 to generate a prepaid account receivables request, e.g., 612b. The transferee client may
6 provide the prepaid account receivables request, e.g., 613b, to the server, e.g., 603,
7 which may trigger the server to execute transfer of funds as requested by the transferee
8 user.

9 **[0056]** As another example, the server, e.g., 603, may query a database, e.g.,
10 schedules database 604e, for a transfer schedule, e.g., 611c. For example, the server may
11 query the database on a continuous, periodically, on-demand, on-trigger and/or like
12 basis. In response to the query, the database may provide a transfer schedule, e.g., 612c.
13 The server may obtain the transfer schedule, and parse it to extract schedule data
14 pertaining to transfers between the transferor user, e.g., 601, and the transferee user,
15 e.g., 606. The server may analyze the data, e.g., 613c, to determine whether a transfer
16 should be initiated between the prepaid account of the transferor user and the prepaid
17 account of the transferee user.

18 **[0057]** In general, it is contemplated that a variety of events and/or changes in
19 data at any of the PAFT components and/or affiliated entities may trigger the PAFT to
20 perform prepaid account funds transfer from a transferor user's prepaid account to a
21 transferee user's prepaid account.

22 **[0058]** In some implementations, the PAFT server may obtain a trigger, such as
23 those described above, to perform prepaid account funds transfer from a transferor

1 user's prepaid account to a transferee user's prepaid account. In response to the trigger,
 2 the server may generate a query for a link data record, and provide the query, e.g., 614,
 3 to a database, e.g., links database 604d. For example, the server may issue PHP/SQL
 4 comands similar to the example provided below:

```

5  <?PHP
6  header('Content-Type: text/plain');
7  function link_data_query($template_type, $DBserver, $password) {
8  mysql_connect("254.92.185.102",$DBserver,$password); // access database server
9  mysql_select_db("LINKS.SQL"); // select database table to search
10 //create query for invitation template
11 $query = "SELECT link_ID, bidirectional_flag, transferor_ID,
12         transferor_account_num, transferor_bank, transferor_branch, transferor_ABA,
13         transferee_ID, transferee_account_num, transferee_bank, transferee_branch,
14         transferee_ABA FROM LinksTable WHERE transferor_ID LIKE '%' $transferor AND
15         transferee_ID LIKE '%' $transferee";
16 $result = mysql_query($query); // perform the search query
17 mysql_close("LINKS.SQL"); // close database access
18 return $result; // return search results
19 ?>
20
```

21 **[0059]** The database may provide the transfer link data record, e.g., 615, in
 22 response to obtaining the query from the server. In some implementations, the server
 23 may generate a query for one or more user profile data records (e.g., the user profile
 24 data records of the transferor user and/or transferee user). The server may provide the
 25 user profile data query, e.g., 616, to a database, e.g., users database 604c. For example,
 26 the server may issue PHP/SQL commands similar to the example above to the users
 27 database. In response to the query, the users database may provide the requested user
 28 profile data records.

29 **[0060]** In some implementations, using the link data record and/or the user
 30 profile data records, the server may initiate transfer for funds from the prepaid account
 31 of the transferor user to the prepaid account of the transferee user. For example, the
 32 server may determine whether the prepaid accounts of the transferor user and the
 33 transferee user are both stored locally. If the prepaid accounts are both stored locally,

1 the server may debit the prepaid account of the transferor user, and credit the prepaid
 2 account of the transferee user, e.g., 618. Alternatively, in another example, if a prepaid
 3 account of the transferor user and/or transferee user is not stored locally, the server
 4 may generate one or more transaction request messages using the link data and/or user
 5 profile data, and provide the transaction request messages, e.g., 619, to one or more
 6 bank/branch servers, e.g., 608. For example, the server may provide a HTTP(S) POST
 7 message for the bank/branch server similar to the example provided below:

```

8  POST /transfers.php HTTP/1.1
9  Host: www.bankbranch.com
10 Content-Type: Application/XML
11 Content-Length: 569
12 <?XML version = "1.0" encoding = "UTF-8"?>
13 <account_transfer_request>
14   <request_id>VQ358GFBE</request_id>
15   <originator>manager@paft.com</originator>
16   <auth>A1834597QG26ABF49T</auth>
17   <timestamp>2011-02-09 10:27:13</timestamp>
18   <action_type>debit</action_type>
19   <action_amount>$250.00</action_amount>
20   <account_details>
21     <user_ID>transferor@paft.com</user_ID>
22     <user_type>individual</user_type>
23     <first_name>John</first_name>
24     <last_name>Public</last_name>
25     <account_num>9087612345</account_num>
26     <bank>Chasing Bank</bank>
27     <branch>455</branch>
28     <ABA>12399876</ABA>
29   </account_details>
30 </account_transfer_request>
31

```

32 **[0061]** In response, the bank/branch server may process the transaction request,
 33 e.g., 620, and provide a transaction completion message, e.g., 621, for the PAFT server.
 34 Upon obtaining the transaction completion message(s) for the bank/branch server(s)
 35 involved in the prepaid account funds transfer, the PAFT server may generate
 36 transaction notification message(s), and provide the transaction notification message(s),
 37 e.g., 623a-b, to the transferor client and/or transferee client. For example, the server
 38 may provide a HTTP(S) POST message similar to the examples above. The client(s) may
 39 render, e.g., 442, the transaction notification message(s) for presentation to the user(s).

1 For example, the client(s) may render a webpage, electronic message, text/SMS
2 message, buffer a voicemail, emit a ring tone for the user(s) to answer a phone and/or
3 play an audio message, etc. The client(s) may then present the transaction notification
4 message(s) for the user(s), e.g., 443. For example, the client(s) may provide output
5 including, but not limited to: sounds, music, audio, video, images, tactile feedback,
6 vibration alerts (e.g., on vibration-capable client devices such as a smartphone etc.),
7 and/or the like.

8 **[0062]** FIGURE 7 shows a logic flow diagram illustrating example aspects of
9 implementing a transferor user-initiated trigger for prepaid account funds transfer in
10 some embodiments of the PAFT, e.g., a Transferor-Initiated Funds Transfer Trigger (Tr-
11 FTT) component 700. In some implementations, a transferor user may provide
12 payment order input, e.g., 701, into a transferor client. In response, the transferor client
13 may generate a prepaid account payment order, e.g., 702. The transferor client may
14 provide the generated prepaid account payment order to the PAFT server. In some
15 implementations, the server may, upon obtaining the order, e.g., 703, parse the obtained
16 prepaid account payment order, e.g., 704. The server may extract the details of the
17 prepaid account payment order, and determine, e.g., 705, whether funds should be
18 transferred from a prepaid account of the transferor user to the prepaid account of the
19 transferee user, based on the details provided in the prepaid account transfer order. For
20 example, server may perform security checks to ensure that the order is issued by an
21 authorized entity, that the amount of the transfer passes fraud checks, and/or the like.
22 The server may also determine whether the account information, amount information,
23 etc. are provided properly in the prepaid account transfer order. If the server
24 determines that the prepaid account funds transfer must be performed, e.g., 706, option

1 “Yes,” the server may generate a prepaid account funds transfer trigger, e.g., 710. In
2 some implementations, the server may push the generated prepaid account funds
3 transfer trigger to a queue of prepaid account funds transfer triggers. In some
4 implementations, the server may determine that the prepaid account funds transfer
5 should not take place, e.g., 706, option “No.” The server may, in such implementations,
6 generate a payment order cancel notice, e.g., 707, and provide the generated payment
7 order cancel notice for the transferor client. The transferor client may render the
8 obtained payment order cancel notice, e.g., 708, and provide the rendered payment
9 order cancel notice for the transferor user, e.g., 709. For example, the client(s) may
10 provide output including, but not limited to: sounds, music, audio, video, images, tactile
11 feedback, vibration alerts (e.g., on vibration-capable client devices such as a smartphone
12 etc.), and/or the like.

13 **[0063]** FIGURE 8 shows a logic flow diagram illustrating example aspects of
14 implementing a transferee user-initiated trigger for prepaid account funds transfer in
15 some embodiments of the PAFT, e.g., a Transferee-Initiated Funds Transfer Trigger (Te-
16 FTT) component 800. In some implementations, a transferee user may provide
17 receivables request input, e.g., 801, into a transferee client. In response, the transferee
18 client may generate a prepaid account receivables request, e.g., 802. The transferee
19 client may provide the generated prepaid account receivables request to the PAFT
20 server. In some implementations, the server may, upon obtaining the prepaid account
21 receivables request, e.g., 803, parse the obtained prepaid account receivables request,
22 e.g., 804. The server may extract the details of the prepaid account receivables request,
23 and determine, e.g., 805, whether funds should be transferred from a prepaid account of
24 the transferor user to the prepaid account of the transferee user, based on the details

1 provided in the prepaid account receivables request. For example, server may perform
2 security checks to ensure that the order is issued by an authorized entity, that the
3 amount of the transfer passes fraud checks, and/or the like. The server may also
4 determine whether the account information, amount information, etc. are provided
5 properly in the prepaid account receivables request. If the server determines that the
6 prepaid account funds transfer must be performed, e.g., 806, option “Yes,” the server
7 may generate a prepaid account funds transfer trigger, e.g., 810. In some
8 implementations, the server may push the generated prepaid account funds transfer
9 trigger to a queue of prepaid account funds transfer triggers. In some implementations,
10 the server may determine that the prepaid account funds transfer should not take place,
11 e.g., 806, option “No.” The server may, in such implementations, generate an account
12 receivables rejection notice, e.g., 807, and provide the generated account receivables
13 rejection notice for the transferee client. The transferee client may render the obtained
14 account receivables rejection notice, e.g., 808, and provide the rendered account
15 receivables rejection notice for the transferee user, e.g., 809. For example, the client(s)
16 may provide output including, but not limited to: sounds, music, audio, video, images,
17 tactile feedback, vibration alerts (e.g., on vibration-capable client devices such as a
18 smartphone etc.), and/or the like.

19 **[0064]** FIGURE 9 shows a logic flow diagram illustrating example aspects of
20 implementing a server-initiated trigger for prepaid account funds transfer in some
21 embodiments of the PAFT, e.g., a Server-Initiated Funds Transfer Trigger (S-FTT)
22 component 900. In some implementations, the PAFT server may generate, e.g., 901, a
23 request for a transfer schedule for specified transferor and transferee user(s), and may
24 provide the request for a database, e.g., schedules database. For example, the server

1 may query the database on a continuous, periodically, on-demand, on-trigger and/or
2 like basis. In response to the request, the database may provide the requested transfer
3 schedule, e.g., 902. The server may obtain the transfer schedule, e.g., 903, and parse it
4 to extract schedule data pertaining to transfers between the transferor user(s) and
5 transferee user(s), e.g., 904. The server may analyze the data, e.g., 905, to determine
6 whether a transfer should be initiated between the prepaid account of the transferor
7 user and the prepaid account of the transferee user. If the server determines that a
8 transfer should be initiated between the prepaid account of the transferor user and
9 prepaid account of the transferee user, e.g., 906, option "Yes," the server may generate a
10 prepaid account funds transfer trigger, e.g., 907. If, however, the server determines that
11 a transfer should not be initiated between the prepaid account of the transferor user and
12 prepaid account of the transferee user, e.g., option 906, option "No," the server may
13 reset the procedure, and repeat the procedure if necessary.

14 **[0065]** FIGURES 10A-C show logic flow diagrams illustrating example aspects of
15 transferring funds from a prepaid account of a user to a prepaid account of another user
16 in some embodiments of the PAFT, e.g., a Prepaid Account Funds Transfer Processing
17 (PA-FTP) component 1000. In some implementations, the PAFT server may obtain a
18 prepaid account funds transfer trigger, e.g., 1001, to transfer funds between a transferor
19 prepaid user account and a transferee prepaid user account. For example, the server
20 may obtain the trigger from a queue of prepaid account funds transfer triggers, from a
21 Tr-FTT component, e.g., 700, a Te-FTT component, e.g., 800, a S-FTT component, e.g.,
22 900, and/or the like PAFT entity and/or component. Upon obtaining the prepaid
23 account funds transfer trigger, the server may determine an amount of the funds to be
24 transferred from the prepaid account of the transferor user to the prepaid account of the

1 transferee user, e.g., 1002. For example, the server may parse a payment order, a
2 receivables request, a transfer schedule, and/or the like data records to obtain an
3 amount of funds to be transferred. Upon determining the transfer amount, the server
4 may generate a transfer link query for link data pertaining to a link between the prepaid
5 accounts of the transferor user and the transferee user, e.g., 1003. The server may
6 provide the transfer link query to a database, e.g., a links database. In response, the
7 database may provide the requested link data, e.g., 1004. In some implementations, the
8 server may generate a query for user profile data of the transferor and/or transferee
9 user(s), e.g., 1005. The server may issue the user profile data queries to a database, e.g.,
10 a users database. In response to receiving the user profile data queries, the database
11 may provide the requested user profile data, e.g., 1006. Upon obtaining the link data
12 and/or the user data from the database(s), the server may determine whether the
13 prepaid accounts of the transferor and transferee users are stored locally, e.g., 1007.

14 **[0066]** If the server determines that both prepaid accounts are stored locally, e.g.,
15 1008, option “Yes,” the server may debit the prepaid account of the transferor user by
16 the determined amount of the prepaid account funds transfer, and credit the prepaid
17 account of the transferee user by the determined amount of the prepaid account funds
18 transfer, e.g., 1009. In some implementations, the server may determine that both
19 prepaid accounts are not stored locally, e.g., 1008, option “No.” The server may, in such
20 implementations, generate one or more transaction request message(s) to perform
21 prepaid account actions (e.g., credit and/or debit). The server may provide the
22 transactio request message(s) to one or more bank/branch servers. The bank/branch
23 server, upon obtaining the transaction request message, e.g., 1011, may perform the
24 transaction according to the request message, e.g., 1012. The bank/branch server may

1 generate a transaction completion message, e.g., 1013, and provide the transaction
2 completion message to the PAFT server.

3 **[0067]** In some implementations, the PAFT server may obtain an indication that
4 funds transfer has been successfully completed between the prepaid accounts of the
5 transferor user and transferee user. If the server determines that the funds transfer was
6 successful, e.g., 1014, option “Yes,” the server may generate a funds transfer notification
7 message for the transferor and/or transferee user, e.g., 1016. If the server determines
8 that the funds transfer was not successful, e.g., 1014, option “No,” the server may
9 generate a funds transfer failure notification message for the transferor and/or
10 transferee user, e.g., 1015. The server may provide the generated notifications for the
11 transferor and/or transferee clients. The client may render, e.g., 1017, the notifications,
12 and present, e.g., 1018, the notifications for the users. In some implementations, the
13 server may generate, e.g., 1019, updated transaction data, user profile data, and/or
14 schedule data records based on the prepaid account funds transfer (or failure thereof),
15 and provide the data records to database(s). The database(s), e.g., schedules
16 database(s) and/or users database(s), may store, e.g., 1020, the data records provided
17 by the server.

18 **[0068]** FIGURE 11 shows a user interface diagram illustrating example aspects of
19 managing prepaid account funds transfers in some embodiments of the PAFT. In some
20 implementations, the PAFT may enable a user to manage a prepaid account (e.g.,
21 manage transfer of funds to and from the prepaid account) via an interface. For
22 example, the user may be able to manage the prepaid account via a web browser
23 interface, e.g., 1101. In some implementations, the user may be able to download an

1 application (e.g., an iPhone® app, Android™ app, HP Palm OS app, Windows® Mobile
2 app, Blackberry® app, etc.) on to their client device, which may then provide an
3 interface for prepaid account management (see, e.g., 1112). The interface may provide
4 notifications of the status of one or more prepaid account, e.g., 1102. For example, the
5 interface may provide details such as, but not limited to: number of accounts, account
6 numbers, account balances, account transaction history, transaction history statistics
7 (e.g., number of transactions within a predetermined/flexible time window, frequency of
8 transactions, cumulative transaction amounts across time, accounts, and/or other users,
9 transactions by user, indications of amount of bidirectionality of transfers, etc.),
10 upcoming scheduled transfer, and/or the like. In some implementations, the
11 information such as those listed above may be provided on a single screen
12 simultaneously. In alternate implementations, the user interface may be designed as a
13 complex multiple-screen interface that may present detailed information across various
14 types of analyses using multiple screens. In some implementations, the interface may
15 provide a listing of contacts, e.g., 1103. For example, the contacts listing may include a
16 listing of others users with whom the user has engaged in prepaid account funds
17 transfers (e.g., as transferor and/or transferee). In some implementations, the user may
18 be able to activate a link (see, e.g., 1104a) to initiate a transfer with another user, and/or
19 to create a transfer schedule for prepaid account funds transfers with that user. For
20 example, one or more user(s) listed in the contacts listing may be selected
21 simultaneously (e.g., by clicking a single user, selecting multiple users and activating a
22 graphical user interface element for initiating/scheduling transfers with each of the
23 selected users simultaneously), which when activated, provides the user an interface
24 wherein the user can initiate and/or schedule prepaid account funds transfer(s) with the

1 selected user(s) (see, e.g., 1104b). In some implementations, the user interface may
2 include one or more graphical user interface elements to provide information on
3 upcoming scheduled transfers, e.g., graphical calendar 1105 and textual schedule listing
4 1106. The graphical user elements may provide indication, alarms, notifications, etc. of
5 due and/or upcoming transfers. In some implementations, the PAFT may send alerts,
6 notices, notifications, and/or the like to a contact of the user (e.g., pager number, mobile
7 number, email address, postal address) for the user.

8 **[0069]** In some implementations, the user interface may provide various elements
9 for inviting (see, e.g., 1107) other users to apply for obtaining new prepaid accounts with
10 the PAFT. For example, the user interface may provide a hyperlink, e.g., 1108, upon
11 activation of which the user may be provided with graphical user interface elements to
12 generate and send invitation requests for the PAFT. In various implementations, the
13 user may be able to invite other user to create prepaid accounts with the PAFT via text
14 message, e.g., 1109, phone, e.g., 1110, chat, social networking, e.g., 1111, and/or the like.

15 PAFT Controller

16 **[0070]** FIGURE 12 illustrates inventive aspects of a PAFT controller 1201 in a
17 block diagram. In this embodiment, the PAFT controller 1201 may serve to aggregate,
18 process, store, search, serve, identify, instruct, generate, match, and/or facilitate
19 interactions with a computer through various technologies, and/or other related data.

20 **[0071]** Typically, users, which may be people and/or other systems, may engage
21 information technology systems (e.g., computers) to facilitate information processing.
22 In turn, computers employ processors to process information; such processors 1203

1 may be referred to as central processing units (CPU). One form of processor is referred
2 to as a microprocessor. CPUs use communicative circuits to pass binary encoded signals
3 acting as instructions to enable various operations. These instructions may be
4 operational and/or data instructions containing and/or referencing other instructions
5 and data in various processor accessible and operable areas of memory 1229 (e.g.,
6 registers, cache memory, random access memory, etc.). Such communicative
7 instructions may be stored and/or transmitted in batches (e.g., batches of instructions)
8 as programs and/or data components to facilitate desired operations. These stored
9 instruction codes, e.g., programs, may engage the CPU circuit components and other
10 motherboard and/or system components to perform desired operations. One type of
11 program is a computer operating system, which, may be executed by CPU on a
12 computer; the operating system enables and facilitates users to access and operate
13 computer information technology and resources. Some resources that may be employed
14 in information technology systems include: input and output mechanisms through
15 which data may pass into and out of a computer; memory storage into which data may
16 be saved; and processors by which information may be processed. These information
17 technology systems may be used to collect data for later retrieval, analysis, and
18 manipulation, which may be facilitated through a database program. These information
19 technology systems provide interfaces that allow users to access and operate various
20 system components.

21 **[0072]** In one embodiment, the PAFT controller 1201 may be connected to and/or
22 communicate with entities such as, but not limited to: one or more users from user
23 input devices 1211; peripheral devices 1212; an optional cryptographic processor device
24 1228; and/or a communications network 1213.

1 **[0073]** Networks are commonly thought to comprise the interconnection and
2 interoperation of clients, servers, and intermediary nodes in a graph topology. It should
3 be noted that the term “server” as used throughout this application refers generally to a
4 computer, other device, program, or combination thereof that processes and responds to
5 the requests of remote users across a communications network. Servers serve their
6 information to requesting “clients.” The term “client” as used herein refers generally to a
7 computer, program, other device, user and/or combination thereof that is capable of
8 processing and making requests and obtaining and processing any responses from
9 servers across a communications network. A computer, other device, program, or
10 combination thereof that facilitates, processes information and requests, and/or
11 furthers the passage of information from a source user to a destination user is
12 commonly referred to as a “node.” Networks are generally thought to facilitate the
13 transfer of information from source points to destinations. A node specifically tasked
14 with furthering the passage of information from a source to a destination is commonly
15 called a “router.” There are many forms of networks such as Local Area Networks
16 (LANs), Pico networks, Wide Area Networks (WANs), Wireless Networks (WLANs), etc.
17 For example, the Internet is generally accepted as being an interconnection of a
18 multitude of networks whereby remote clients and servers may access and interoperate
19 with one another.

20 **[0074]** The PAFT controller 1201 may be based on computer systems that may
21 comprise, but are not limited to, components such as: a computer systemization 1202
22 connected to memory 1229.

Computer Systemization

1
2 **[0075]** A computer systemization 1202 may comprise a clock 1230, central
3 processing unit (“CPU(s)” and/or “processor(s)” (these terms are used interchangeable
4 throughout the disclosure unless noted to the contrary)) 1203, a memory 1229 (e.g., a
5 read only memory (ROM) 1206, a random access memory (RAM) 1205, etc.), and/or an
6 interface bus 1207, and most frequently, although not necessarily, are all interconnected
7 and/or communicating through a system bus 1204 on one or more (mother)board(s)
8 1202 having conductive and/or otherwise transportive circuit pathways through which
9 instructions (e.g., binary encoded signals) may travel to effect communications,
10 operations, storage, etc. Optionally, the computer systemization may be connected to an
11 internal power source 1286; e.g., optionally the power source may be internal.
12 Optionally, a cryptographic processor 1226 and/or transceivers (e.g., ICs) 1274 may be
13 connected to the system bus. In another embodiment, the cryptographic processor
14 and/or transceivers may be connected as either internal and/or external peripheral
15 devices 1212 via the interface bus I/O. In turn, the transceivers may be connected to
16 antenna(s) 1275, thereby effectuating wireless transmission and reception of various
17 communication and/or sensor protocols; for example the antenna(s) may connect to: a
18 Texas Instruments WiLink WL1283 transceiver chip (e.g., providing 802.11n, Bluetooth
19 3.0, FM, global positioning system (GPS) (thereby allowing PAFT controller to
20 determine its location)); Broadcom BCM4329FKUBG transceiver chip (e.g., providing
21 802.11n, Bluetooth 2.1 + EDR, FM, etc.); a Broadcom BCM4750IUB8 receiver chip (e.g.,
22 GPS); an Infineon Technologies X-Gold 618-PMB9800 (e.g., providing 2G/3G
23 HSDPA/HSUPA communications); and/or the like. The system clock typically has a
24 crystal oscillator and generates a base signal through the computer systemization’s

1 circuit pathways. The clock is typically coupled to the system bus and various clock
2 multipliers that will increase or decrease the base operating frequency for other
3 components interconnected in the computer systemization. The clock and various
4 components in a computer systemization drive signals embodying information
5 throughout the system. Such transmission and reception of instructions embodying
6 information throughout a computer systemization may be commonly referred to as
7 communications. These communicative instructions may further be transmitted,
8 received, and the cause of return and/or reply communications beyond the instant
9 computer systemization to: communications networks, input devices, other computer
10 systemizations, peripheral devices, and/or the like. Of course, any of the above
11 components may be connected directly to one another, connected to the CPU, and/or
12 organized in numerous variations employed as exemplified by various computer
13 systems.

14 **[0076]** The CPU comprises at least one high-speed data processor adequate to
15 execute program components for executing user and/or system-generated requests.
16 Often, the processors themselves will incorporate various specialized processing units,
17 such as, but not limited to: integrated system (bus) controllers, memory management
18 control units, floating point units, and even specialized processing sub-units like
19 graphics processing units, digital signal processing units, and/or the like. Additionally,
20 processors may include internal fast access addressable memory, and be capable of
21 mapping and addressing memory 529 beyond the processor itself; internal memory may
22 include, but is not limited to: fast registers, various levels of cache memory (e.g., level 1,
23 2, 3, etc.), RAM, etc. The processor may access this memory through the use of a
24 memory address space that is accessible via instruction address, which the processor

1 can construct and decode allowing it to access a circuit path to a specific memory
2 address space having a memory state. The CPU may be a microprocessor such as:
3 AMD's Athlon, Duron and/or Opteron; ARM's application, embedded and secure
4 processors; IBM and/or Motorola's DragonBall and PowerPC; IBM's and Sony's Cell
5 processor; Intel's Celeron, Core (2) Duo, Itanium, Pentium, Xeon, and/or XScale;
6 and/or the like processor(s). The CPU interacts with memory through instruction
7 passing through conductive and/or transportive conduits (e.g., (printed) electronic
8 and/or optic circuits) to execute stored instructions (i.e., program code) according to
9 conventional data processing techniques. Such instruction passing facilitates
10 communication within the PAFT controller and beyond through various interfaces.
11 Should processing requirements dictate a greater amount speed and/or capacity,
12 distributed processors (e.g., Distributed PAFT), mainframe, multi-core, parallel, and/or
13 super-computer architectures may similarly be employed. Alternatively, should
14 deployment requirements dictate greater portability, smaller Personal Digital Assistants
15 (PDAs) may be employed.

16 **[0077]** Depending on the particular implementation, features of the PAFT may be
17 achieved by implementing a microcontroller such as CAST's R8051XC2 microcontroller;
18 Intel's MCS 51 (i.e., 8051 microcontroller); and/or the like. Also, to implement certain
19 features of the PAFT, some feature implementations may rely on embedded
20 components, such as: Application-Specific Integrated Circuit ("ASIC"), Digital Signal
21 Processing ("DSP"), Field Programmable Gate Array ("FPGA"), and/or the like
22 embedded technology. For example, any of the PAFT component collection (distributed
23 or otherwise) and/or features may be implemented via the microprocessor and/or via
24 embedded components; e.g., via ASIC, coprocessor, DSP, FPGA, and/or the like.

1 Alternately, some implementations of the PAFT may be implemented with embedded
2 components that are configured and used to achieve a variety of features or signal
3 processing.

4 **[0078]** Depending on the particular implementation, the embedded components
5 may include software solutions, hardware solutions, and/or some combination of both
6 hardware/software solutions. For example, PAFT features discussed herein may be
7 achieved through implementing FPGAs, which are a semiconductor devices containing
8 programmable logic components called "logic blocks", and programmable
9 interconnects, such as the high performance FPGA Virtex series and/or the low cost
10 Spartan series manufactured by Xilinx. Logic blocks and interconnects can be
11 programmed by the customer or designer, after the FPGA is manufactured, to
12 implement any of the PAFT features. A hierarchy of programmable interconnects allow
13 logic blocks to be interconnected as needed by the PAFT system designer/administrator,
14 somewhat like a one-chip programmable breadboard. An FPGA's logic blocks can be
15 programmed to perform the function of basic logic gates such as AND, and XOR, or
16 more complex combinational functions such as decoders or simple mathematical
17 functions. In most FPGAs, the logic blocks also include memory elements, which may be
18 simple flip-flops or more complete blocks of memory. In some circumstances, the PAFT
19 may be developed on regular FPGAs and then migrated into a fixed version that more
20 resembles ASIC implementations. Alternate or coordinating implementations may
21 migrate PAFT controller features to a final ASIC instead of or in addition to FPGAs.
22 Depending on the implementation all of the aforementioned embedded components and
23 microprocessors may be considered the "CPU" and/or "processor" for the PAFT.

Power Source

1
2 **[0079]** The power source 1286 may be of any standard form for powering small
3 electronic circuit board devices such as the following power cells: alkaline, lithium
4 hydride, lithium ion, lithium polymer, nickel cadmium, solar cells, and/or the like.
5 Other types of AC or DC power sources may be used as well. In the case of solar cells, in
6 one embodiment, the case provides an aperture through which the solar cell may
7 capture photonic energy. The power cell 1286 is connected to at least one of the
8 interconnected subsequent components of the PAFT thereby providing an electric
9 current to all subsequent components. In one example, the power source 1286 is
10 connected to the system bus component 1204. In an alternative embodiment, an outside
11 power source 1286 is provided through a connection across the I/O 1208 interface. For
12 example, a USB and/or IEEE 1394 connection carries both data and power across the
13 connection and is therefore a suitable source of power.

Interface Adapters

14
15 **[0080]** Interface bus(es) 1207 may accept, connect, and/or communicate to a
16 number of interface adapters, conventionally although not necessarily in the form of
17 adapter cards, such as but not limited to: input output interfaces (I/O) 1208, storage
18 interfaces 1209, network interfaces 1210, and/or the like. Optionally, cryptographic
19 processor interfaces 1227 similarly may be connected to the interface bus. The interface
20 bus provides for the communications of interface adapters with one another as well as
21 with other components of the computer systemization. Interface adapters are adapted
22 for a compatible interface bus. Interface adapters conventionally connect to the
23 interface bus via a slot architecture. Conventional slot architectures may be employed,

1 such as, but not limited to: Accelerated Graphics Port (AGP), Card Bus, (Extended)
2 Industry Standard Architecture ((E)ISA), Micro Channel Architecture (MCA), NuBus,
3 Peripheral Component Interconnect (Extended) (PCI(X)), PCI Express, Personal
4 Computer Memory Card International Association (PCMCIA), and/or the like.

5 **[0081]** Storage interfaces 1209 may accept, communicate, and/or connect to a
6 number of storage devices such as, but not limited to: storage devices 1214, removable
7 disc devices, and/or the like. Storage interfaces may employ connection protocols such
8 as, but not limited to: (Ultra) (Serial) Advanced Technology Attachment (Packet
9 Interface) ((Ultra) (Serial) ATA(PI)), (Enhanced) Integrated Drive Electronics ((E)IDE),
10 Institute of Electrical and Electronics Engineers (IEEE) 1394, fiber channel, Small
11 Computer Systems Interface (SCSI), Universal Serial Bus (USB), and/or the like.

12 **[0082]** Network interfaces 1210 may accept, communicate, and/or connect to a
13 communications network 1213. Through a communications network 1213, the PAFT
14 controller is accessible through remote clients 1233b (e.g., computers with web
15 browsers) by users 1233a. Network interfaces may employ connection protocols such as,
16 but not limited to: direct connect, Ethernet (thick, thin, twisted pair 10/100/1000 Base
17 T, and/or the like), Token Ring, wireless connection such as IEEE 802.11a-x, and/or the
18 like. Should processing requirements dictate a greater amount speed and/or capacity,
19 distributed network controllers (e.g., Distributed PAFT), architectures may similarly be
20 employed to pool, load balance, and/or otherwise increase the communicative
21 bandwidth required by the PAFT controller. A communications network may be any one
22 and/or the combination of the following: a direct interconnection; the Internet; a Local
23 Area Network (LAN); a Metropolitan Area Network (MAN); an Operating Missions as

1 Nodes on the Internet (OMNI); a secured custom connection; a Wide Area Network
2 (WAN); a wireless network (e.g., employing protocols such as, but not limited to a
3 Wireless Application Protocol (WAP), I-mode, and/or the like); and/or the like. A
4 network interface may be regarded as a specialized form of an input output interface.
5 Further, multiple network interfaces 1210 may be used to engage with various
6 communications network types 1213. For example, multiple network interfaces may be
7 employed to allow for the communication over broadcast, multicast, and/or unicast
8 networks.

9 **[0083]** Input Output interfaces (I/O) 1208 may accept, communicate, and/or
10 connect to user input devices 1211, peripheral devices 1212, cryptographic processor
11 devices 1228, and/or the like. I/O may employ connection protocols such as, but not
12 limited to: audio: analog, digital, monaural, RCA, stereo, and/or the like; data: Apple
13 Desktop Bus (ADB), IEEE 1394a-b, serial, universal serial bus (USB); infrared; joystick;
14 keyboard; midi; optical; PC AT; PS/2; parallel; radio; video interface: Apple Desktop
15 Connector (ADC), BNC, coaxial, component, composite, digital, Digital Visual Interface
16 (DVI), high-definition multimedia interface (HDMI), RCA, RF antennae, S-Video, VGA,
17 and/or the like; wireless transceivers: 802.11a/b/g/n/x; Bluetooth; cellular (e.g., code
18 division multiple access (CDMA), high speed packet access (HSPA(+)), high-speed
19 downlink packet access (HSDPA), global system for mobile communications (GSM),
20 long term evolution (LTE), WiMax, etc.); and/or the like. One typical output device may
21 include a video display, which typically comprises a Cathode Ray Tube (CRT) or Liquid
22 Crystal Display (LCD) based monitor with an interface (e.g., DVI circuitry and cable)
23 that accepts signals from a video interface, may be used. The video interface composites
24 information generated by a computer systemization and generates video signals based

1 on the composited information in a video memory frame. Another output device is a
2 television set, which accepts signals from a video interface. Typically, the video interface
3 provides the composited video information through a video connection interface that
4 accepts a video display interface (e.g., an RCA composite video connector accepting an
5 RCA composite video cable; a DVI connector accepting a DVI display cable, etc.).

6 **[0084]** User input devices 1211 often are a type of peripheral device 512 (see
7 below) and may include: card readers, dongles, finger print readers, gloves, graphics
8 tablets, joysticks, keyboards, microphones, mouse (mice), remote controls, retina
9 readers, touch screens (e.g., capacitive, resistive, etc.), trackballs, trackpads, sensors
10 (e.g., accelerometers, ambient light, GPS, gyroscopes, proximity, etc.), styluses, and/or
11 the like.

12 **[0085]** Peripheral devices 1212 may be connected and/or communicate to I/O
13 and/or other facilities of the like such as network interfaces, storage interfaces, directly
14 to the interface bus, system bus, the CPU, and/or the like. Peripheral devices may be
15 external, internal and/or part of the PAFT controller. Peripheral devices may include:
16 antenna, audio devices (e.g., line-in, line-out, microphone input, speakers, etc.),
17 cameras (e.g., still, video, webcam, etc.), dongles (e.g., for copy protection, ensuring
18 secure transactions with a digital signature, and/or the like), external processors (for
19 added capabilities; e.g., crypto devices 528), force-feedback devices (e.g., vibrating
20 motors), network interfaces, printers, scanners, storage devices, transceivers (e.g.,
21 cellular, GPS, etc.), video devices (e.g., goggles, monitors, etc.), video sources, visors,
22 and/or the like. Peripheral devices often include types of input devices (e.g., cameras).

1 **[0086]** It should be noted that although user input devices and peripheral devices
2 may be employed, the PAFT controller may be embodied as an embedded, dedicated,
3 and/or monitor-less (i.e., headless) device, wherein access would be provided over a
4 network interface connection.

5 **[0087]** Cryptographic units such as, but not limited to, microcontrollers,
6 processors 1226, interfaces 1227, and/or devices 1228 may be attached, and/or
7 communicate with the PAFT controller. A MC68HC16 microcontroller, manufactured by
8 Motorola Inc., may be used for and/or within cryptographic units. The MC68HC16
9 microcontroller utilizes a 16-bit multiply-and-accumulate instruction in the 16 MHz
10 configuration and requires less than one second to perform a 512-bit RSA private key
11 operation. Cryptographic units support the authentication of communications from
12 interacting agents, as well as allowing for anonymous transactions. Cryptographic units
13 may also be configured as part of CPU. Equivalent microcontrollers and/or processors
14 may also be used. Other commercially available specialized cryptographic processors
15 include: the Broadcom's CryptoNetX and other Security Processors; nCipher's nShield,
16 SafeNet's Luna PCI (e.g., 7100) series; Semaphore Communications' 40 MHz
17 Roadrunner 184; Sun's Cryptographic Accelerators (e.g., Accelerator 6000 PCIe Board,
18 Accelerator 500 Daughtercard); Via Nano Processor (e.g., L2100, L2200, U2400) line,
19 which is capable of performing 500+ MB/s of cryptographic instructions; VLSI
20 Technology's 33 MHz 6868; and/or the like.

21

Memory

22 **[0088]** Generally, any mechanization and/or embodiment allowing a processor to
23 affect the storage and/or retrieval of information is regarded as memory 1229. However,

1 memory is a fungible technology and resource, thus, any number of memory
2 embodiments may be employed in lieu of or in concert with one another. It is to be
3 understood that the PAFT controller and/or a computer systemization may employ
4 various forms of memory 1229. For example, a computer systemization may be
5 configured wherein the functionality of on-chip CPU memory (e.g., registers), RAM,
6 ROM, and any other storage devices are provided by a paper punch tape or paper punch
7 card mechanism; of course such an embodiment would result in an extremely slow rate
8 of operation. In a typical configuration, memory 1229 will include ROM 1206, RAM
9 1205, and a storage device 1214. A storage device 1214 may be any conventional
10 computer system storage. Storage devices may include a drum; a (fixed and/or
11 removable) magnetic disk drive; a magneto-optical drive; an optical drive (i.e., Blu-ray,
12 CD ROM/RAM/Recordable (R)/ReWritable (RW), DVD R/RW, HD DVD R/RW etc.);
13 an array of devices (e.g., Redundant Array of Independent Disks (RAID)); solid state
14 memory devices (USB memory, solid state drives (SSD), etc.); other processor-readable
15 storage mediums; and/or other devices of the like. Thus, a computer systemization
16 generally requires and makes use of memory.

17 **Component Collection**

18 **[0089]** The memory 1229 may contain a collection of program and/or database
19 components and/or data such as, but not limited to: operating system component(s)
20 1215 (operating system); information server component(s) 1216 (information server);
21 user interface component(s) 1217 (user interface); Web browser component(s) 1218
22 (Web browser); database(s) 1219; mail server component(s) 1221; mail client
23 component(s) 1222; cryptographic server component(s) 1220 (cryptographic server);

1 the PAFT component(s) 1235; and/or the like (i.e., collectively a component collection).
2 These components may be stored and accessed from the storage devices and/or from
3 storage devices accessible through an interface bus. Although non-conventional
4 program components such as those in the component collection, typically, are stored in
5 a local storage device 1214, they may also be loaded and/or stored in memory such as:
6 peripheral devices, RAM, remote storage facilities through a communications network,
7 ROM, various forms of memory, and/or the like.

8 **Operating System**

9 **[0090]** The operating system component 1215 is an executable program
10 component facilitating the operation of the PAFT controller. Typically, the operating
11 system facilitates access of I/O, network interfaces, peripheral devices, storage devices,
12 and/or the like. The operating system may be a highly fault tolerant, scalable, and
13 secure system such as: Apple Macintosh OS X (Server); AT&T Plan 9; Be OS; Unix and
14 Unix-like system distributions (such as AT&T's UNIX; Berkley Software Distribution
15 (BSD) variations such as FreeBSD, NetBSD, OpenBSD, and/or the like; Linux
16 distributions such as Red Hat, Ubuntu, and/or the like); and/or the like operating
17 systems. However, more limited and/or less secure operating systems also may be
18 employed such as Apple Macintosh OS, IBM OS/2, Microsoft DOS, Microsoft Windows
19 2000/2003/3.1/95/98/CE/Millennium/NT/Vista/XP (Server), Palm OS, and/or the like.
20 An operating system may communicate to and/or with other components in a
21 component collection, including itself, and/or the like. Most frequently, the operating
22 system communicates with other program components, user interfaces, and/or the like.
23 For example, the operating system may contain, communicate, generate, obtain, and/or

1 provide program component, system, user, and/or data communications, requests,
2 and/or responses. The operating system, once executed by the CPU, may enable the
3 interaction with communications networks, data, I/O, peripheral devices, program
4 components, memory, user input devices, and/or the like. The operating system may
5 provide communications protocols that allow the PAFT controller to communicate with
6 other entities through a communications network 1213. Various communication
7 protocols may be used by the PAFT controller as a subcarrier transport mechanism for
8 interaction, such as, but not limited to: multicast, TCP/IP, UDP, unicast, and/or the
9 like.

10 **Information Server**

11 **[0091]** An information server component 1216 is a stored program component
12 that is executed by a CPU. The information server may be a conventional Internet
13 information server such as, but not limited to Apache Software Foundation's Apache,
14 Microsoft's Internet Information Server, and/or the like. The information server may
15 allow for the execution of program components through facilities such as Active Server
16 Page (ASP), ActiveX, (ANSI) (Objective-) C (++), C# and/or .NET, Common Gateway
17 Interface (CGI) scripts, dynamic (D) hypertext markup language (HTML), FLASH, Java,
18 JavaScript, Practical Extraction Report Language (PERL), Hypertext Pre-Processor
19 (PHP), pipes, Python, wireless application protocol (WAP), WebObjects, and/or the like.
20 The information server may support secure communications protocols such as, but not
21 limited to, File Transfer Protocol (FTP); HyperText Transfer Protocol (HTTP); Secure
22 Hypertext Transfer Protocol (HTTPS), Secure Socket Layer (SSL), messaging protocols
23 (e.g., America Online (AOL) Instant Messenger (AIM), Application Exchange (APEX),

1 ICQ, Internet Relay Chat (IRC), Microsoft Network (MSN) Messenger Service, Presence
2 and Instant Messaging Protocol (PRIM), Internet Engineering Task Force's (IETF's)
3 Session Initiation Protocol (SIP), SIP for Instant Messaging and Presence Leveraging
4 Extensions (SIMPLE), open XML-based Extensible Messaging and Presence Protocol
5 (XMPP) (i.e., Jabber or Open Mobile Alliance's (OMA's) Instant Messaging and
6 Presence Service (IMPS)), Yahoo! Instant Messenger Service, and/or the like. The
7 information server provides results in the form of Web pages to Web browsers, and
8 allows for the manipulated generation of the Web pages through interaction with other
9 program components. After a Domain Name System (DNS) resolution portion of an
10 HTTP request is resolved to a particular information server, the information server
11 resolves requests for information at specified locations on the PAFT controller based on
12 the remainder of the HTTP request. For example, a request such as
13 `http://123.124.125.126/myInformation.html` might have the IP portion of the request
14 "123.124.125.126" resolved by a DNS server to an information server at that IP address;
15 that information server might in turn further parse the http request for the
16 `"/myInformation.html"` portion of the request and resolve it to a location in memory
17 containing the information "myInformation.html." Additionally, other information
18 serving protocols may be employed across various ports, e.g., FTP communications
19 across port 21, and/or the like. An information server may communicate to and/or with
20 other components in a component collection, including itself, and/or facilities of the
21 like. Most frequently, the information server communicates with the PAFT database
22 1219, operating systems, other program components, user interfaces, Web browsers,
23 and/or the like.

1 **[0092]** Access to the PAFT database may be achieved through a number of
2 database bridge mechanisms such as through scripting languages as enumerated below
3 (e.g., CGI) and through inter-application communication channels as enumerated below
4 (e.g., CORBA, WebObjects, etc.). Any data requests through a Web browser are parsed
5 through the bridge mechanism into appropriate grammars as required by the PAFT. In
6 one embodiment, the information server would provide a Web form accessible by a Web
7 browser. Entries made into supplied fields in the Web form are tagged as having been
8 entered into the particular fields, and parsed as such. The entered terms are then passed
9 along with the field tags, which act to instruct the parser to generate queries directed to
10 appropriate tables and/or fields. In one embodiment, the parser may generate queries in
11 standard SQL by instantiating a search string with the proper join/select commands
12 based on the tagged text entries, wherein the resulting command is provided over the
13 bridge mechanism to the PAFT as a query. Upon generating query results from the
14 query, the results are passed over the bridge mechanism, and may be parsed for
15 formatting and generation of a new results Web page by the bridge mechanism. Such a
16 new results Web page is then provided to the information server, which may supply it to
17 the requesting Web browser.

18 **[0093]** Also, an information server may contain, communicate, generate, obtain,
19 and/or provide program component, system, user, and/or data communications,
20 requests, and/or responses.

21 User Interface

22 **[0094]** Computer interfaces in some respects are similar to automobile operation
23 interfaces. Automobile operation interface elements such as steering wheels, gearshifts,

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

16 **[0095]** A user interface component 1217 is a stored program component that is
17 executed by a CPU. The user interface may be a conventional graphic user interface as
18 provided by, with, and/or atop operating systems and/or operating environments such
19 as already discussed. The user interface may allow for the display, execution,
20 interaction, manipulation, and/or operation of program components and/or system
21 facilities through textual and/or graphical facilities. The user interface provides a facility
22 through which users may affect, interact, and/or operate a computer system. A user
23 interface may communicate to and/or with other components in a component
24 collection, including itself, and/or facilities of the like. Most frequently, the user

1 interface communicates with operating systems, other program components, and/or the
2 like. The user interface may contain, communicate, generate, obtain, and/or provide
3 program component, system, user, and/or data communications, requests, and/or
4 responses.

5 **Web Browser**

6 **[0096]** A Web browser component 1218 is a stored program component that is
7 executed by a CPU. The Web browser may be a conventional hypertext viewing
8 application such as Microsoft Internet Explorer or Netscape Navigator. Secure Web
9 browsing may be supplied with 128bit (or greater) encryption by way of HTTPS, SSL,
10 and/or the like. Web browsers allowing for the execution of program components
11 through facilities such as ActiveX, AJAX, (D)HTML, FLASH, Java, JavaScript, web
12 browser plug-in APIs (e.g., FireFox, Safari Plug-in, and/or the like APIs), and/or the
13 like. Web browsers and like information access tools may be integrated into PDAs,
14 cellular telephones, and/or other mobile devices. A Web browser may communicate to
15 and/or with other components in a component collection, including itself, and/or
16 facilities of the like. Most frequently, the Web browser communicates with information
17 servers, operating systems, integrated program components (e.g., plug-ins), and/or the
18 like; e.g., it may contain, communicate, generate, obtain, and/or provide program
19 component, system, user, and/or data communications, requests, and/or responses. Of
20 course, in place of a Web browser and information server, a combined application may
21 be developed to perform similar functions of both. The combined application would
22 similarly affect the obtaining and the provision of information to users, user agents,

1 and/or the like from the PAFT enabled nodes. The combined application may be
2 nugatory on systems employing standard Web browsers.

3 **Mail Server**

4 **[0097]** A mail server component 1221 is a stored program component that is
5 executed by a CPU 1203. The mail server may be a conventional Internet mail server
6 such as, but not limited to sendmail, Microsoft Exchange, and/or the like. The mail
7 server may allow for the execution of program components through facilities such as
8 ASP, ActiveX, (ANSI) (Objective-) C (++), C# and/or .NET, CGI scripts, Java,
9 JavaScript, PERL, PHP, pipes, Python, WebObjects, and/or the like. The mail server
10 may support communications protocols such as, but not limited to: Internet message
11 access protocol (IMAP), Messaging Application Programming Interface
12 (MAPI)/Microsoft Exchange, post office protocol (POP3), simple mail transfer protocol
13 (SMTP), and/or the like. The mail server can route, forward, and process incoming and
14 outgoing mail messages that have been sent, relayed and/or otherwise traversing
15 through and/or to the PAFT.

16 **[0098]** Access to the PAFT mail may be achieved through a number of APIs
17 offered by the individual Web server components and/or the operating system.

18 **[0099]** Also, a mail server may contain, communicate, generate, obtain, and/or
19 provide program component, system, user, and/or data communications, requests,
20 information, and/or responses.

Mail Client

1
2 **[00100]** A mail client component 1222 is a stored program component that is
3 executed by a CPU 1203. The mail client may be a conventional mail viewing application
4 such as Apple Mail, Microsoft Entourage, Microsoft Outlook, Microsoft Outlook
5 Express, Mozilla, Thunderbird, and/or the like. Mail clients may support a number of
6 transfer protocols, such as: IMAP, Microsoft Exchange, POP3, SMTP, and/or the like. A
7 mail client may communicate to and/or with other components in a component
8 collection, including itself, and/or facilities of the like. Most frequently, the mail client
9 communicates with mail servers, operating systems, other mail clients, and/or the like;
10 e.g., it may contain, communicate, generate, obtain, and/or provide program
11 component, system, user, and/or data communications, requests, information, and/or
12 responses. Generally, the mail client provides a facility to compose and transmit
13 electronic mail messages.

Cryptographic Server

14
15 **[00101]** A cryptographic server component 1220 is a stored program component
16 that is executed by a CPU 1203, cryptographic processor 1226, cryptographic processor
17 interface 1227, cryptographic processor device 1228, and/or the like. Cryptographic
18 processor interfaces will allow for expedition of encryption and/or decryption requests
19 by the cryptographic component; however, the cryptographic component, alternatively,
20 may run on a conventional CPU. The cryptographic component allows for the
21 encryption and/or decryption of provided data. The cryptographic component allows for
22 both symmetric and asymmetric (e.g., Pretty Good Protection (PGP)) encryption and/or
23 decryption. The cryptographic component may employ cryptographic techniques such

1 as, but not limited to: digital certificates (e.g., X.509 authentication framework), digital
2 signatures, dual signatures, enveloping, password access protection, public key
3 management, and/or the like. The cryptographic component will facilitate numerous
4 (encryption and/or decryption) security protocols such as, but not limited to: checksum,
5 Data Encryption Standard (DES), Elliptical Curve Encryption (ECC), International Data
6 Encryption Algorithm (IDEA), Message Digest 5 (MD5, which is a one way hash
7 function), passwords, Rivest Cipher (RC5), Rijndael, RSA (which is an Internet
8 encryption and authentication system that uses an algorithm developed in 1977 by Ron
9 Rivest, Adi Shamir, and Leonard Adleman), Secure Hash Algorithm (SHA), Secure
10 Socket Layer (SSL), Secure Hypertext Transfer Protocol (HTTPS), and/or the like.
11 Employing such encryption security protocols, the PAFT may encrypt all incoming
12 and/or outgoing communications and may serve as node within a virtual private
13 network (VPN) with a wider communications network. The cryptographic component
14 facilitates the process of "security authorization" whereby access to a resource is
15 inhibited by a security protocol wherein the cryptographic component effects authorized
16 access to the secured resource. In addition, the cryptographic component may provide
17 unique identifiers of content, e.g., employing and MD5 hash to obtain a unique
18 signature for an digital audio file. A cryptographic component may communicate to
19 and/or with other components in a component collection, including itself, and/or
20 facilities of the like. The cryptographic component supports encryption schemes
21 allowing for the secure transmission of information across a communications network
22 to enable the PAFT component to engage in secure transactions if so desired. The
23 cryptographic component facilitates the secure accessing of resources on the PAFT and
24 facilitates the access of secured resources on remote systems; i.e., it may act as a client

1 and/or server of secured resources. Most frequently, the cryptographic component
2 communicates with information servers, operating systems, other program components,
3 and/or the like. The cryptographic component may contain, communicate, generate,
4 obtain, and/or provide program component, system, user, and/or data communications,
5 requests, and/or responses.

The PAFT Database

7 **[00102]** The PAFT database component 1219 may be embodied in a database and
8 its stored data. The database is a stored program component, which is executed by the
9 CPU; the stored program component portion configuring the CPU to process the stored
10 data. The database may be a conventional, fault tolerant, relational, scalable, secure
11 database such as Oracle or Sybase. Relational databases are an extension of a flat file.
12 Relational databases consist of a series of related tables. The tables are interconnected
13 via a key field. Use of the key field allows the combination of the tables by indexing
14 against the key field; i.e., the key fields act as dimensional pivot points for combining
15 information from various tables. Relationships generally identify links maintained
16 between tables by matching primary keys. Primary keys represent fields that uniquely
17 identify the rows of a table in a relational database. More precisely, they uniquely
18 identify rows of a table on the “one” side of a one-to-many relationship.

[00103] Alternatively, the PAFT database may be implemented using various
standard data-structures, such as an array, hash, (linked) list, struct, structured text file
(e.g., XML), table, and/or the like. Such data-structures may be stored in memory
and/or in (structured) files. In another alternative, an object-oriented database may be
used, such as Frontier, ObjectStore, Poet, Zope, and/or the like. Object databases can

1 include a number of object collections that are grouped and/or linked together by
2 common attributes; they may be related to other object collections by some common
3 attributes. Object-oriented databases perform similarly to relational databases with the
4 exception that objects are not just pieces of data but may have other types of
5 functionality encapsulated within a given object. If the PAFT database is implemented
6 as a data-structure, the use of the PAFT database 1219 may be integrated into another
7 component such as the PAFT component 1235. Also, the database may be implemented
8 as a mix of data structures, objects, and relational structures. Databases may be
9 consolidated and/or distributed in countless variations through standard data
10 processing techniques. Portions of databases, e.g., tables, may be exported and/or
11 imported and thus decentralized and/or integrated.

12 **[00104]** In one embodiment, the database component 1219 includes several tables
13 1219a-k. A User table 1219a may include fields such as, but not limited to: user_id,
14 applicant_id, firstname, lastname, address_line1, address_line2, dob, ssn,
15 credit_check_flag, zipcode, city, state, account_params_list, account_mode,
16 account_type, account_expiry, preferred_bank_name, preferred_branch_name,
17 credit_report, and/or the like. The User table may support and/or track multiple entity
18 accounts on a PAFT. A Clients table 1219b may include fields such as, but not limited to:
19 client_ID, client_type, client_MAC, client_IP, presentation_format, pixel_count,
20 resolution, screen_size, audio_fidelity, hardware_settings_list, software_
21 compatibilities_list, installed_apps_list, and/or the like. An Invitations table 1219c may
22 include fields such as, but not limited to: request_id, timestamp, transferor_id,
23 client_IP, transferee_details_list, transferee_type, first_name, last_name,
24 contact_type, contact_info, alt_contact_info, alt_contact_type, alt_contact_info,

1 client_type, num_invite_attempts, account_params_list, same_bank_flag,
2 same_branch_flag, persistent_link_flag, transfer_param_list, on_schedule_flag,
3 one_time_flag, amount, instant_transfer_flag, custom_invitation, transferee_details_
4 list, transferee_client_IP, and/or the like. A Forms table 1219d may include fields such
5 as, but not limited to: invitation_template_list, invitation_type, invitation_data, and/or
6 the like. A Clearance table 1219e may include fields such as, but not limited to:
7 applicant_firstname, applicant_lastname, applicant_address_line1, applicant_
8 address_line2, consumer_bureau_data_list, consumer_bureau_data, applicant_clear_
9 flag, credit_limit, credit_score, account_balances, delinquency_flag, quality_flags,
10 and/or the like. A Links table 1219f may include fields such as, but not limited to:
11 timestamp, link_ID, bidirectional_flag, transferor_ID, transferor_account_num,
12 transferor_bank, transferor_branch, transferor_ABA, transferee_ID,
13 transferee_account_num, transferee_bank, transferee_branch, transferee_ABA,
14 and/or the like. A Schedules table 1219g may include fields such as, but not limited to:
15 timestamp, transferor_ID, transferee_ID, link_ID, num_transfers transfer_date,
16 transfer_frequency, transfer_interval, notify_transfers_settings, and/or the like. A
17 Bank Accounts table 1219h may include fields such as, but not limited to: account_id,
18 account_firstname, account_lastname, accountholder_type, account_type,
19 contact_type, contact_info, opening_balance, current_balance, and/or the like. A
20 Transfers table 1219i may include fields such as, but not limited to: transfer_ID,
21 transferor_ID, transferee_ID, link_ID, transfer_amount, timestamp,
22 notify_transfers_flags, and/or the like. An App Modules table 1219j may include fields
23 such as, but not limited to: app_ID, app_name, app_type, OS_compatibilities_list,
24 version, timestamp, developer_ID, and/or the like. A Market Data table 1219k may

1 include fields such as, but not limited to: market_data_feed_ID, asset_ID,
2 asset_symbol, asset_name, spot_price, bid_price, ask_price, and/or the like; in one
3 embodiment, the market data table is populated through a market data feed (e.g.,
4 Bloomberg's PhatPipe, Dun & Bradstreet, Reuter's Tib, Triarch, etc.), for example,
5 through Microsoft's Active Template Library and Dealing Object Technology's real-time
6 toolkit Rtt.Multi.

7 **[00105]** In one embodiment, the PAFT database may interact with other database
8 systems. For example, employing a distributed database system, queries and data access
9 by search PAFT component may treat the combination of the PAFT database, an
10 integrated data security layer database as a single database entity.

11 **[00106]** In one embodiment, user programs may contain various user interface
12 primitives, which may serve to update the PAFT. Also, various accounts may require
13 custom database tables depending upon the environments and the types of clients the
14 PAFT may need to serve. It should be noted that any unique fields may be designated as
15 a key field throughout. In an alternative embodiment, these tables have been
16 decentralized into their own databases and their respective database controllers (i.e.,
17 individual database controllers for each of the above tables). Employing standard data
18 processing techniques, one may further distribute the databases over several computer
19 systemizations and/or storage devices. Similarly, configurations of the decentralized
20 database controllers may be varied by consolidating and/or distributing the various
21 database components 1219a-k. The PAFT may be configured to keep track of various
22 settings, inputs, and parameters via database controllers.

1 **[00107]** The PAFT database may communicate to and/or with other components in
2 a component collection, including itself, and/or facilities of the like. Most frequently, the
3 PAFT database communicates with the PAFT component, other program components,
4 and/or the like. The database may contain, retain, and provide information regarding
5 other nodes and data.

6 **The PAFTs**

7 **[00108]** The PAFT component 1235 is a stored program component that is
8 executed by a CPU. In one embodiment, the PAFT component incorporates any and/or
9 all combinations of the aspects of the PAFT discussed in the previous figures. As such,
10 the PAFT affects accessing, obtaining and the provision of information, services,
11 transactions, and/or the like across various communications networks.

12 **[00109]** The PAFT component may transform prepaid account invitation requests
13 via PAFT components into scheduled prepaid account transactions, and/or the like and
14 use of the PAFT. In one embodiment, the PAFT component 1235 takes inputs (e.g.,
15 invitation request input 211, invitation template 216, invitation acceptance input 411,
16 invitation type 416, prepaid account application form 419, application form input 423,
17 applicant screening failure report 428a, applicant screening success report 428b,
18 payment order input 611a, receivables request input 611b, transfer schedule 612c,
19 transfer link data 615, user profile data 617, and/or the like) etc., and transforms the
20 inputs via various components (e.g., PAAI component 1241, PAR component 1242, Tr-
21 FTT component 1243, Te-FTT component 1244, S-FTT component 1245, PA-FTP
22 component 1246, and/or the like), into outputs (e.g., invitation data 218, customized
23 prepaid account invitation 219, invitation confirmation message 222, prepaid account

1 application form 420, application denial notification 429a, prepaid account opening
2 request message 430b, account issue message 436, user profile 438, prepaid account
3 link data 439, prepaid account funds transfer schedule 440, transaction data 622,
4 transactions notifications 623a-b, and/or the like).

5 **[00110]** The PAFT component enabling access of information between nodes may
6 be developed by employing standard development tools and languages such as, but not
7 limited to: Apache components, Assembly, ActiveX, binary executables, (ANSI)
8 (Objective-) C (++), C# and/or .NET, database adapters, CGI scripts, Java, JavaScript,
9 mapping tools, procedural and object oriented development tools, PERL, PHP, Python,
10 shell scripts, SQL commands, web application server extensions, web development
11 environments and libraries (e.g., Microsoft's ActiveX; Adobe AIR, FLEX & FLASH;
12 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 PAFT
15 server employs a cryptographic server to encrypt and decrypt communications. The
16 PAFT component may communicate to and/or with other components in a component
17 collection, including itself, and/or facilities of the like. Most frequently, the PAFT
18 component communicates with the PAFT database, operating systems, other program
19 components, and/or the like. The PAFT may contain, communicate, generate, obtain,
20 and/or provide program component, system, user, and/or data communications,
21 requests, and/or responses.

Distributed PAFTs

1

2 **[00111]** The structure and/or operation of any of the PAFT node controller
3 components may be combined, consolidated, and/or distributed in any number of ways
4 to facilitate development and/or deployment. Similarly, the component collection may
5 be combined in any number of ways to facilitate deployment and/or development. To
6 accomplish this, one may integrate the components into a common code base or in a
7 facility that can dynamically load the components on demand in an integrated fashion.

8 **[00112]** The component collection may be consolidated and/or distributed in
9 countless variations through standard data processing and/or development techniques.
10 Multiple instances of any one of the program components in the program component
11 collection may be instantiated on a single node, and/or across numerous nodes to
12 improve performance through load-balancing and/or data-processing techniques.
13 Furthermore, single instances may also be distributed across multiple controllers
14 and/or storage devices; e.g., databases. All program component instances and
15 controllers working in concert may do so through standard data processing
16 communication techniques.

17 **[00113]** The configuration of the PAFT controller will depend on the context of
18 system deployment. Factors such as, but not limited to, the budget, capacity, location,
19 and/or use of the underlying hardware resources may affect deployment requirements
20 and configuration. Regardless of if the configuration results in more consolidated
21 and/or integrated program components, results in a more distributed series of program
22 components, and/or results in some combination between a consolidated and
23 distributed configuration, data may be communicated, obtained, and/or provided.

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

7 **[00114]** If component collection components are discrete, separate, and/or
8 external to one another, then communicating, obtaining, and/or providing data with
9 and/or to other component components may be accomplished through inter-application
10 data processing communication techniques such as, but not limited to: Application
11 Program Interfaces (API) information passage; (distributed) Component Object Model
12 ((D)COM), (Distributed) Object Linking and Embedding ((D)OLE), and/or the like),
13 Common Object Request Broker Architecture (CORBA), Jini local and remote
14 application program interfaces, JavaScript Object Notation (JSON), Remote Method
15 Invocation (RMI), SOAP, process pipes, shared files, and/or the like. Messages sent
16 between discrete component components for inter-application communication or within
17 memory spaces of a singular component for intra-application communication may be
18 facilitated through the creation and parsing of a grammar. A grammar may be
19 developed by using development tools such as lex, yacc, XML, and/or the like, which
20 allow for grammar generation and parsing capabilities, which in turn may form the basis
21 of communication messages within and between components.

22 **[00115]** For example, a grammar may be arranged to recognize the tokens of an
23 HTTP post command, e.g.:

1 w3c -post http://... Value1

2

3 **[00116]** where Value1 is discerned as being a parameter because “http://” is part of
4 the grammar syntax, and what follows is considered part of the post value. Similarly,
5 with such a grammar, a variable “Value1” may be inserted into an “http://” post
6 command and then sent. The grammar syntax itself may be presented as structured data
7 that is interpreted and/or otherwise used to generate the parsing mechanism (e.g., a
8 syntax description text file as processed by lex, yacc, etc.). Also, once the parsing
9 mechanism is generated and/or instantiated, it itself may process and/or parse
10 structured data such as, but not limited to: character (e.g., tab) delineated text, HTML,
11 structured text streams, XML, and/or the like structured data. In another embodiment,
12 inter-application data processing protocols themselves may have integrated and/or
13 readily available parsers (e.g., JSON, SOAP, and/or like parsers) that may be employed
14 to parse (e.g., communications) data. Further, the parsing grammar may be used
15 beyond message parsing, but may also be used to parse: databases, data collections, data
16 stores, structured data, and/or the like. Again, the desired configuration will depend
17 upon the context, environment, and requirements of system deployment.

18 **[00117]** For example, in some implementations, the PAFT controller may be
19 executing a PHP script implementing a Secure Sockets Layer (“SSL”) socket server via
20 the information server, which listens to incoming communications on a server port to
21 which a client may send data, e.g., data encoded in JSON format. Upon identifying an
22 incoming communication, the PHP script may read the incoming message from the
23 client device, parse the received JSON-encoded text data to extract information from the
24 JSON-encoded text data into PHP script variables, and store the data (e.g., client
25 identifying information, etc.) and/or extracted information in a relational database

1 accessible using the Structured Query Language ("SQL"). An example listing, written
 2 substantially in the form of PHP/SQL commands, to accept JSON-encoded input data
 3 from a client device via a SSL connection, parse the data to extract variables, and store
 4 the data to a database, is provided below:

```

5  <?PHP
6  header('Content-Type: text/plain');
7
8  // set ip address and port to listen to for incoming data
9  $address = '192.168.0.100';
10 $port = 255;
11
12 // create a server-side SSL socket, listen for/accept incoming communication
13 $sock = socket_create(AF_INET, SOCK_STREAM, 0);
14 socket_bind($sock, $address, $port) or die('Could not bind to address');
15 socket_listen($sock);
16 $client = socket_accept($sock);
17
18 // read input data from client device in 1024 byte blocks until end of message
19 do {
20     $input = "";
21     $input = socket_read($client, 1024);
22     $data .= $input;
23 } while($input != "");
24
25 // parse data to extract variables
26 $obj = json_decode($data, true);
27
28 // store input data in a database
29 mysql_connect("201.408.185.132", $DBserver, $password); // access database server
30 mysql_select("CLIENT_DB.SQL"); // select database to append
31 mysql_query("INSERT INTO UserTable (transmission)
32 VALUES ($data)"); // add data to UserTable table in a CLIENT database
33 mysql_close("CLIENT_DB.SQL"); // close connection to database
34 ?>
35

```

36 **[00118]** Also, the following resources may be used to provide example
 37 embodiments regarding SOAP parser implementation:

```

38 http://www.xav.com/perl/site/lib/SOAP/Parser.html
39 http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?topic=/com.ibm
40 .IBMDI.doc/referenceguide295.htm
41

```

42 and other parser implementations:

```

43 http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?topic=/com.ibm
44 .IBMDI.doc/referenceguide259.htm
45

```

46 all of which are hereby expressly incorporated by reference.

1 **[00119]** In order to address various issues and advance the art, the entirety of this
2 application for PREPAID ACCOUNT FUNDS TRANSFER APPARATUSES, METHODS
3 AND SYSTEMS (including the Cover Page, Title, Headings, Field, Background,
4 Summary, Brief Description of the Drawings, Detailed Description, Claims, Abstract,
5 Figures, Appendices and/or otherwise) shows by way of illustration various
6 embodiments in which the claimed inventions may be practiced. The advantages and
7 features of the application are of a representative sample of embodiments only, and are
8 not exhaustive and/or exclusive. They are presented only to assist in understanding and
9 teach the claimed principles. It should be understood that they are not representative of
10 all claimed inventions. As such, certain aspects of the disclosure have not been discussed
11 herein. That alternate embodiments may not have been presented for a specific portion
12 of the invention or that further undescribed alternate embodiments may be available for
13 a portion is not to be considered a disclaimer of those alternate embodiments. It will be
14 appreciated that many of those undescribed embodiments incorporate the same
15 principles of the invention and others are equivalent. Thus, it is to be understood that
16 other embodiments may be utilized and functional, logical, organizational, structural
17 and/or topological modifications may be made without departing from the scope and/or
18 spirit of the disclosure. As such, all examples and/or embodiments are deemed to be
19 non-limiting throughout this disclosure. Also, no inference should be drawn regarding
20 those embodiments discussed herein relative to those not discussed herein other than it
21 is as such for purposes of reducing space and repetition. For instance, it is to be
22 understood that the logical and/or topological structure of any combination of any
23 program components (a component collection), other components and/or any present
24 feature sets as described in the figures and/or throughout are not limited to a fixed

1 operating order and/or arrangement, but rather, any disclosed order is exemplary and
2 all equivalents, regardless of order, are contemplated by the disclosure. Furthermore, it
3 is to be understood that such features are not limited to serial execution, but rather, any
4 number of threads, processes, services, servers, and/or the like that may execute
5 asynchronously, concurrently, in parallel, simultaneously, synchronously, and/or the
6 like are contemplated by the disclosure. As such, some of these features may be
7 mutually contradictory, in that they cannot be simultaneously present in a single
8 embodiment. Similarly, some features are applicable to one aspect of the invention, and
9 inapplicable to others. In addition, the disclosure includes other inventions not
10 presently claimed. Applicant reserves all rights in those presently unclaimed inventions
11 including the right to claim such inventions, file additional applications, continuations,
12 continuations in part, divisions, and/or the like thereof. As such, it should be
13 understood that advantages, embodiments, examples, functional, features, logical,
14 organizational, structural, topological, and/or other aspects of the disclosure are not to
15 be considered limitations on the disclosure as defined by the claims or limitations on
16 equivalents to the claims. It is to be understood that, depending on the particular needs
17 and/or characteristics of a PAFT individual and/or enterprise user, database
18 configuration and/or relational model, data type, data transmission and/or network
19 framework, syntax structure, and/or the like, various embodiments of the PAFT may be
20 implemented that enable a great deal of flexibility and customization. For example,
21 aspects of the PAFT may be adapted for credit cards, certificates of deposit (CDs),
22 money market accounts, stock purchases, trading systems, mortgages, insurance
23 policies, and/or the like. In general, it is contemplated that aspects of the PAFT may be
24 adapted for any mode of transfer of wealth from one entity to another. While various

1 embodiments and discussions of the PAFT have been directed to account-based
2 transactions, however, it is to be understood that the embodiments described herein may
3 be readily configured and/or customized for a wide variety of other applications and/or
4 implementations.

CLAIMS

What is claimed is:

1. A prepaid account funds transfer processor-implemented method, comprising:

obtaining, from a transferor user holding a transferor prepaid account, a prepaid account invitation request specifying a transferee user and a prepaid funds transfer amount;

generating a customized prepaid account invitation for the transferee user based on the prepaid account invitation request;

providing the customized prepaid account invitation for the transferee user;

obtaining an invitation acceptance from the transferee user in response to the provided customized prepaid account invitation;

creating, via a processor, a transferee prepaid account for the transferee user; and

transferring, upon creating the transferee prepaid account, the prepaid funds transfer amount of funds specified in the prepaid account invitation request from the transferor prepaid account to the created transferee prepaid account.

2. The method of claim 1, further comprising:

providing a prepaid account application for the transferee user; and

obtaining a completed prepaid account application;

1 wherein the transferee prepaid account is created based on the completed
2 prepaid account application.

3

4 3. The method of claim 2, wherein creation of the transferee prepaid account is
5 contingent upon the transferee user passing a screening test based on the completed
6 prepaid account application.

7

8 4. The method of claim 1, wherein the transferee prepaid account is created
9 locally.

10

11 5. The method of claim 1, wherein the customized prepaid account invitation is
12 customized based on an attribute of a transferee client device of the transferee user.

13

14 6. The method of claim 1, further comprising:

15 generating a prepaid account funds transfer schedule for transfer of funds
16 between the transferor prepaid account and the transferee prepaid account; and

17 transferring a scheduled transfer amount from the transferor prepaid
18 account to the created transferee prepaid account based on the generated prepaid
19 account funds transfer schedule.

20

21 7. The method of claim 6, wherein at least one scheduled funds transfer included
22 in the prepaid account funds transfer schedule specifies transferring funds from the
23 transferee prepaid account to the transferor prepaid account.

24

1 8. A prepaid account funds transfer system, comprising:

2 a memory; and

3 a processor disposed in communication with the memory and configured to issue
4 processing instructions stored in the memory to:

5 obtain, from a transferor user holding a transferor prepaid account, a
6 prepaid account invitation request specifying a transferee user and a prepaid funds
7 transfer amount;

8 generate a customized prepaid account invitation for the transferee user
9 based on the prepaid account invitation request;

10 provide the customized prepaid account invitation for the transferee user;

11 obtain an invitation acceptance from the transferee user in response to the
12 provided customized prepaid account invitation;

13 create a transferee prepaid account for the transferee user; and

14 transfer, in response to creating the transferee prepaid account, the
15 prepaid funds transfer amount of funds specified in the prepaid account invitation
16 request from the transferor prepaid account to the created transferee prepaid account.

17
18 9. The system of claim 8, the memory further storing processing instructions to:

19 provide a prepaid account application form for the transferee user; and

20 obtain a completed prepaid account application form;

21 wherein the transferee prepaid account is created based on the completed
22 prepaid account application form.

1 10. The system of claim 9, wherein creation of the transferee prepaid account
2 depends on the transferee user passing a screening test based on the completed prepaid
3 account application.

4

5 11. The system of claim 8, wherein the transferee prepaid account is created
6 locally.

7

8 12. The system of claim 8, wherein the customized prepaid account invitation is
9 customized based on an attribute of a transferee client device of the transferee user.

10

11 13. The system of claim 8, the memory further storing processing instructions to:
12 generate a prepaid account funds transfer schedule for transfer of funds
13 between the transferor prepaid account and the transferee prepaid account; and
14 transfer a scheduled transfer amount from the transferor prepaid account
15 to the created transferee prepaid account according to the generated prepaid account
16 funds transfer schedule.

17

18 14. The system of claim 13, wherein at least one scheduled funds transfer
19 included in the prepaid account funds transfer schedule specifies transferring funds
20 from the transferee prepaid account to the transferor prepaid account.

21

22 15. A processor-readable tangible medium storing processor-issuable prepaid
23 account funds transfer instructions to:

1 obtain, from a transferor user holding a transferor prepaid account, a
2 prepaid account invitation request specifying a transferee user and a prepaid funds
3 transfer amount;

4 generate a customized prepaid account invitation for the transferee user
5 based on the prepaid account invitation request;

6 provide the customized prepaid account invitation for the transferee user;

7 obtain an invitation acceptance from the transferee user in response to the
8 provided customized prepaid account invitation;

9 create a transferee prepaid account for the transferee user; and

10 transfer, in response to creating the transferee prepaid account, the
11 prepaid funds transfer amount of funds specified in the prepaid account invitation
12 request from the transferor prepaid account to the created transferee prepaid account.

13

14 16. The medium of claim 15, further storing processing instructions to:

15 provide a prepaid account application for the transferee user; and

16 obtain a completed prepaid account application;

17 wherein the transferee prepaid account is created based on the completed
18 prepaid account application form.

19

20 17. The medium of claim 16, wherein creation of the transferee prepaid account
21 depends on the transferee user passing a screening test based on the completed prepaid
22 account application.

23

1 18. The medium of claim 15, wherein the transferee prepaid account is created
2 locally.

3

4 19. The medium of claim 15, wherein the customized prepaid account invitation is
5 customized based on an attribute of a transferee client device of the transferee user.

6

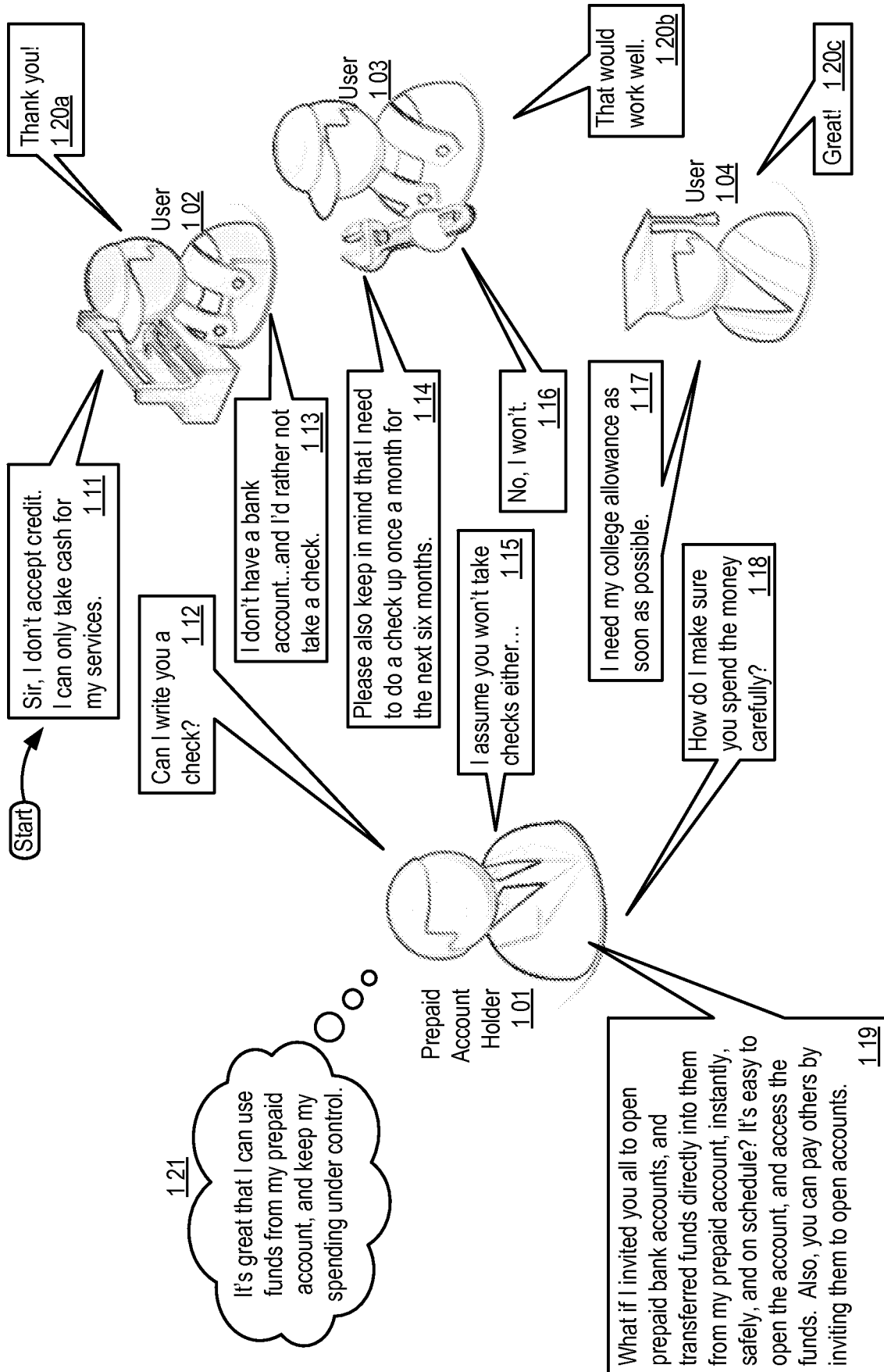
7 20. The medium of claim 15, further storing processing instructions to:
8 generate a prepaid account funds transfer schedule for transfer of funds
9 between the transferor prepaid account and the transferee prepaid account; and
10 transfer a scheduled transfer amount from the transferor prepaid account
11 to the created transferee prepaid account according to the generated prepaid account
12 funds transfer schedule.

13

14 21. The medium of claim 20, wherein at least one scheduled funds transfer
15 included in the prepaid account funds transfer schedule specifies transferring funds
16 from the transferee prepaid account to the transferor prepaid account.

17

1/21



EXAMPLE: PREPAID ACCOUNT FUNDS TRANSFER (PAFT)

FIGURE 1

2/21

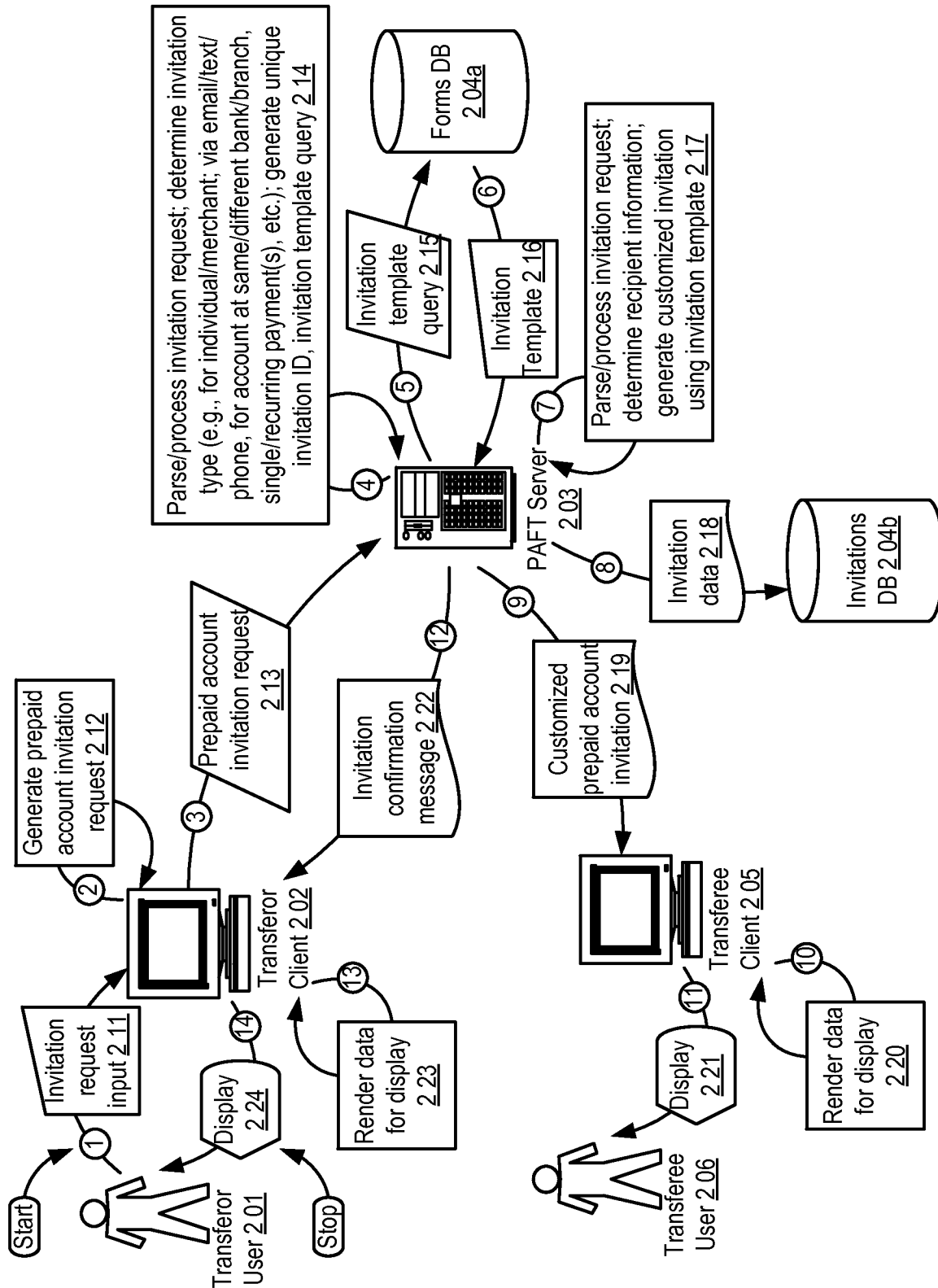


FIGURE 2

Example: Prepaid Account Application Invitation

3/21

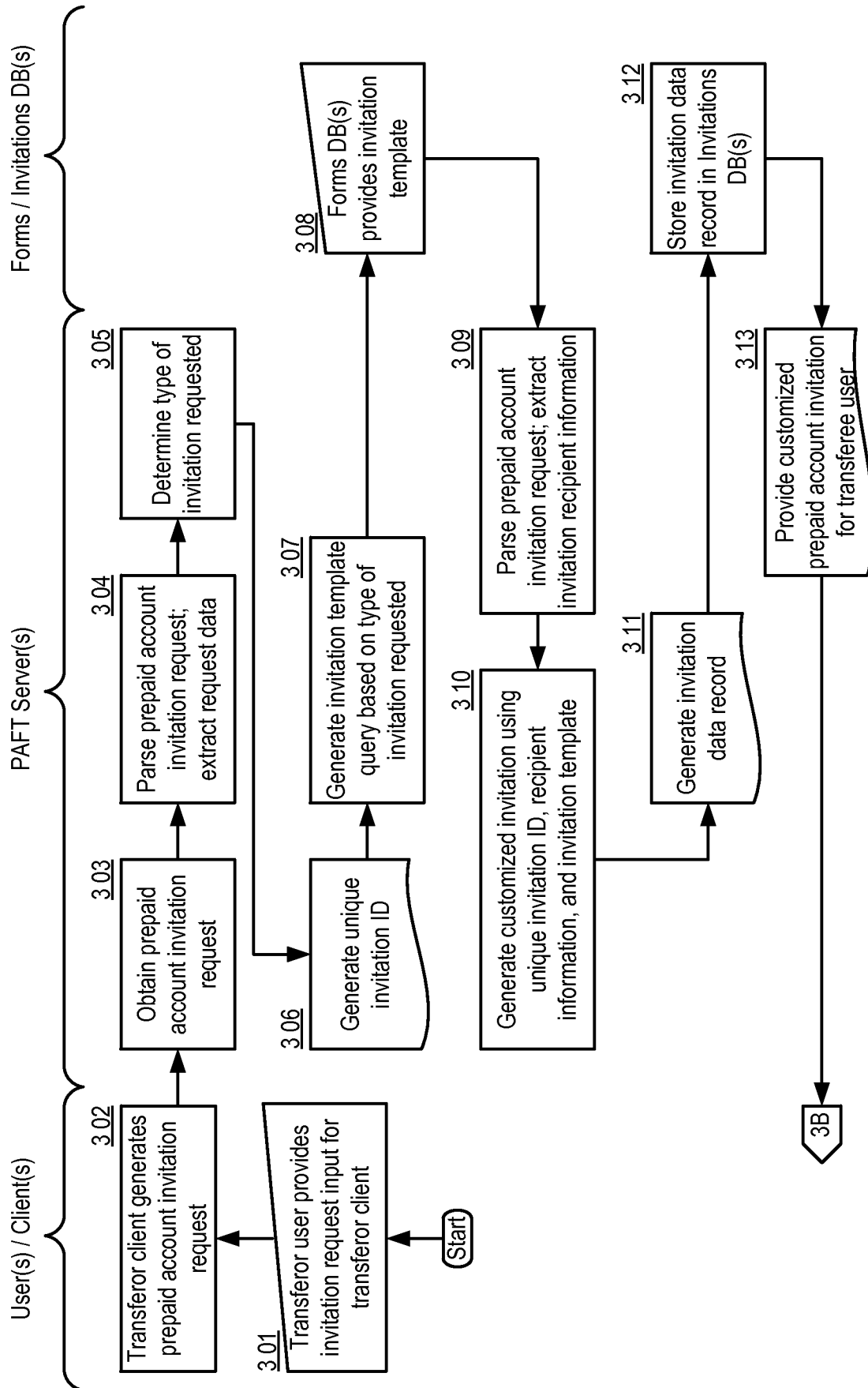


FIGURE 3A

Example: Prepaid Account Application Invitation (PAAI) component 300

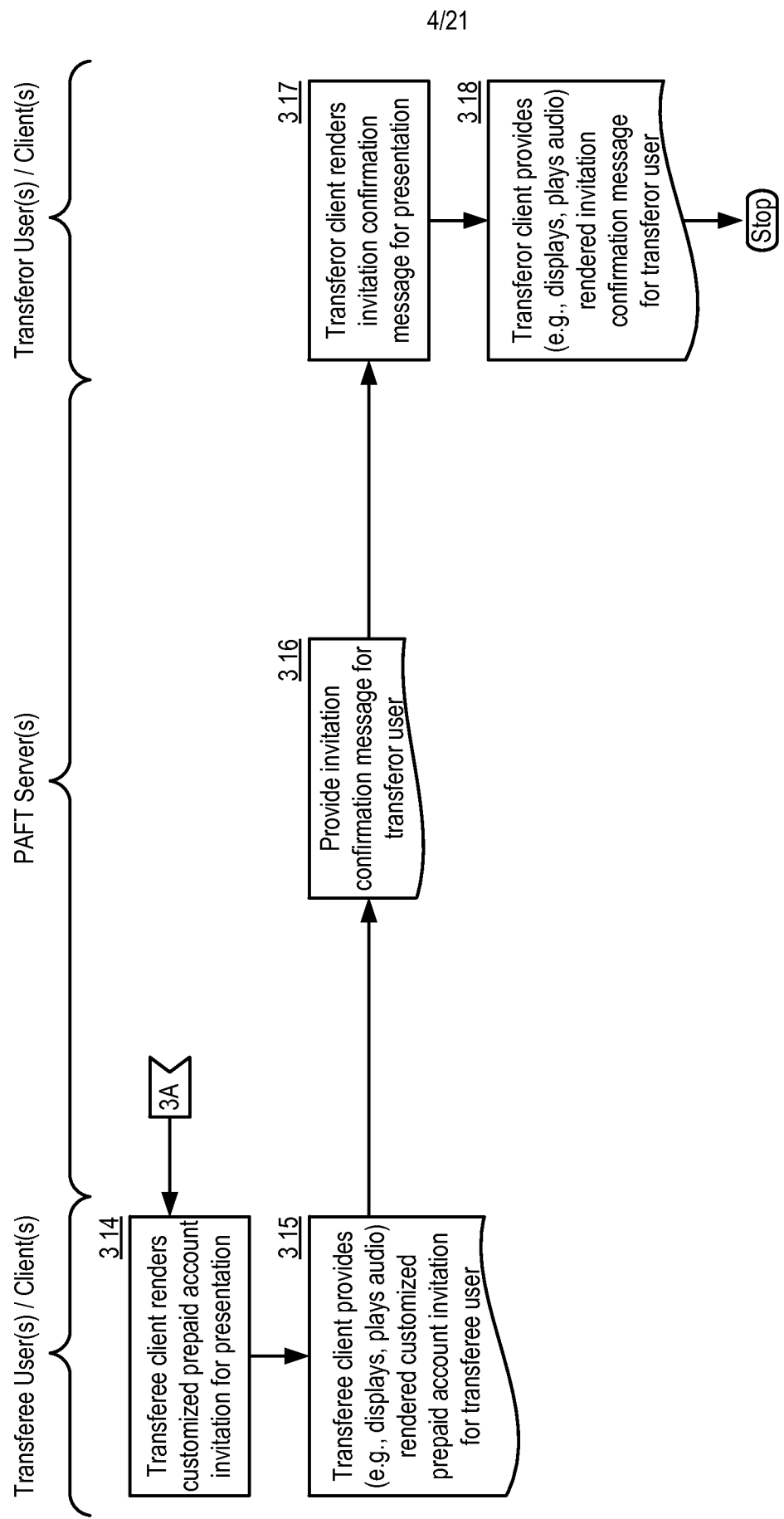


FIGURE 3B

Example: Prepaid Account Application Invitation (PAAI) component 300

5/21

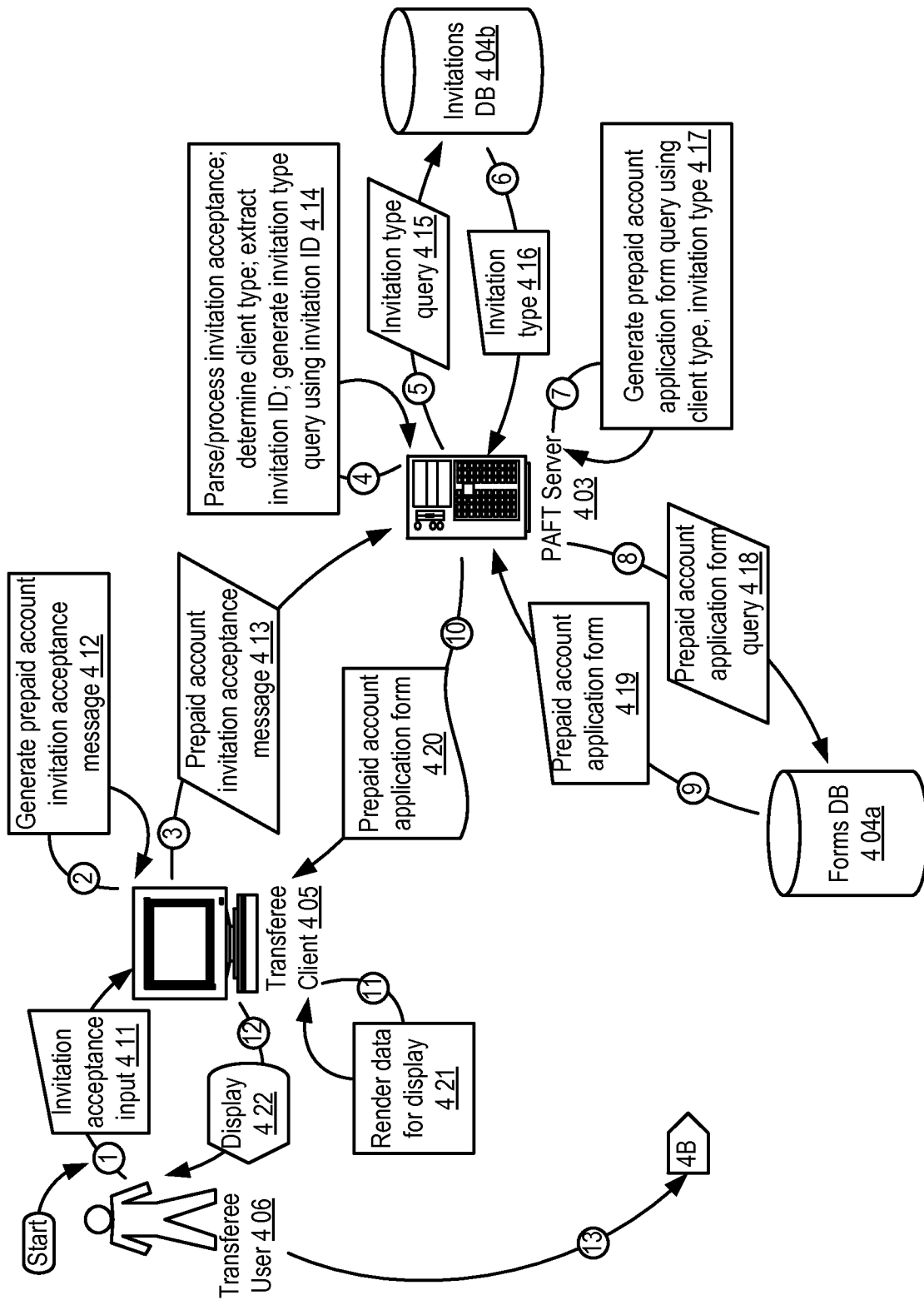


FIGURE 4A

Example: Prepaid Account Registration

6/21

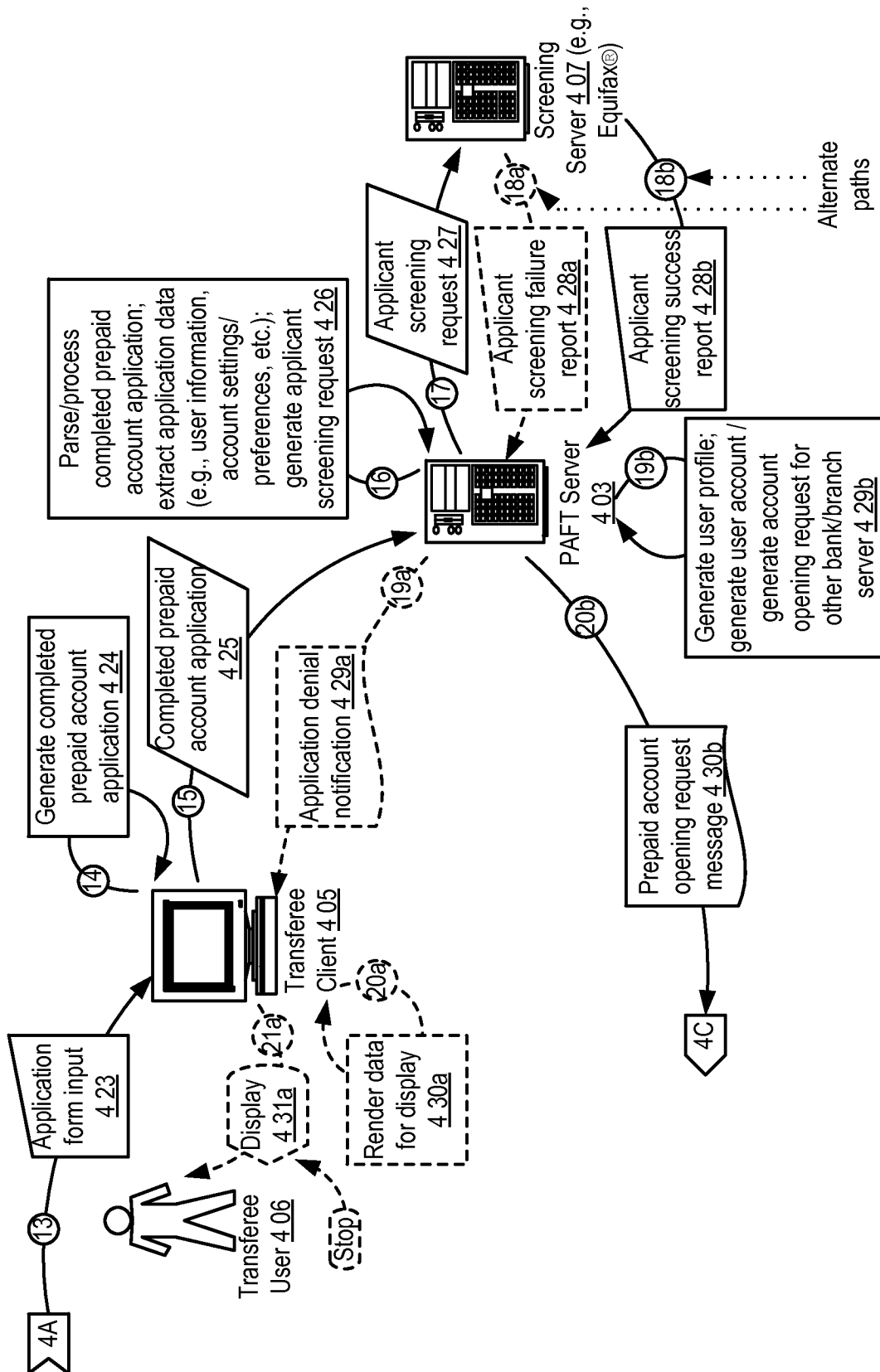


FIGURE 4B

Example: Prepaid Account Registration

7/21

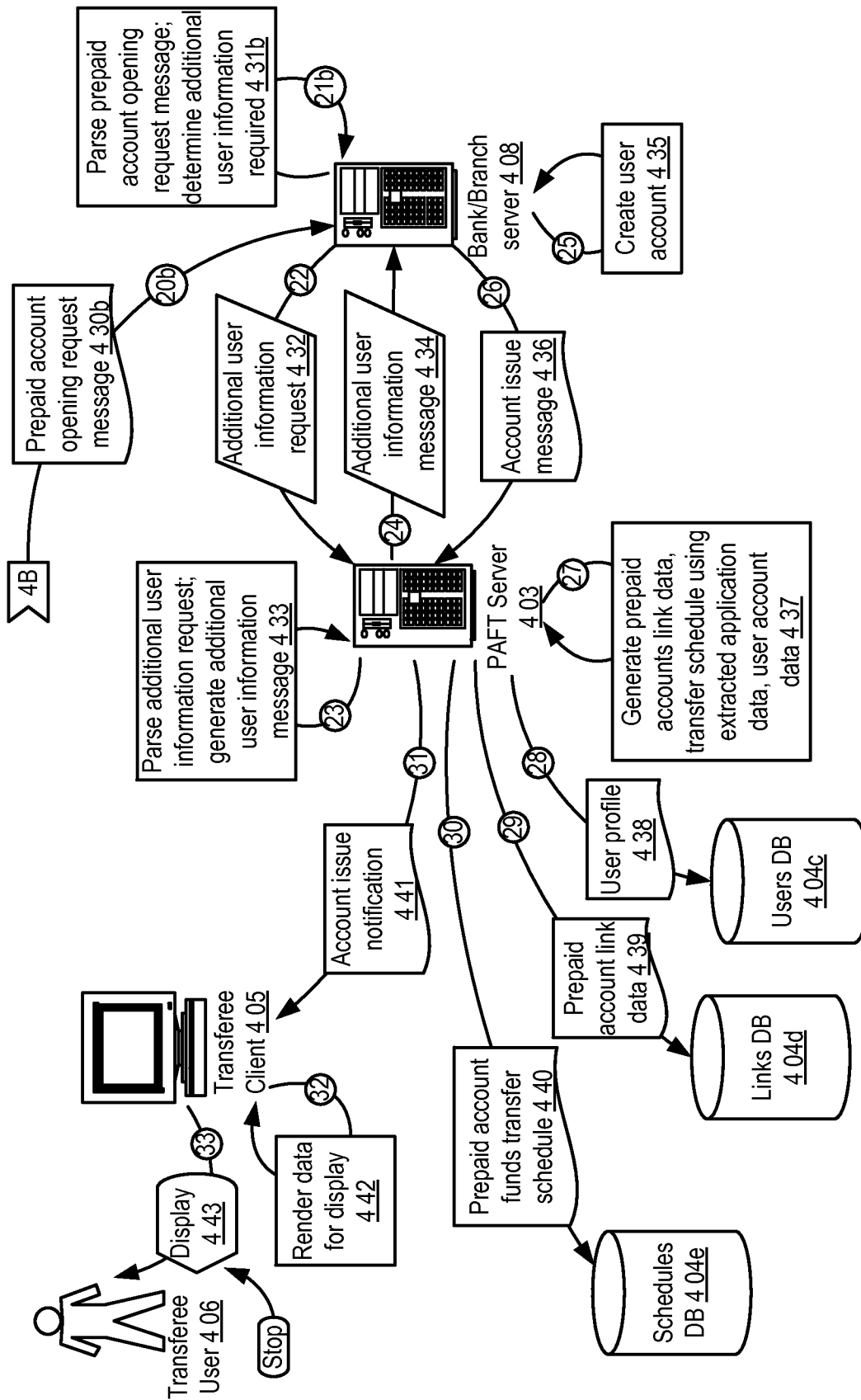


FIGURE 4C

Example: Prepaid Account Registration

8/21

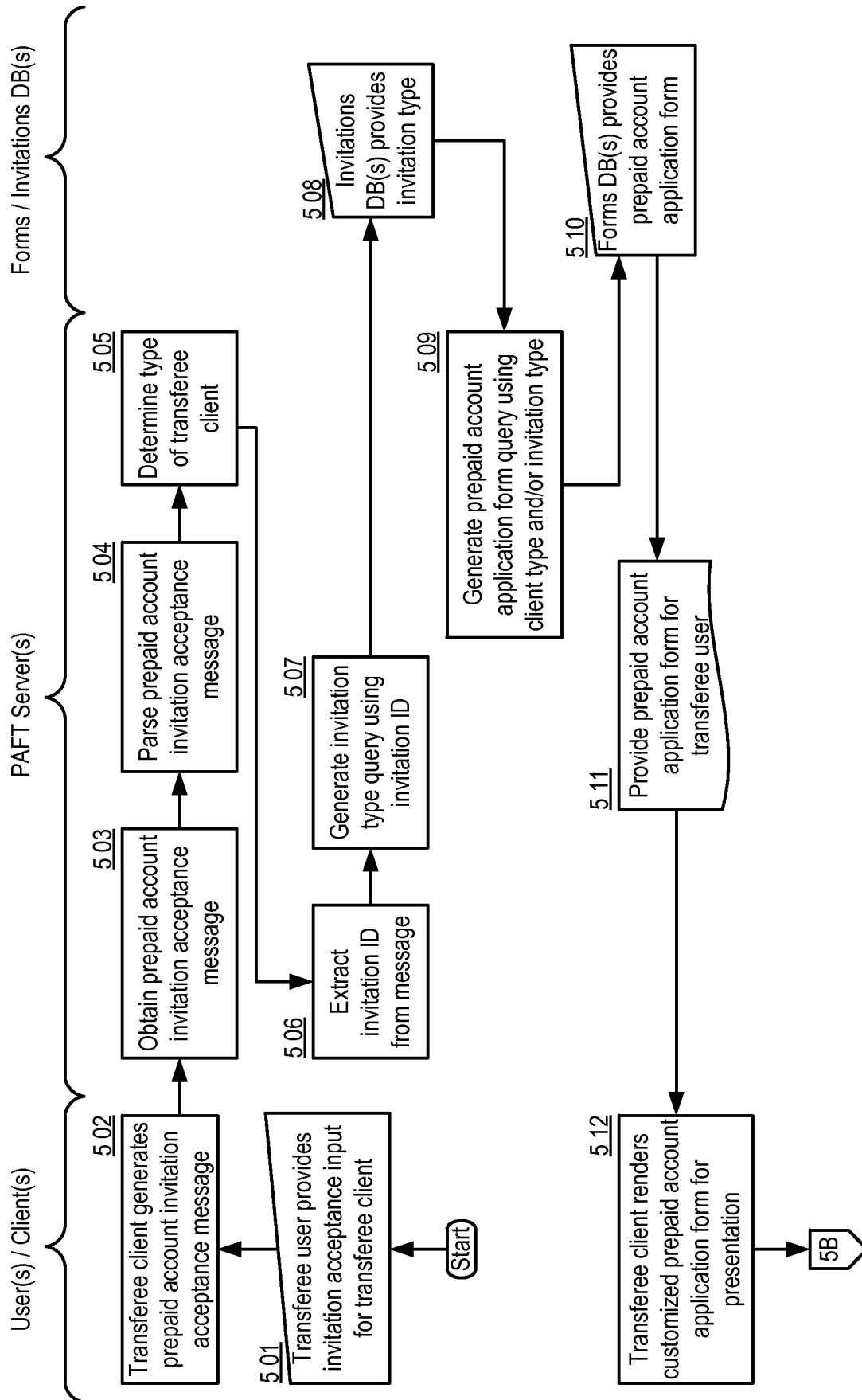


FIGURE 5A

Example: Prepaid Account Registration (PAR) component 500

9/21

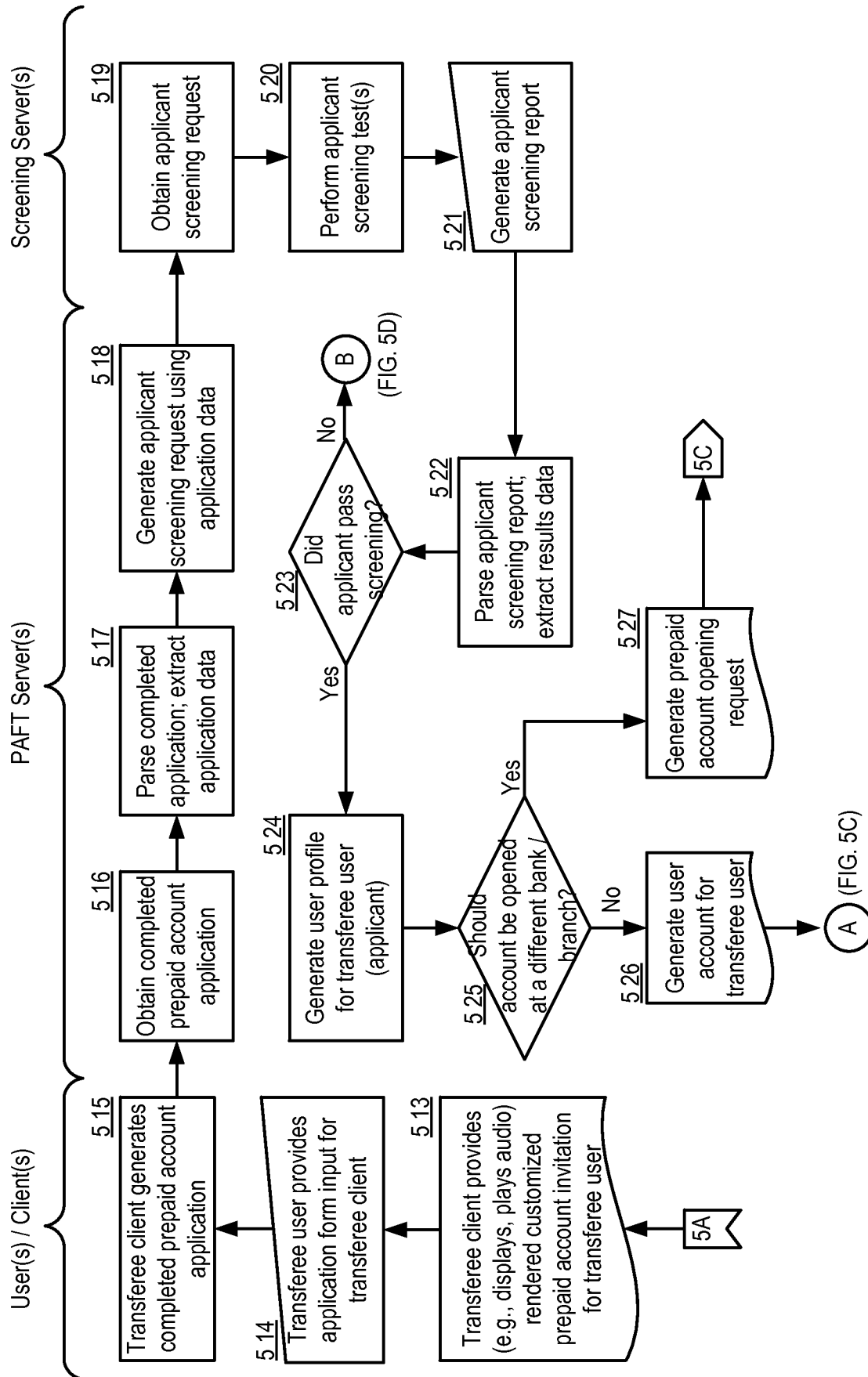


FIGURE 5B

Example: Prepaid Account Registration (PAR) component 500

10/21

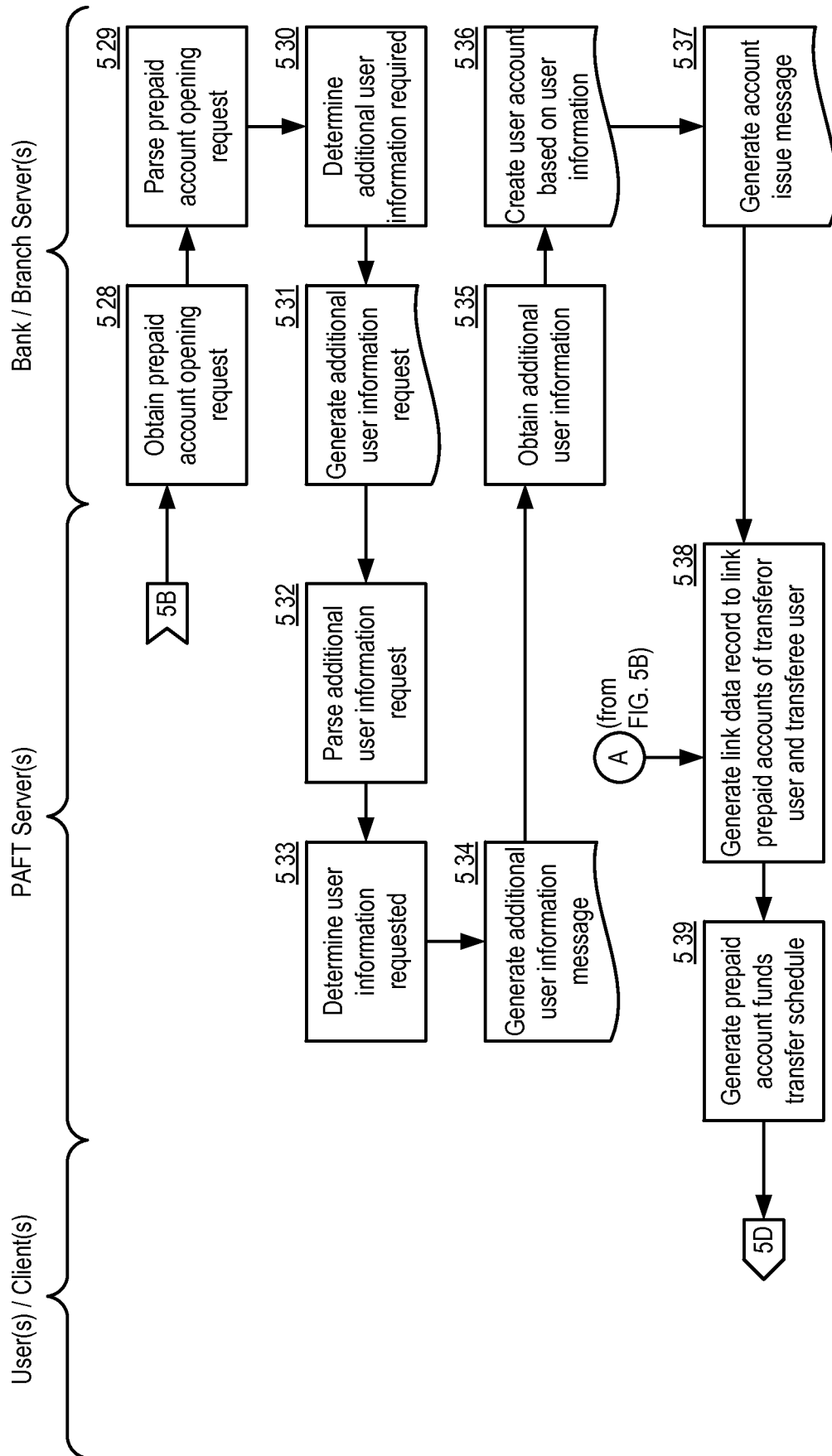


FIGURE 5C

Example: Prepaid Account Registration (PAR) component 500

11/21

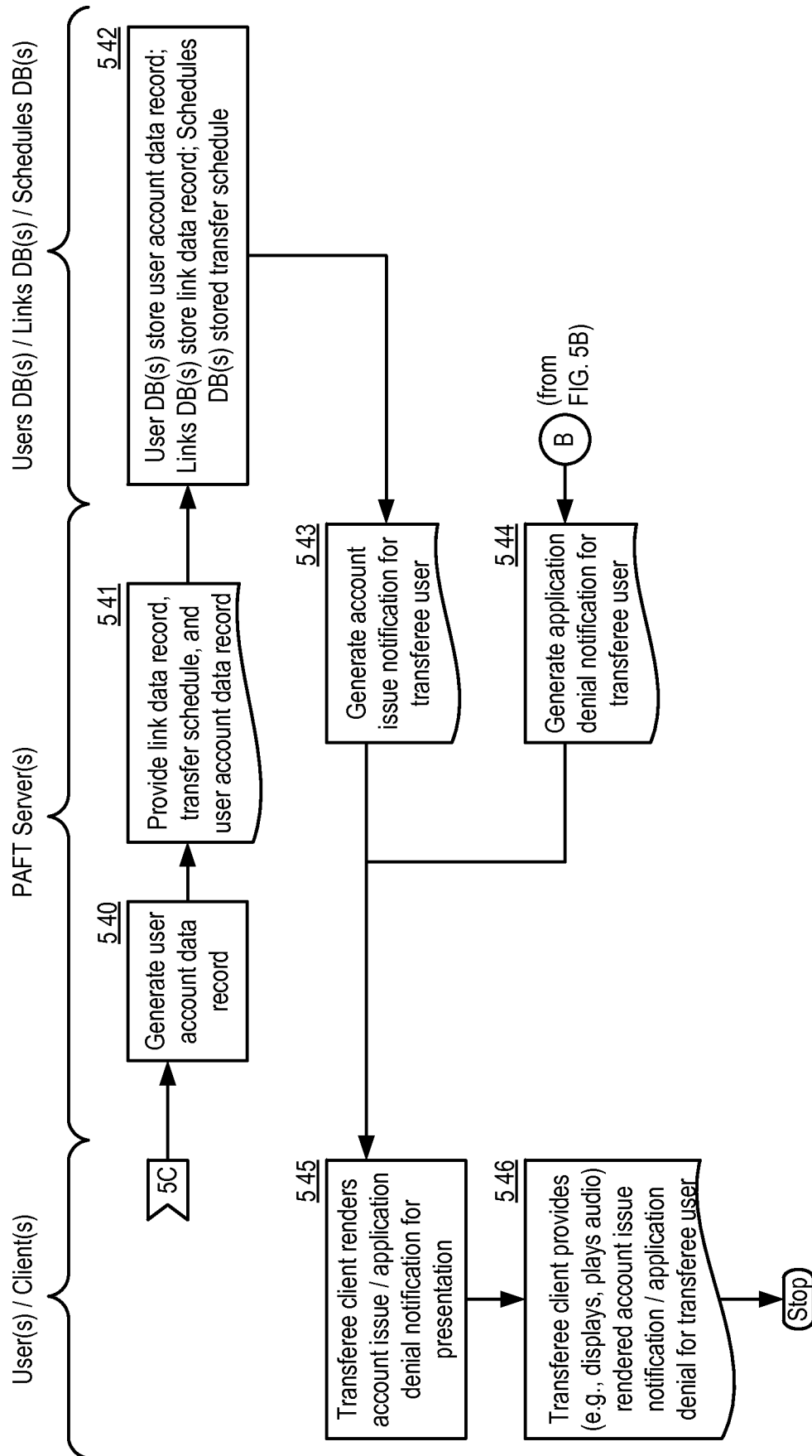


FIGURE 5D

Example: Prepaid Account Registration (PAR) component 500

12/21

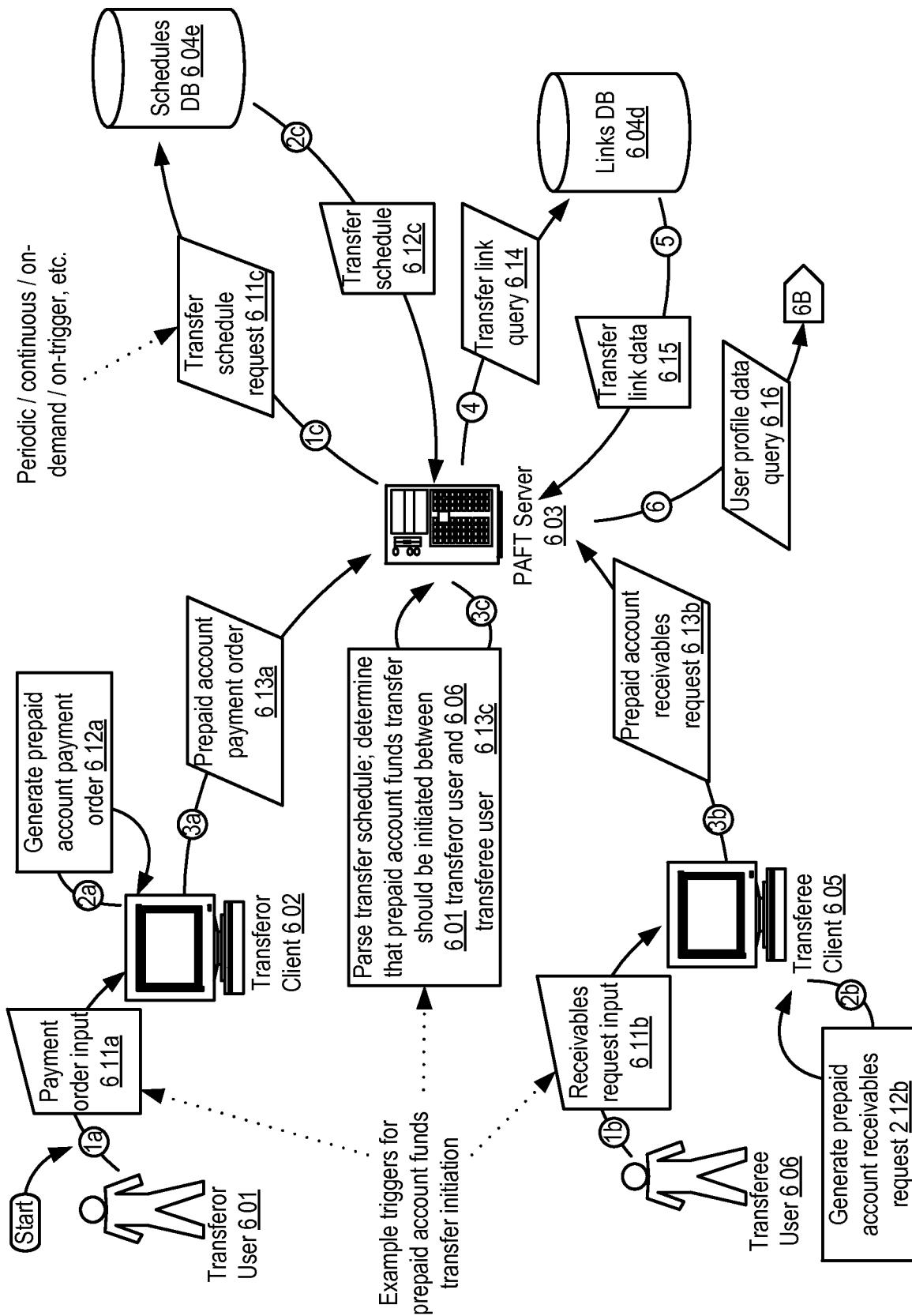


FIGURE 6A

Example: Prepaid Account Funds Transfer

13/21

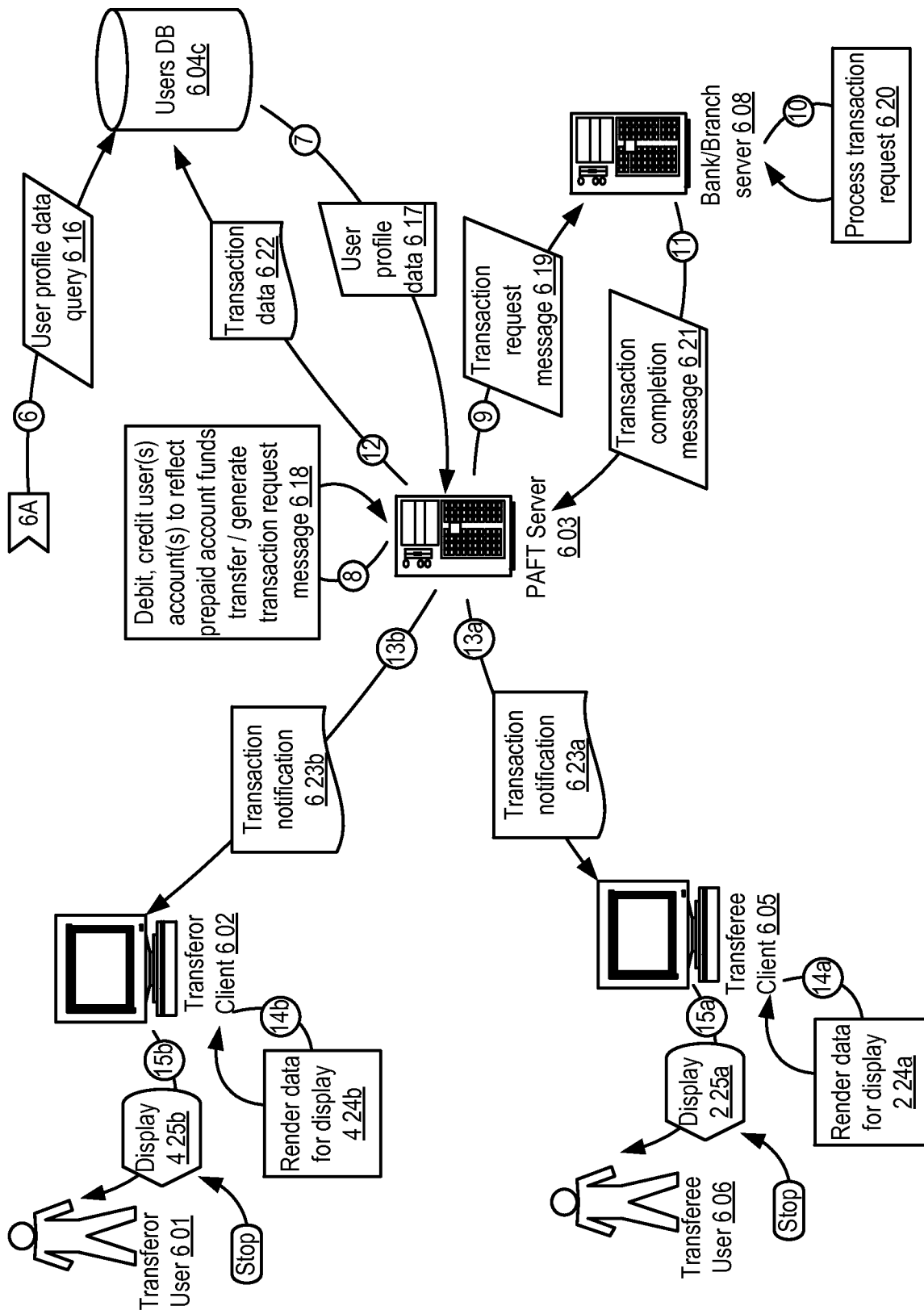


FIGURE 6B

Example: Prepaid Account Funds Transfer

14/21

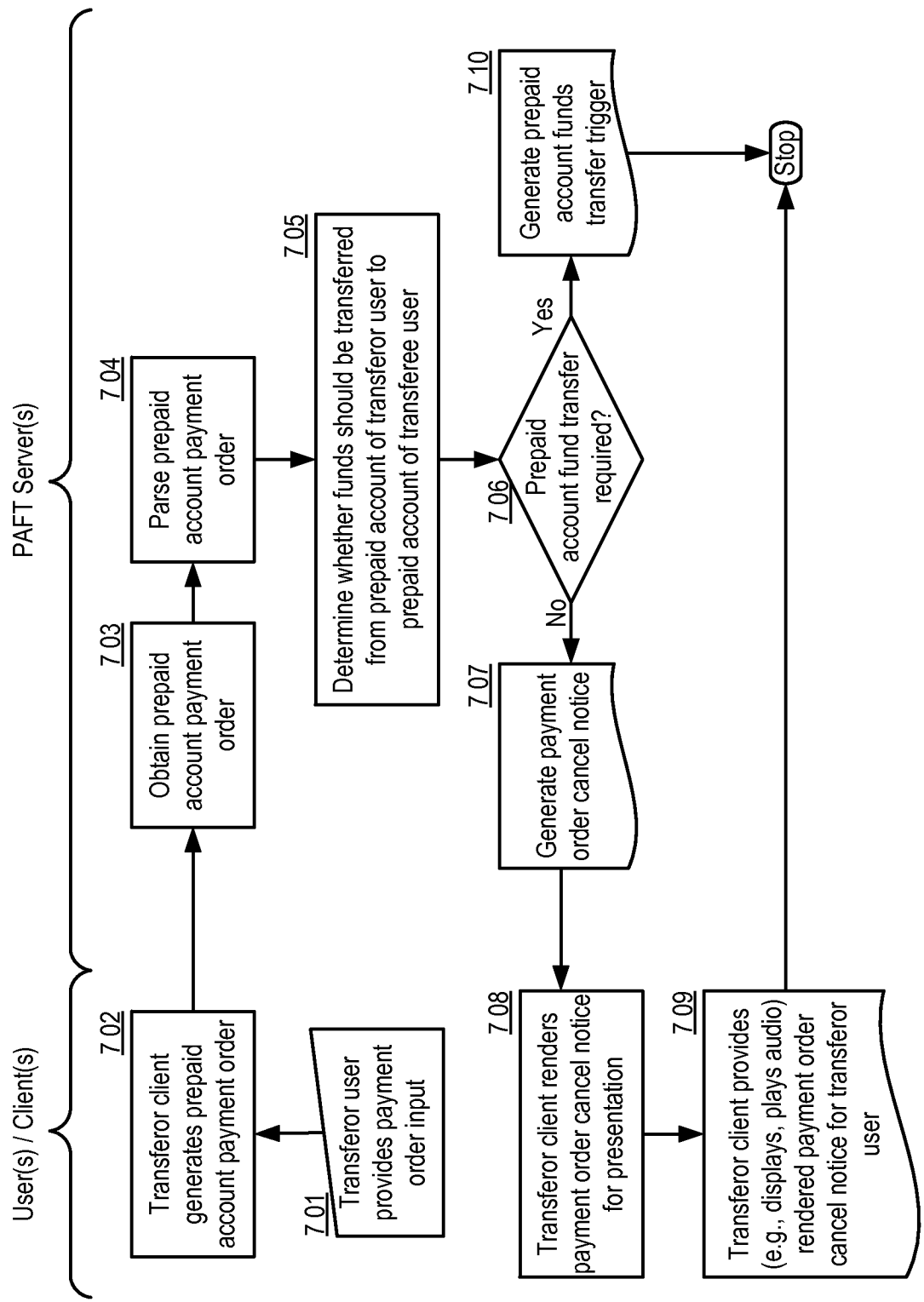


FIGURE 7 Example: Transferor-Initiated Funds Transfer Trigger (Tr-FTT) component 700

15/21

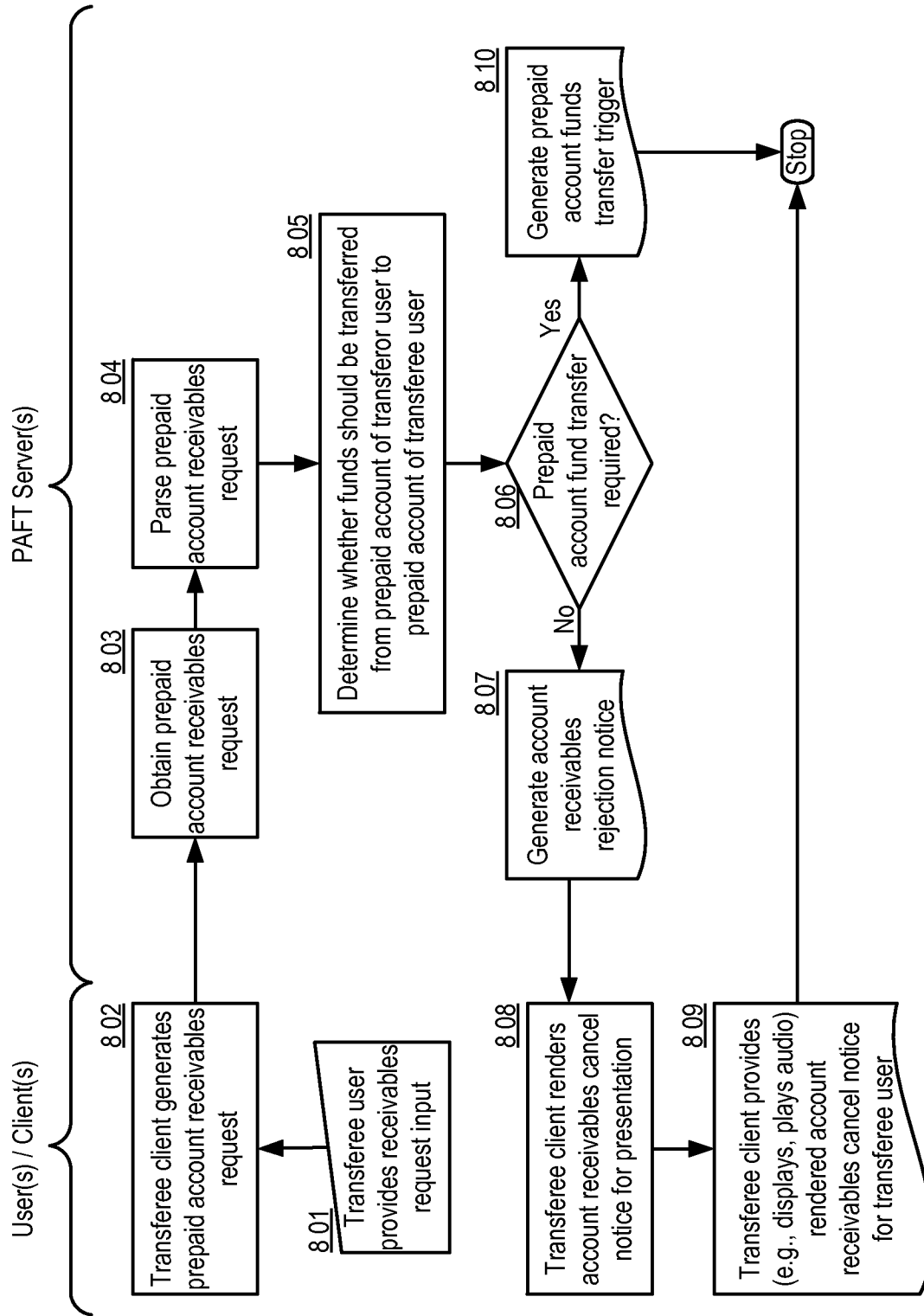


FIGURE 8

Example: Transferee-Initiated Funds Transfer Trigger (Te-FTT) component 800

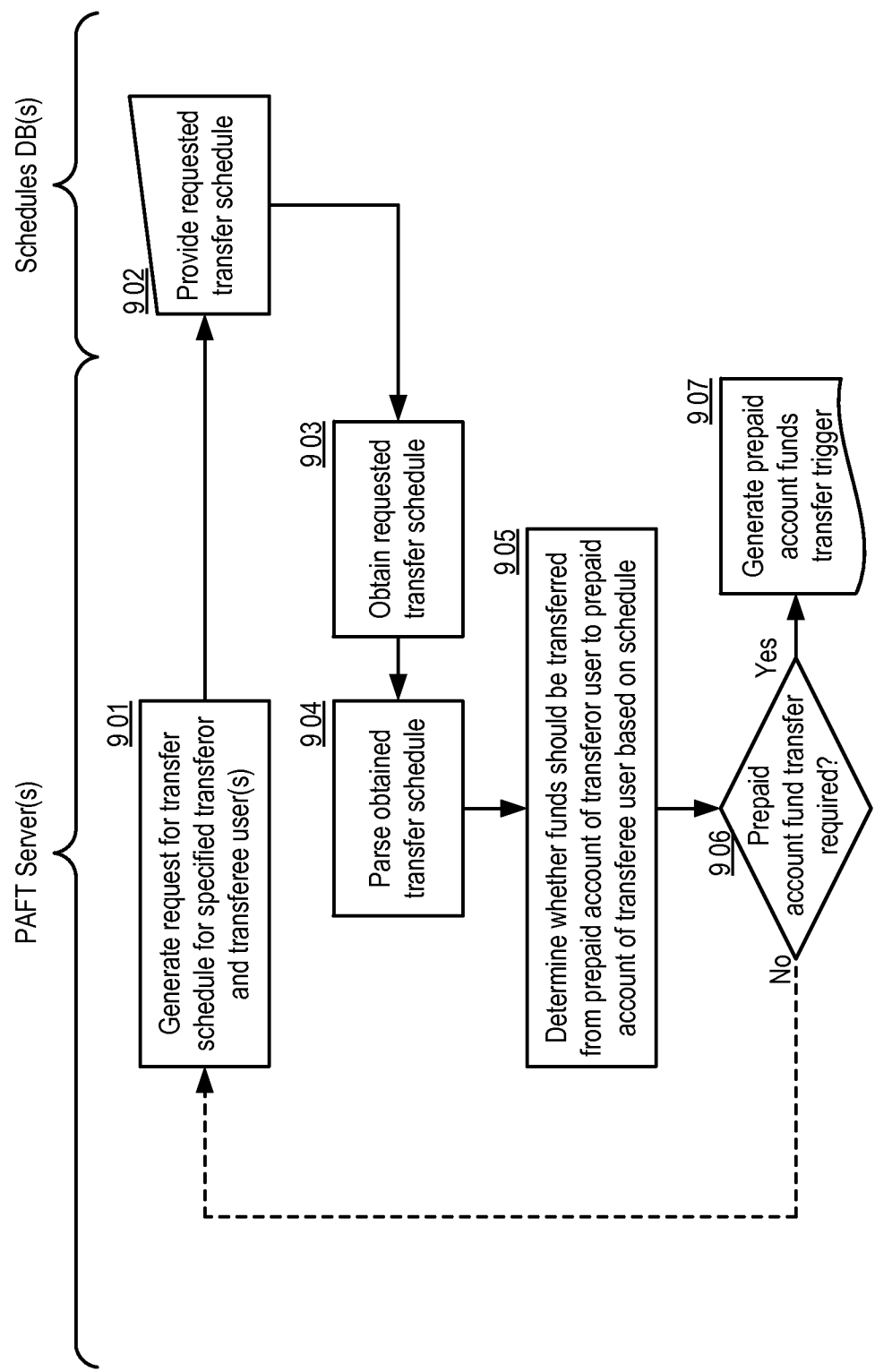


FIGURE 9

Example: Server-Initiated Funds Transfer Trigger (S-FTT) component 900

17/21

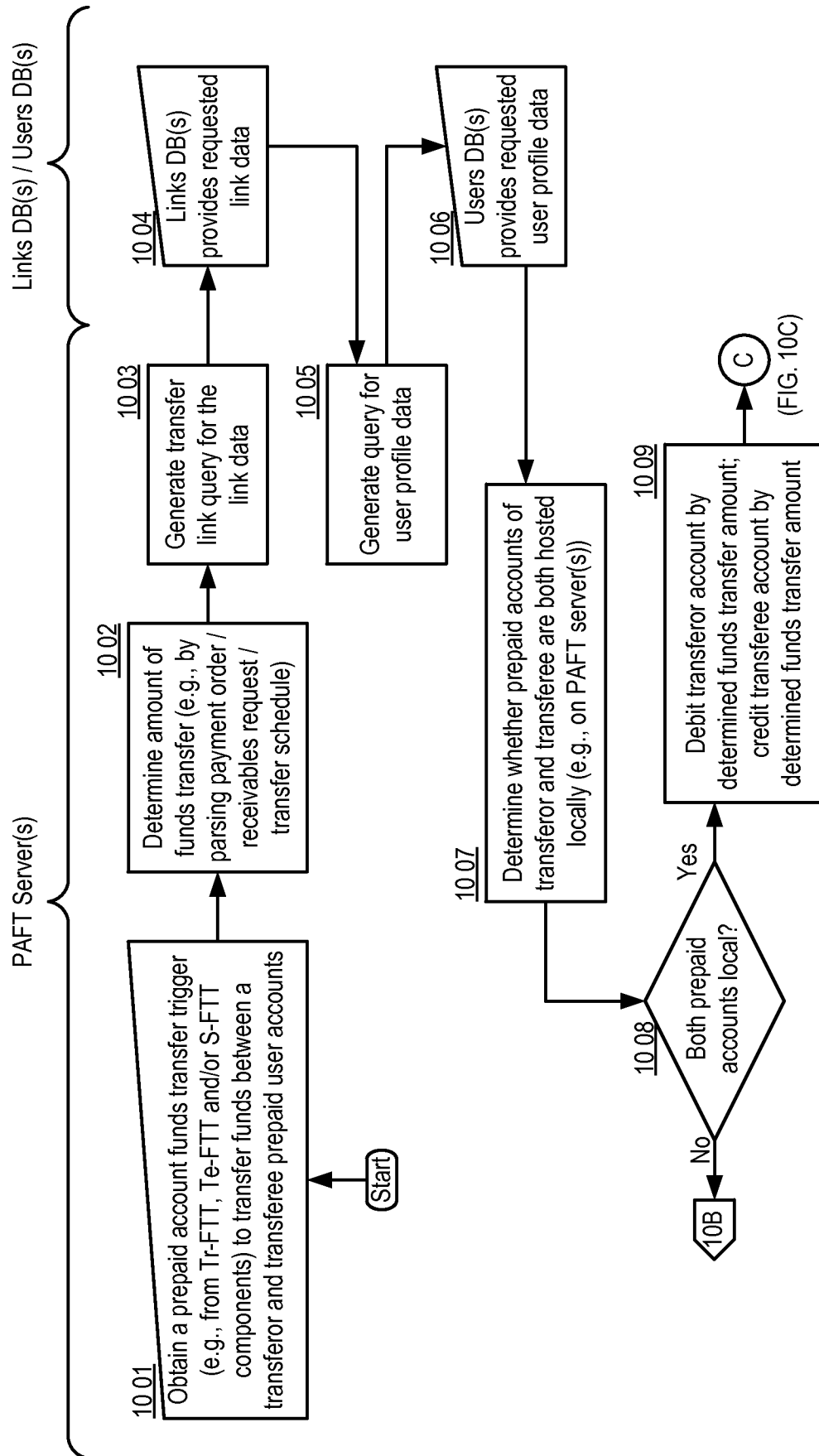


FIGURE 10A

Example: Prepaid Account Funds Transfer Processing (PA-FTP) component 1000

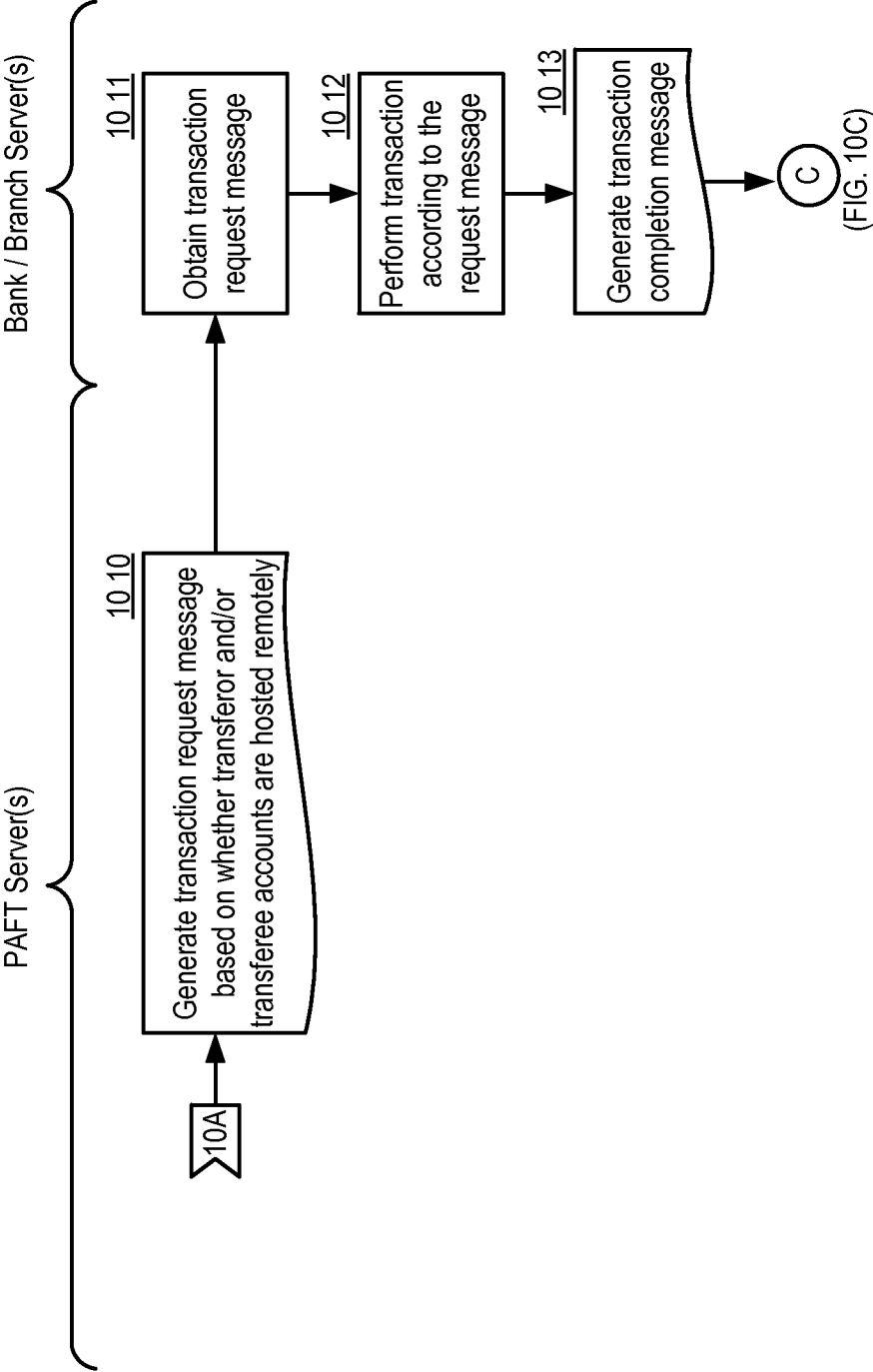


FIGURE 10B

Example: Prepaid Account Funds Transfer Processing (PA-FTP) component 1000

19/21

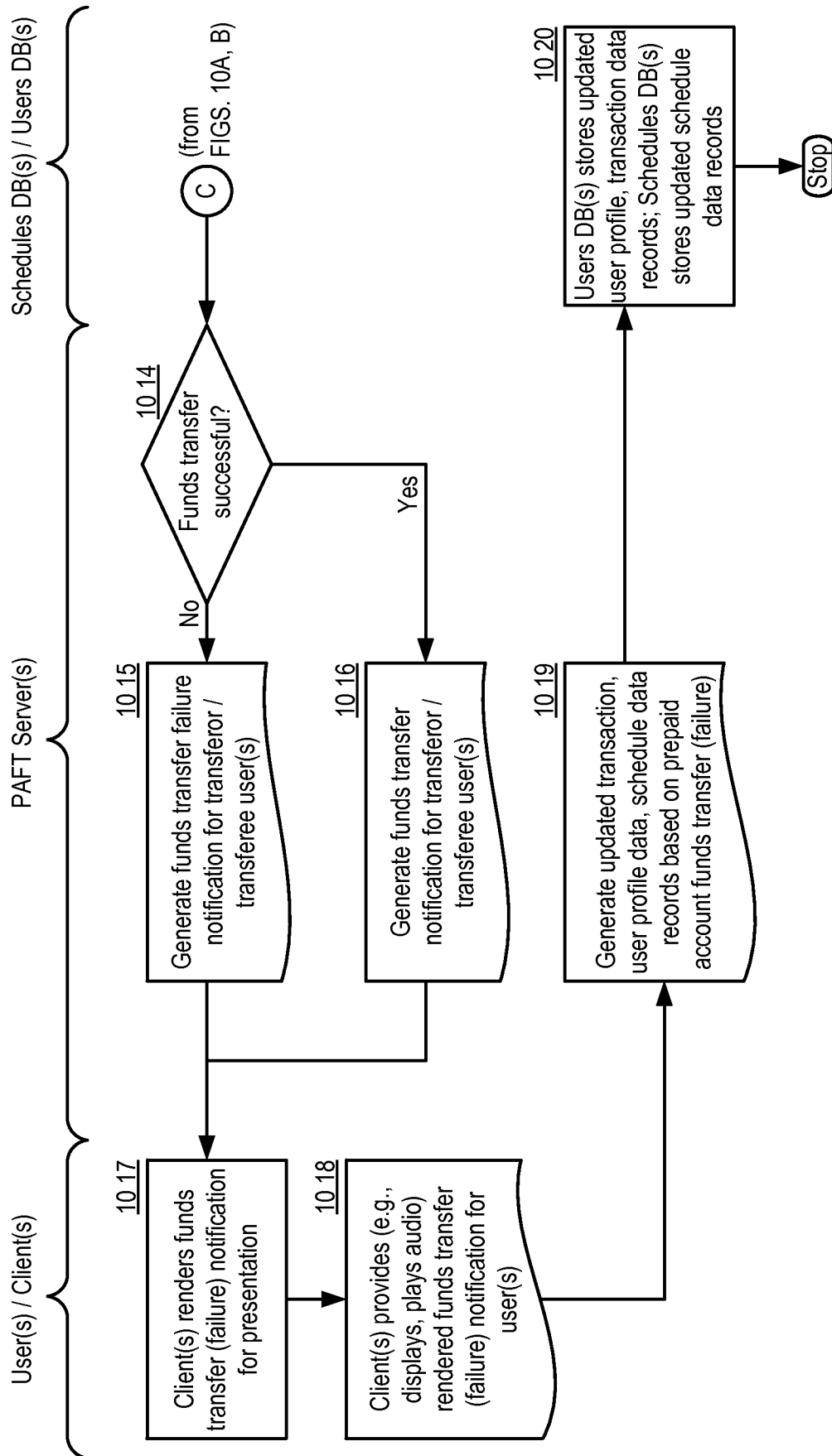


FIGURE 10C

Example: Prepaid Account Funds Transfer Processing (PA-FTP) component 1000

20/21

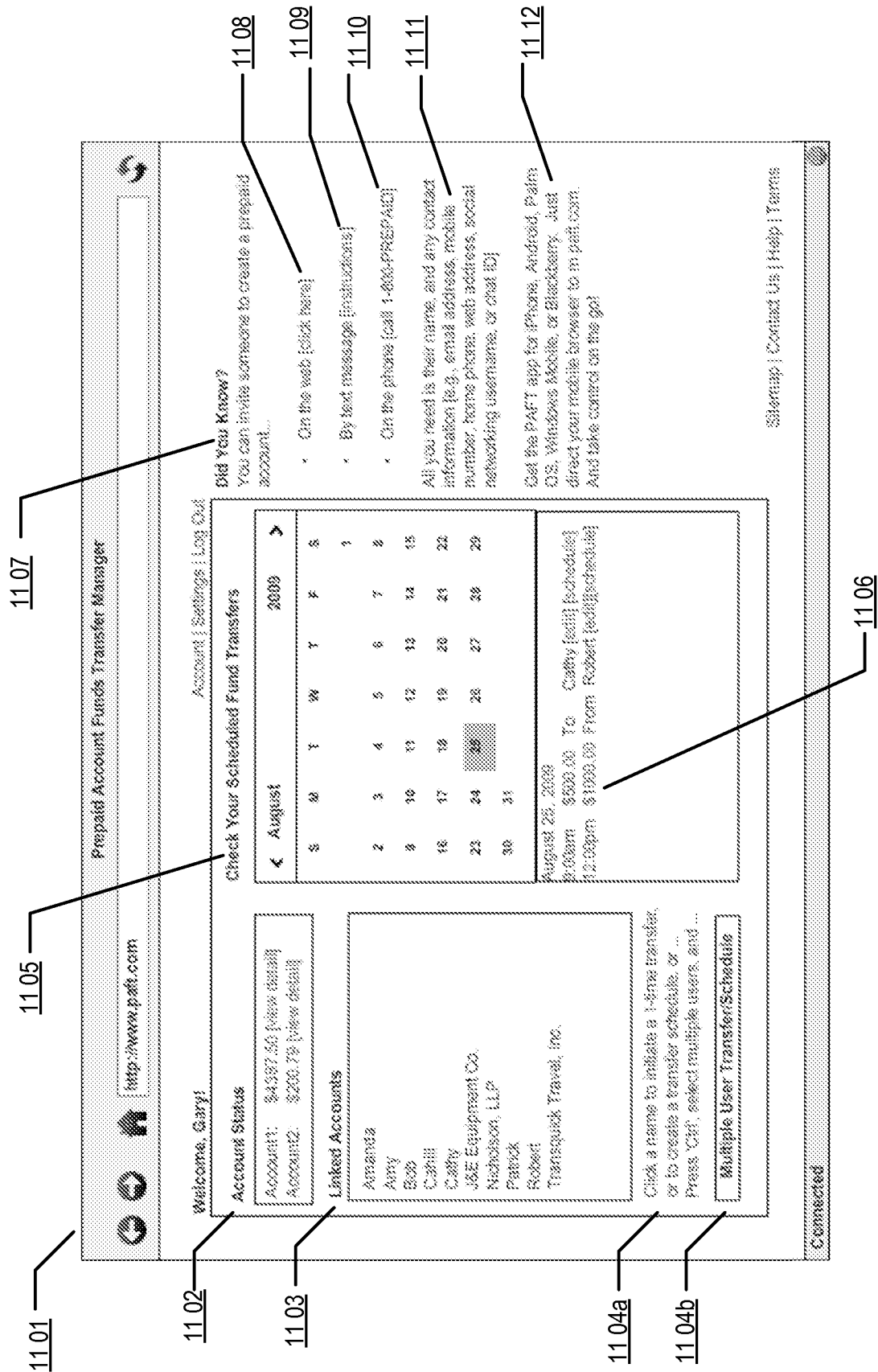
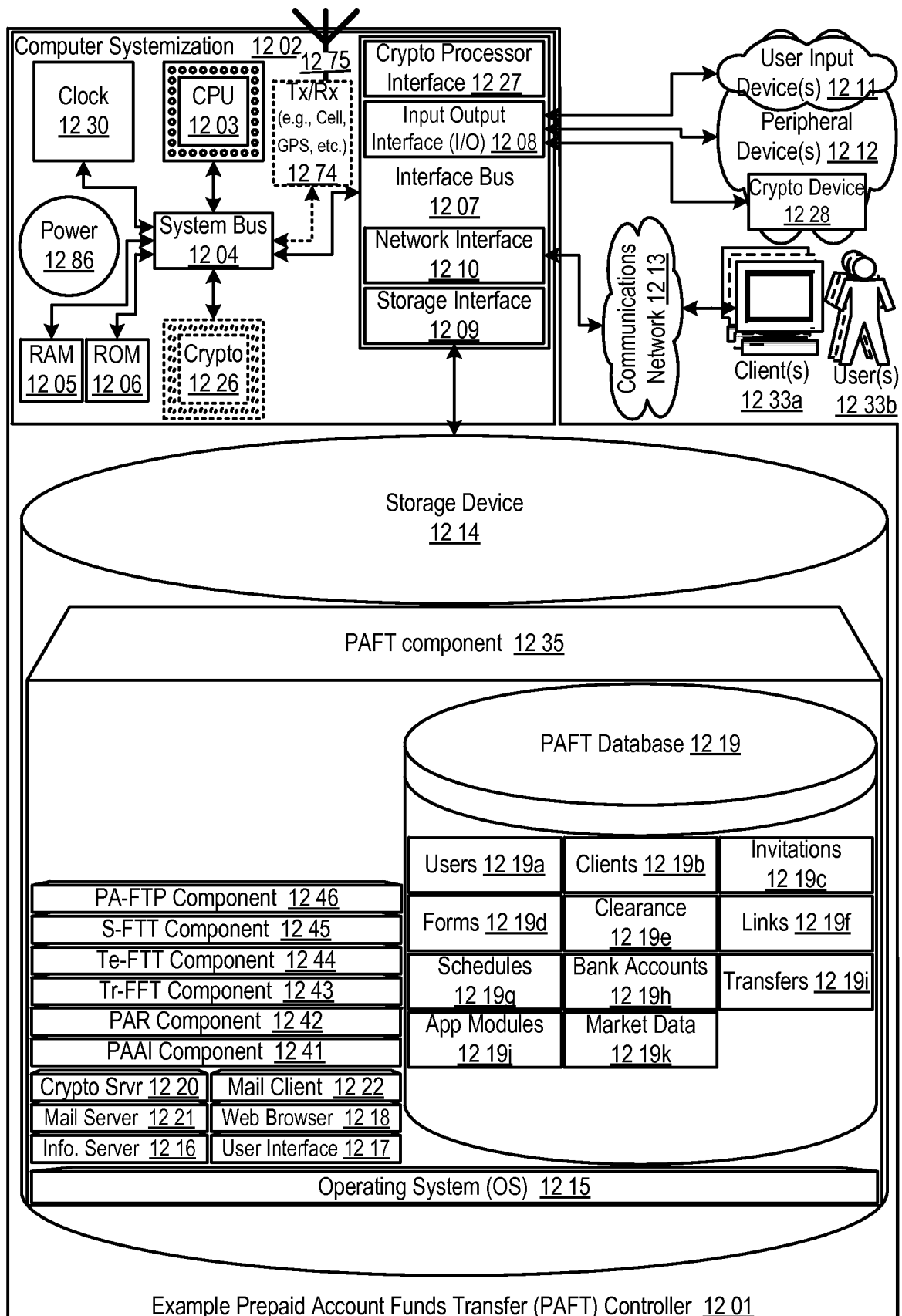


FIGURE 11

Example: Prepaid Account Funds Transfer Management User Interface

FIGURE 12

21/21



INTERNATIONAL SEARCH REPORT

International Application No.

PCT/US2011/024941

A. CLASSIFICATION OF SUBJECT MATTER

IPC(8) - G06Q 20/30 (2011.01)

USPC - 705/35

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC(8) - G06Q 20/30; G06Q 30/00; G07F 19/00; H04W 4/24 (2011.01)

USPC - 705/35, 39, 40, 44

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Google Patents, Google Scholar, PatBase, Orbit

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 6,934,528 B2 (LOUREIRO et al) 23 August 2005 (23.08.2005) entire document	1-21
Y	US 2003/0080185 A1 (WERTHER) 01 May 2003 (01.05.2003) entire document	1-21
Y	US 7,356,505 B2 (MARCH) 08 April 2008 (08.04.2008) entire document	2-3, 9-10, 16-17
A	US 2007/0208662 A1 (JERONIMUS et al) 06 September 2007 (06.09.2007) entire document	1-21
A	US 2006/0277143 A1 (ALMONTE et al) 07 December 2006 (07.12.2006) entire document	1-21

☐ Further documents are listed in the continuation of Box C.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

31 March 2011

Date of mailing of the international search report

19 APR 2011

Name and mailing address of the ISA/US

Mail Stop PCT, Attn: ISA/US, Commissioner for Patents
P.O. Box 1450, Alexandria, Virginia 22313-1450

Facsimile No. 571-273-3201

Authorized officer:

Blaine R. Copenheaver

PCT Helpdesk: 571-272-4300
PCT OSP: 571-272-7774