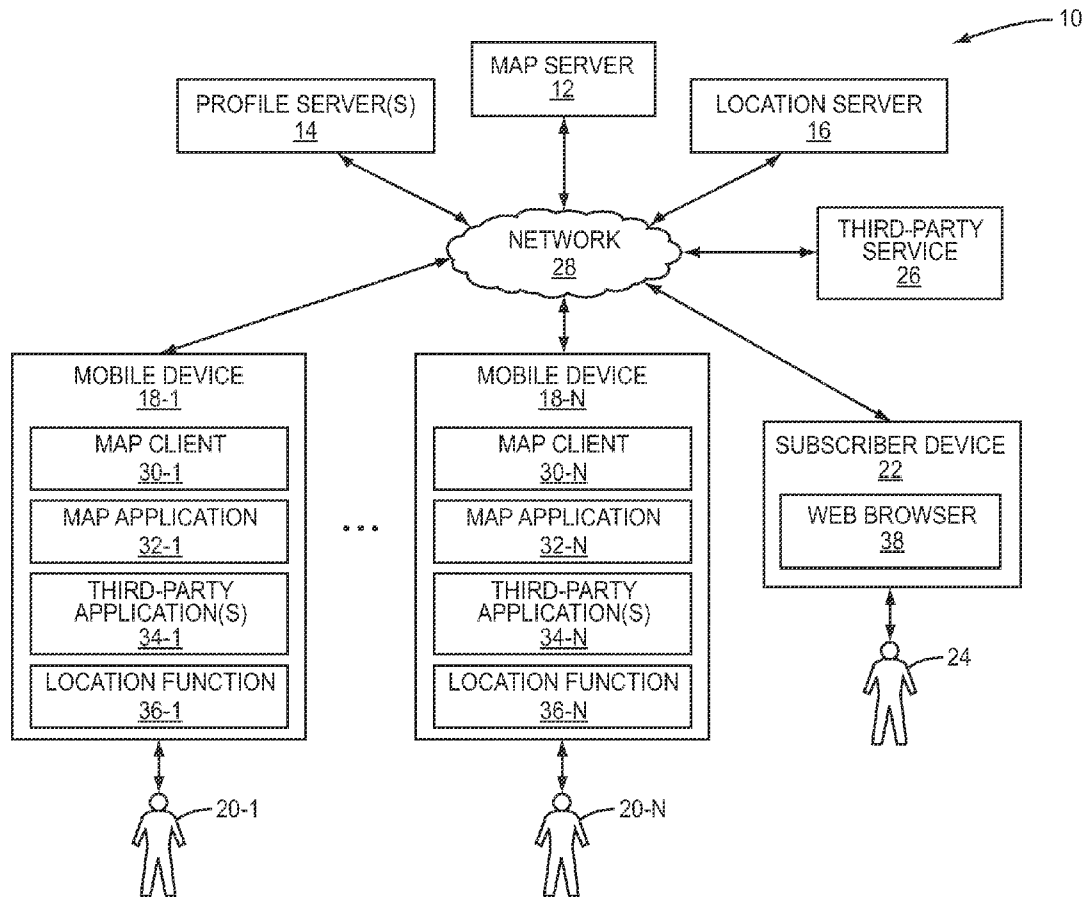


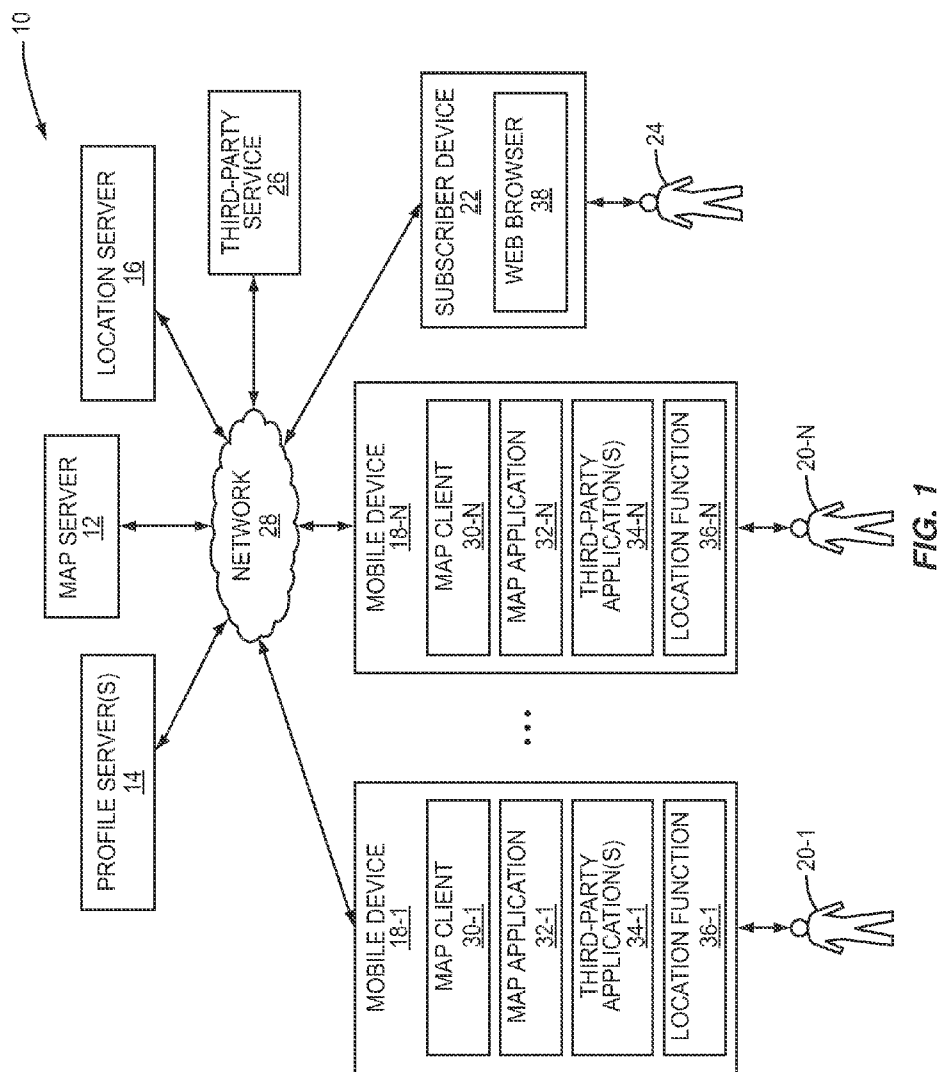


US 20120046017A1

(19) **United States**(12) **Patent Application Publication**
Jennings(10) **Pub. No.: US 2012/0046017 A1**(43) **Pub. Date: Feb. 23, 2012**(54) **SYSTEM AND METHOD FOR PREVENTION
OF INDIRECT USER TRACKING THROUGH
AGGREGATE PROFILE DATA****Publication Classification**(51) **Int. Cl.****H04W 4/00** (2009.01)**G06F 15/173** (2006.01)**G06F 15/16** (2006.01)**H04W 24/00** (2009.01)(75) Inventor: **Kenneth Jennings**, Raleigh, NC
(US)(73) Assignee: **Waldeck Technology, LLC**,
Wilmington, DE (US)(52) **U.S. Cl. 455/414.1; 455/456.1; 709/225;
709/233**(21) Appl. No.: **12/768,973**(57) **ABSTRACT**(22) Filed: **Apr. 28, 2010****Related U.S. Application Data**(60) Provisional application No. 61/173,625, filed on Apr.
29, 2009.

Systems and methods are provided for preventing indirect user tracking in a mobile aggregate profiling system. In general, access to aggregate profile data for groups of users is controlled in a mobile aggregate profiling system based on rate of change values for one or more characteristics, such as population, of the groups of users.





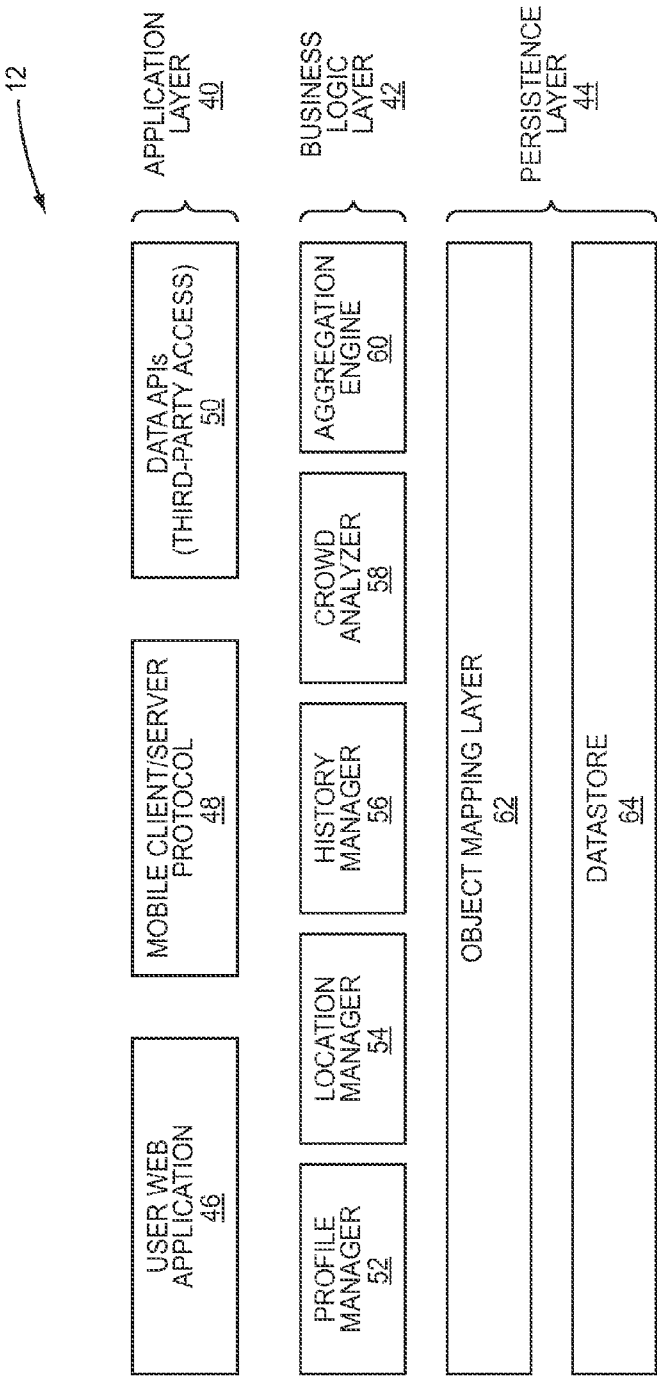


FIG. 2

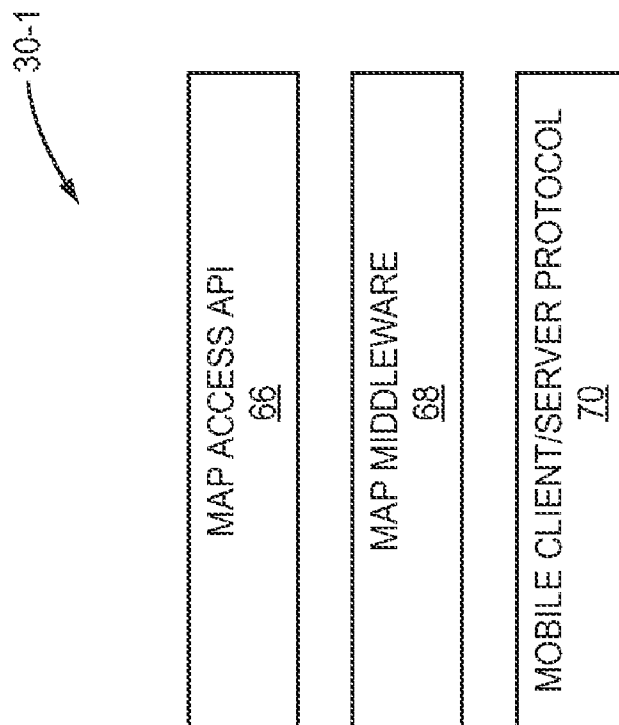


FIG. 3

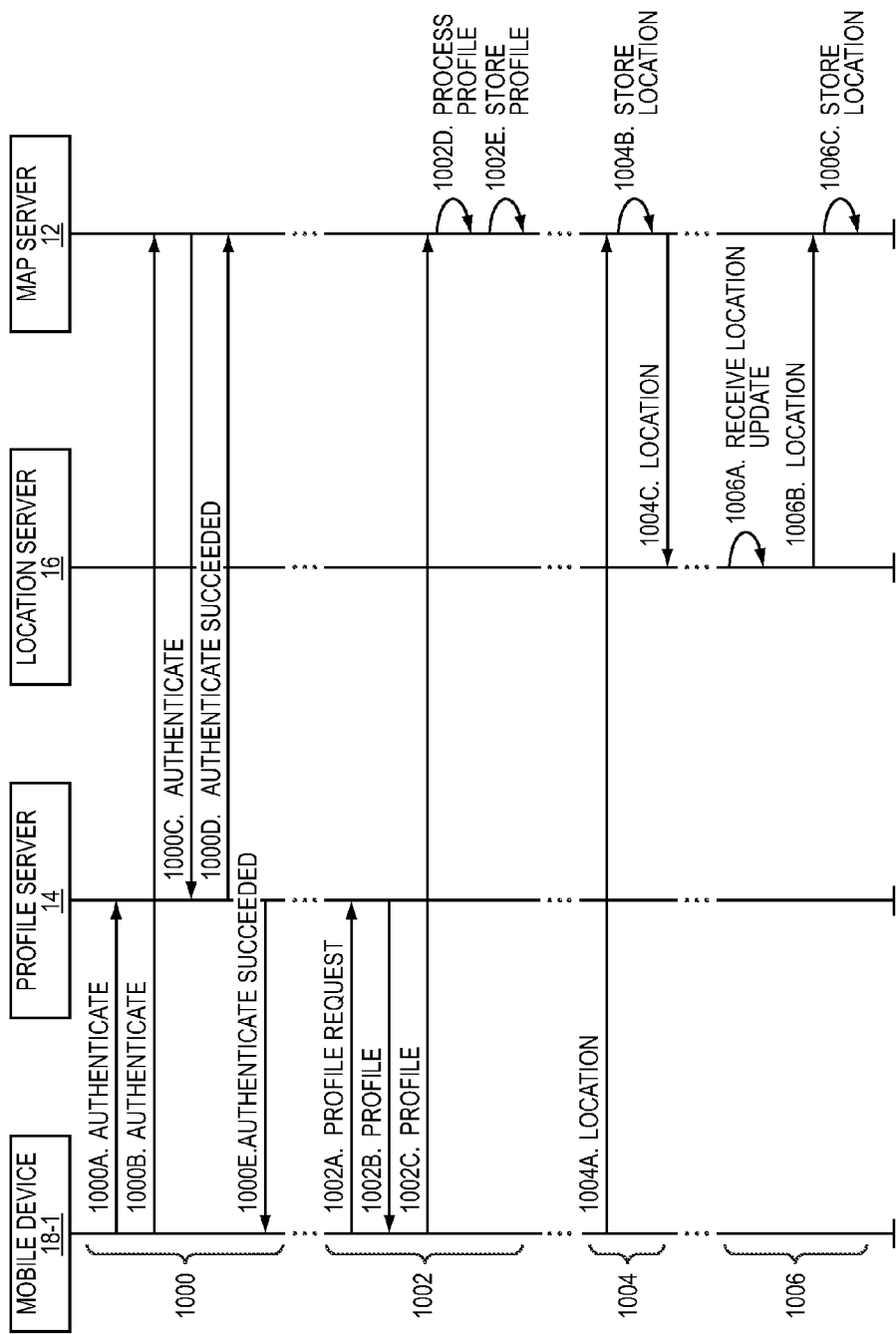


FIG. 4

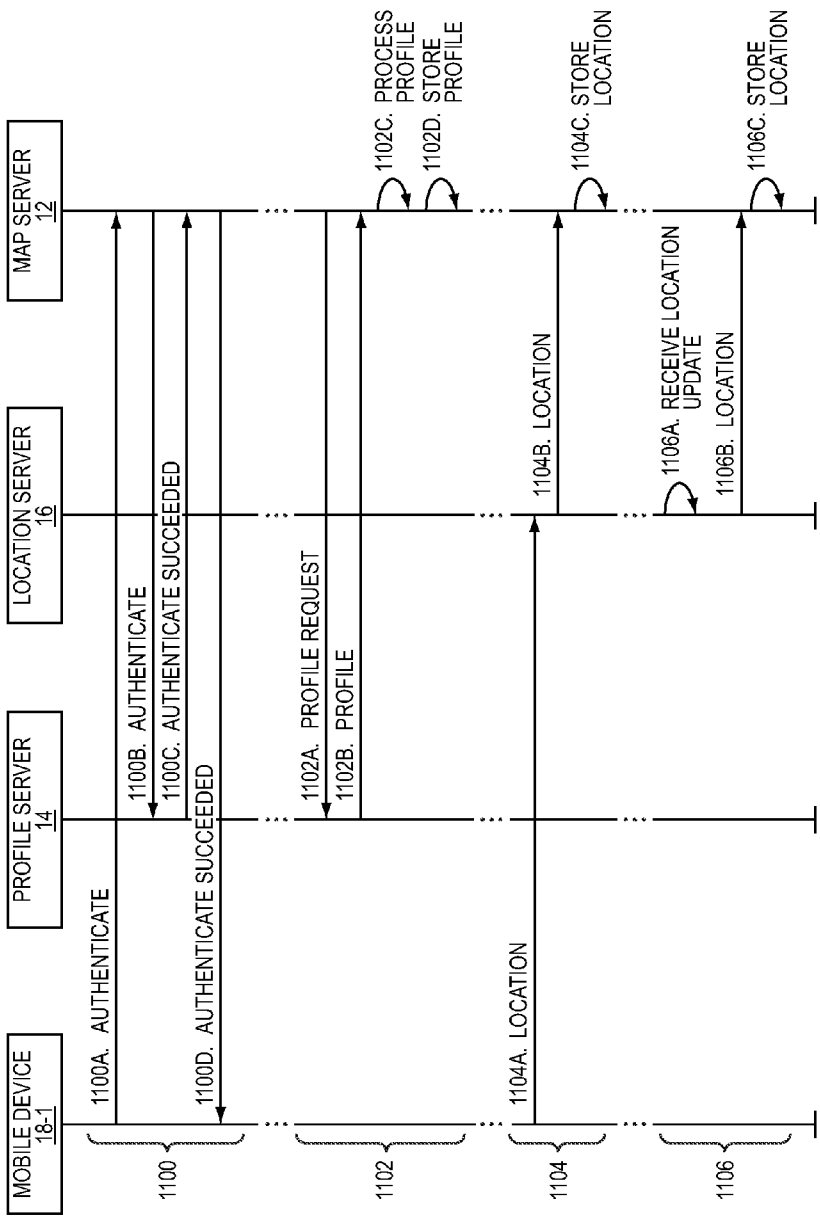


FIG. 5

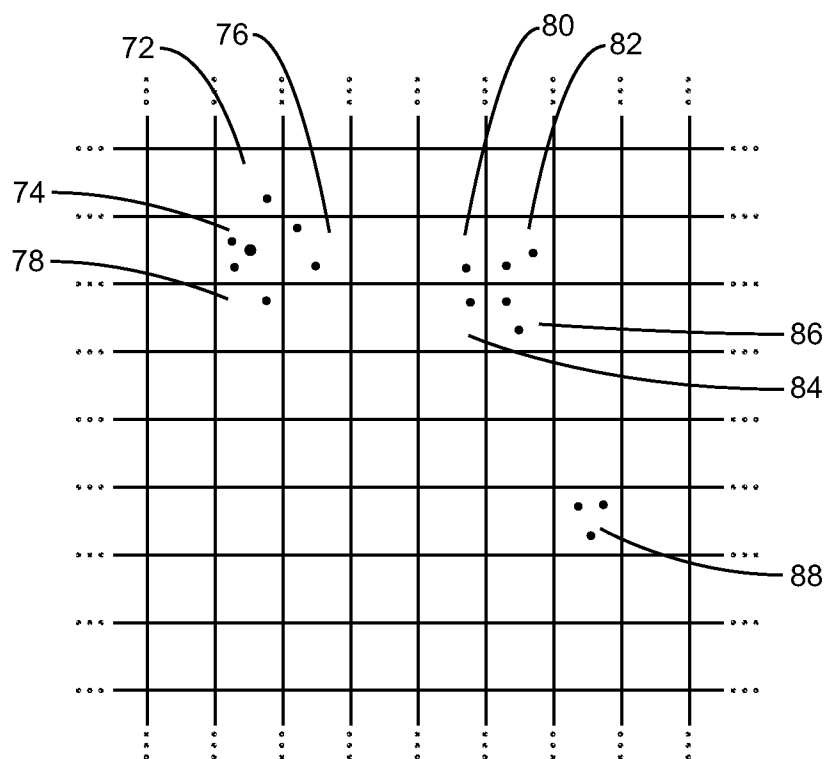


FIG. 6

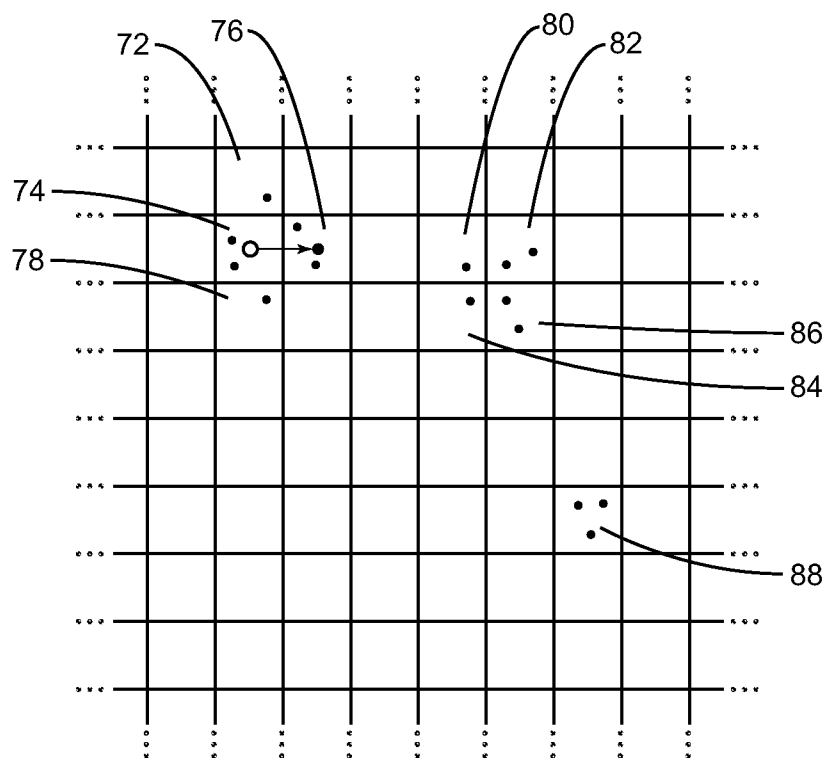
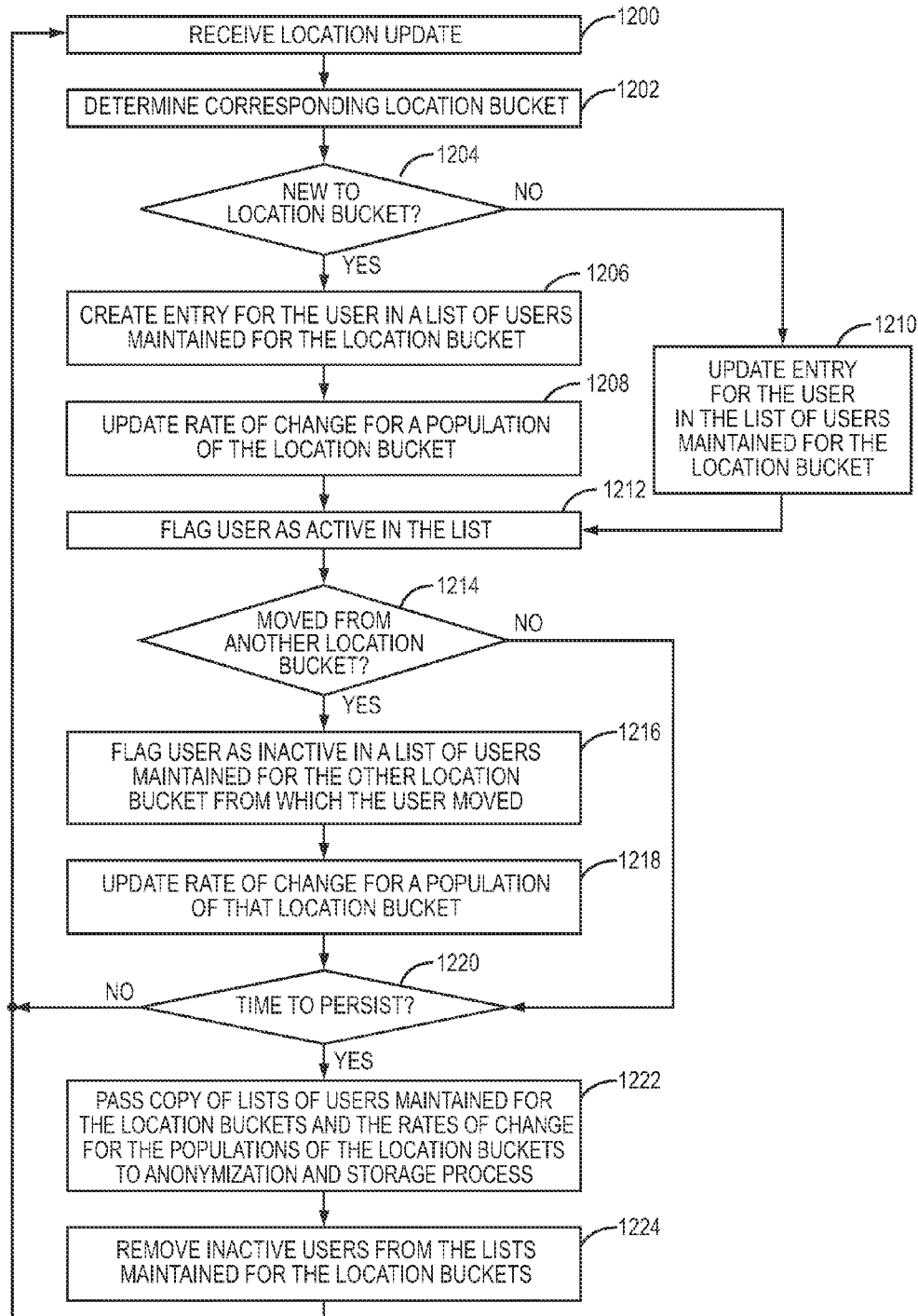
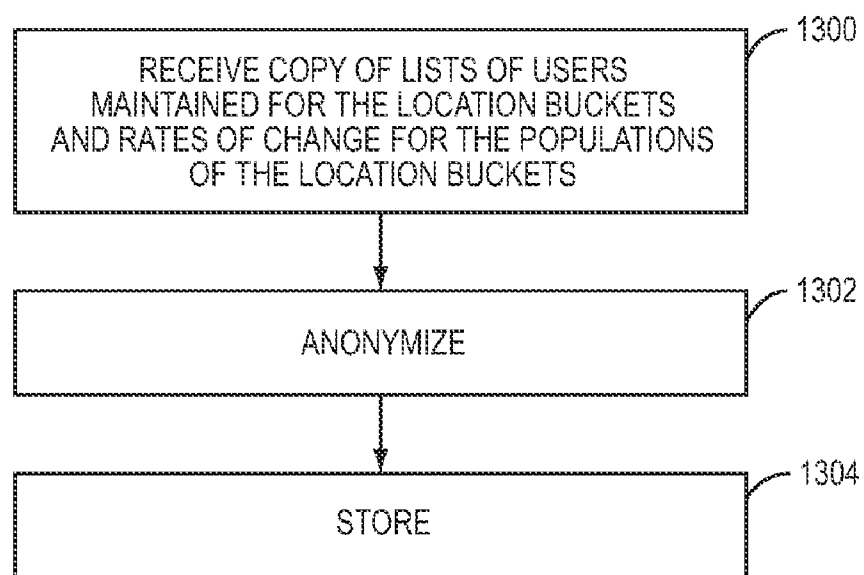


FIG. 7

**FIG. 8**

**FIG. 9**

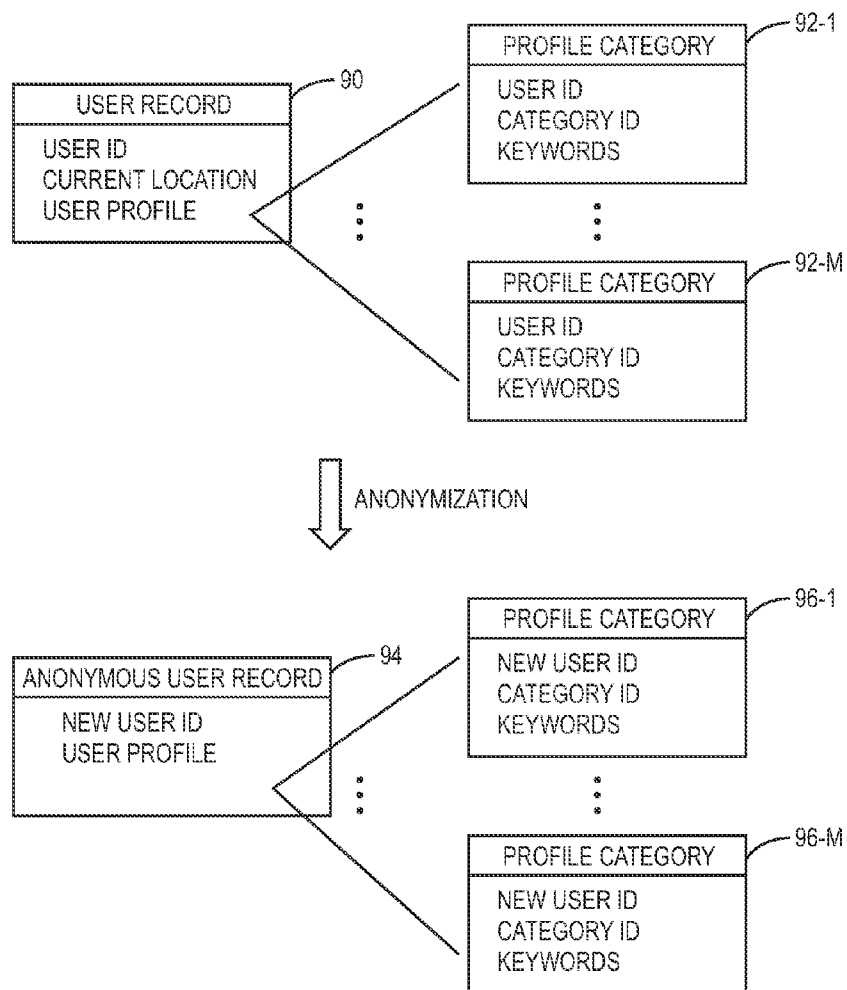


FIG. 10

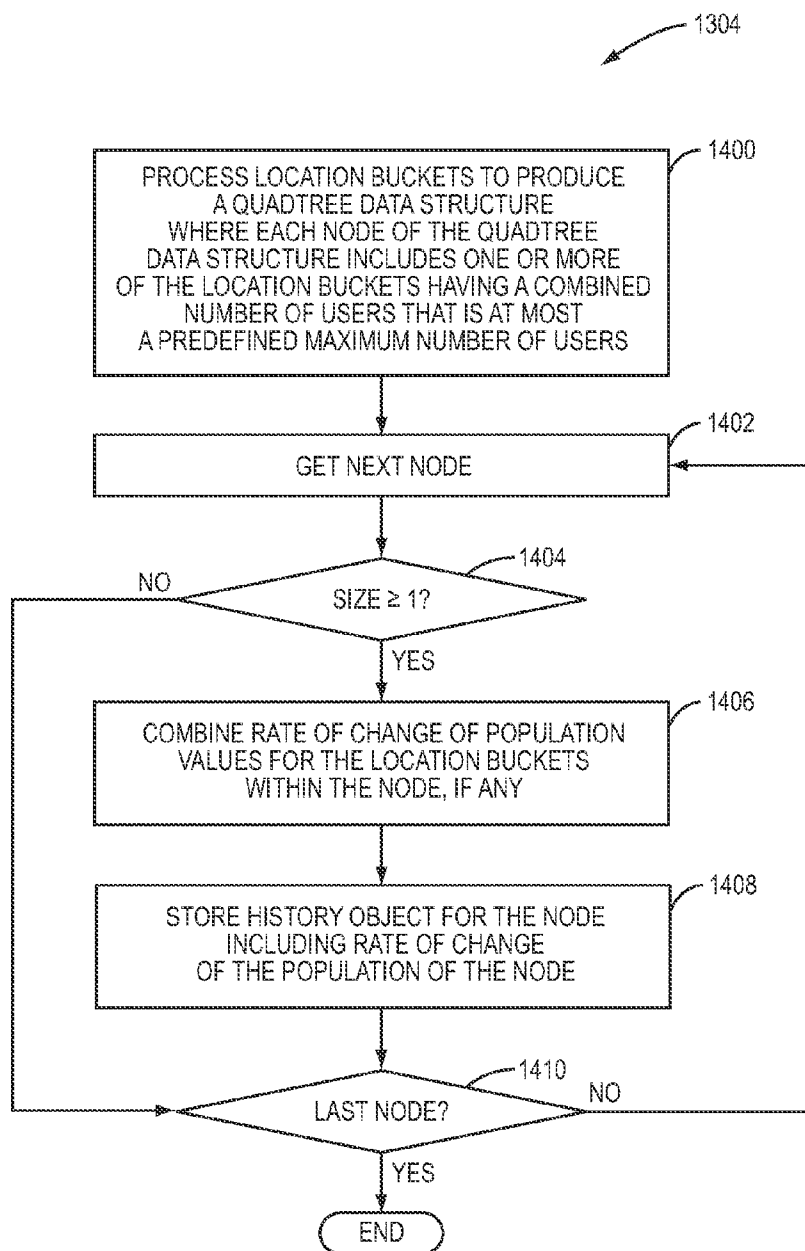


FIG. 11

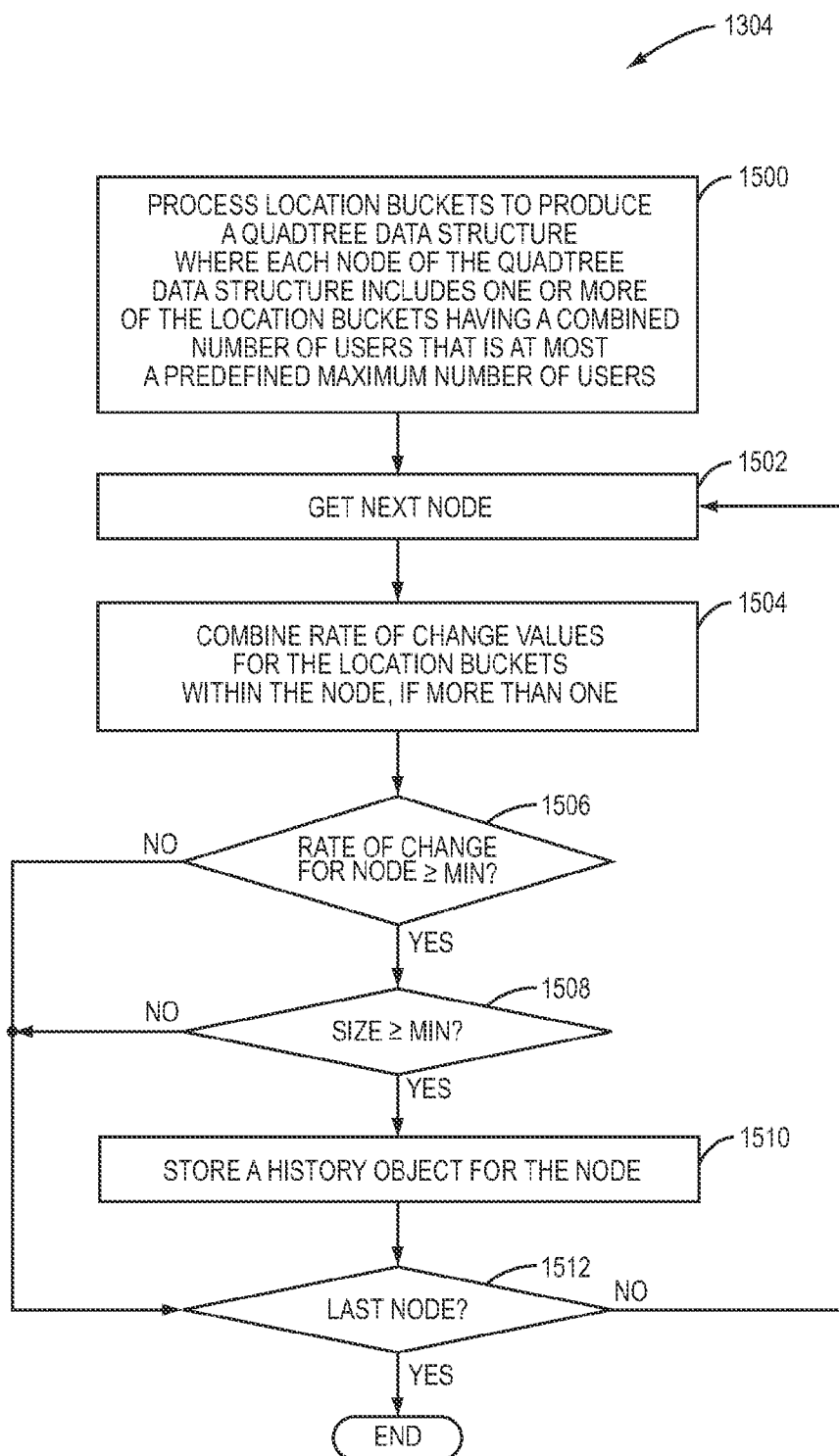


FIG. 12

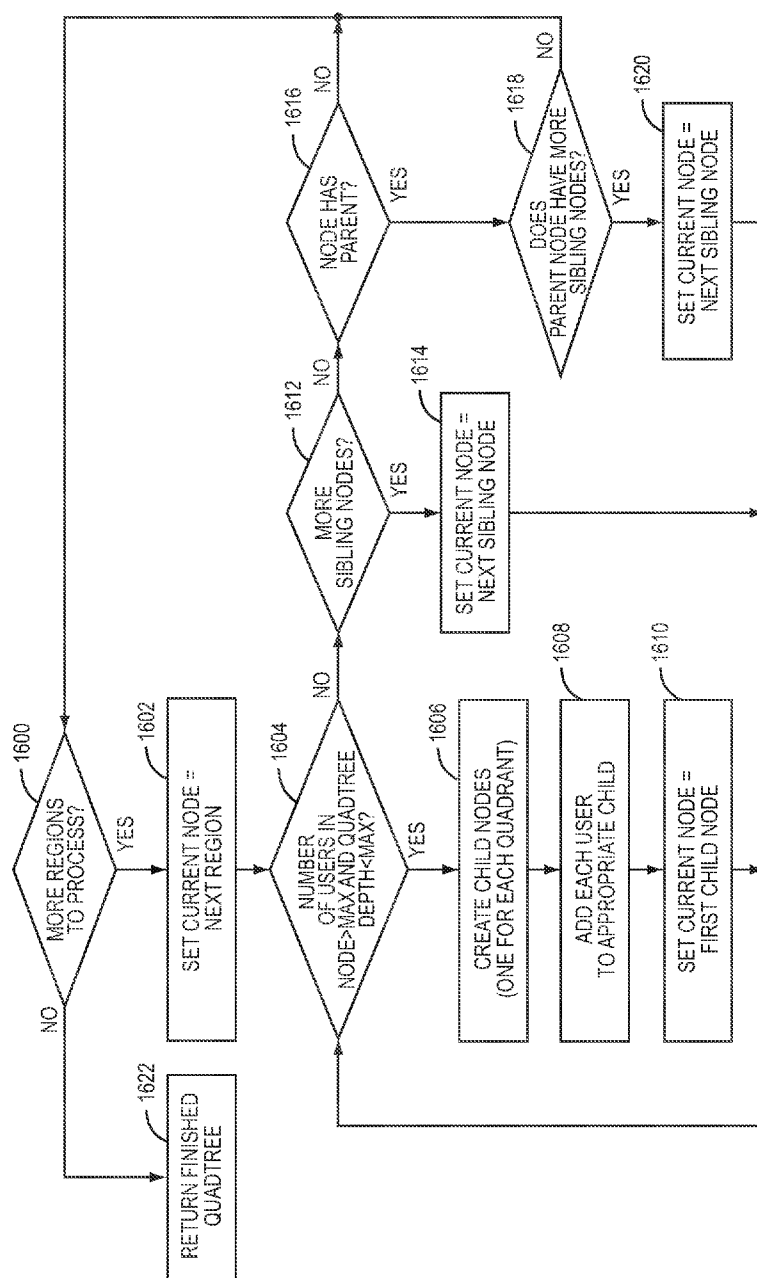


FIG. 13

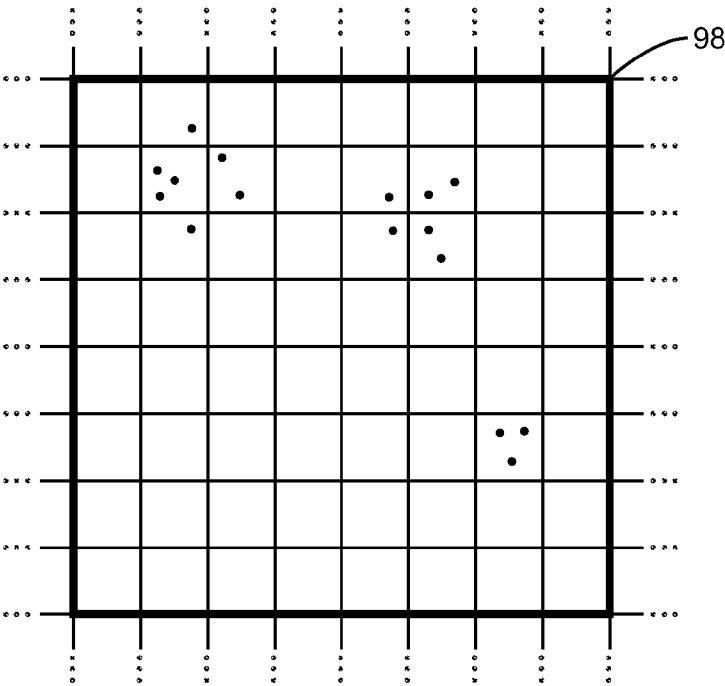


FIG. 14A

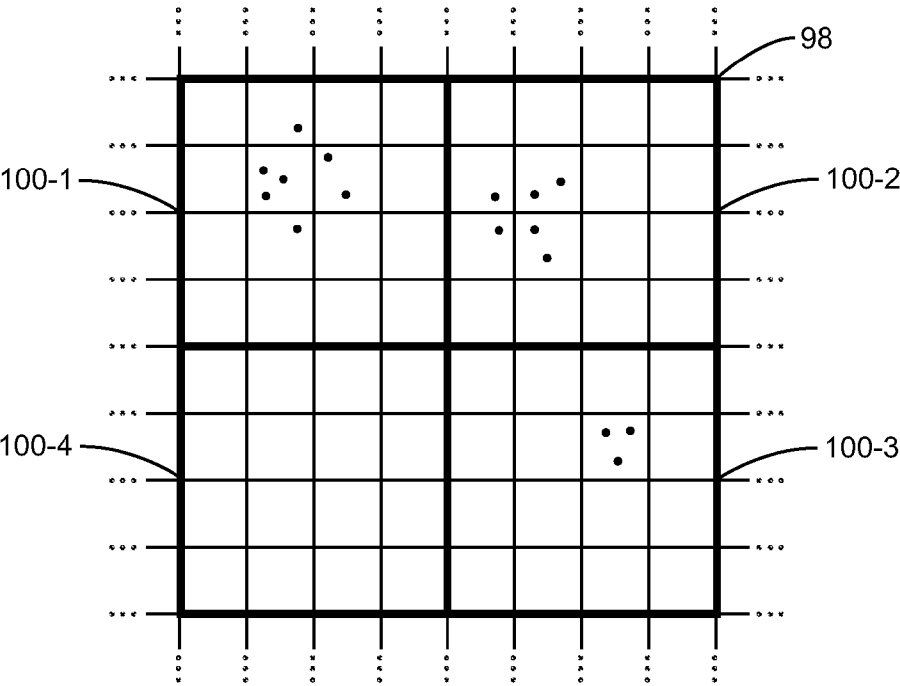


FIG. 14B

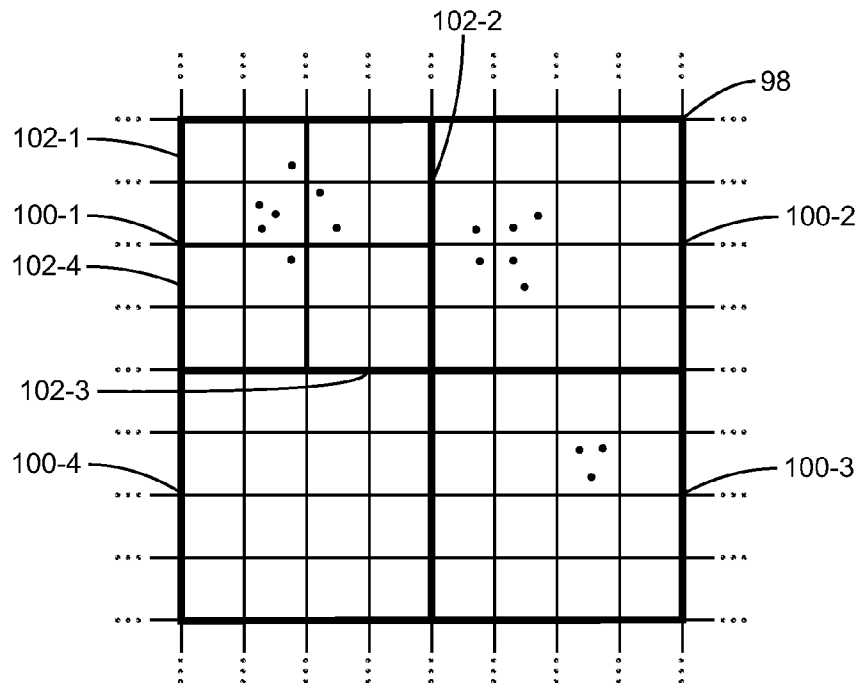


FIG. 14C

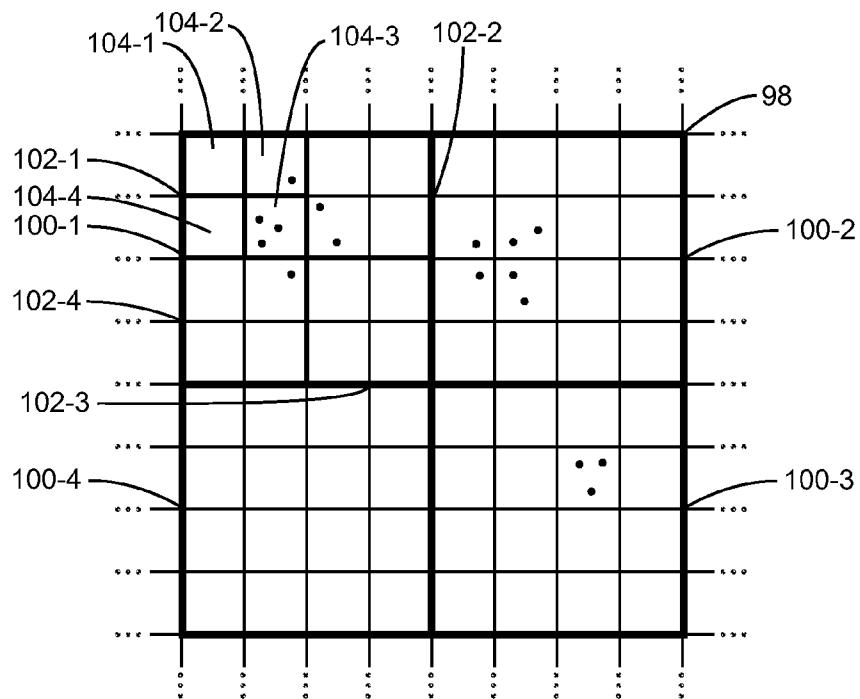


FIG. 14D

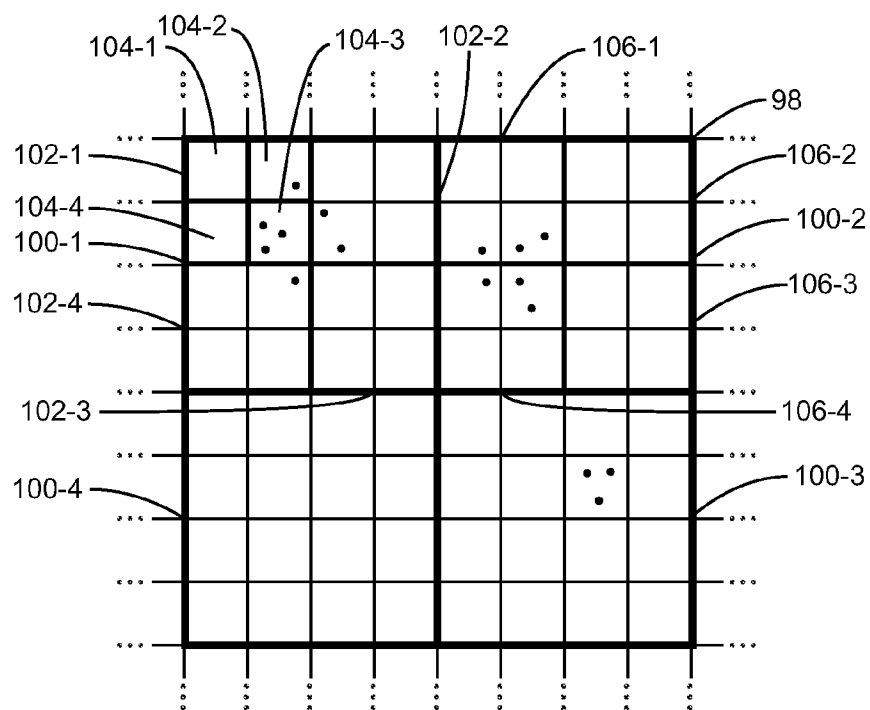


FIG. 14E

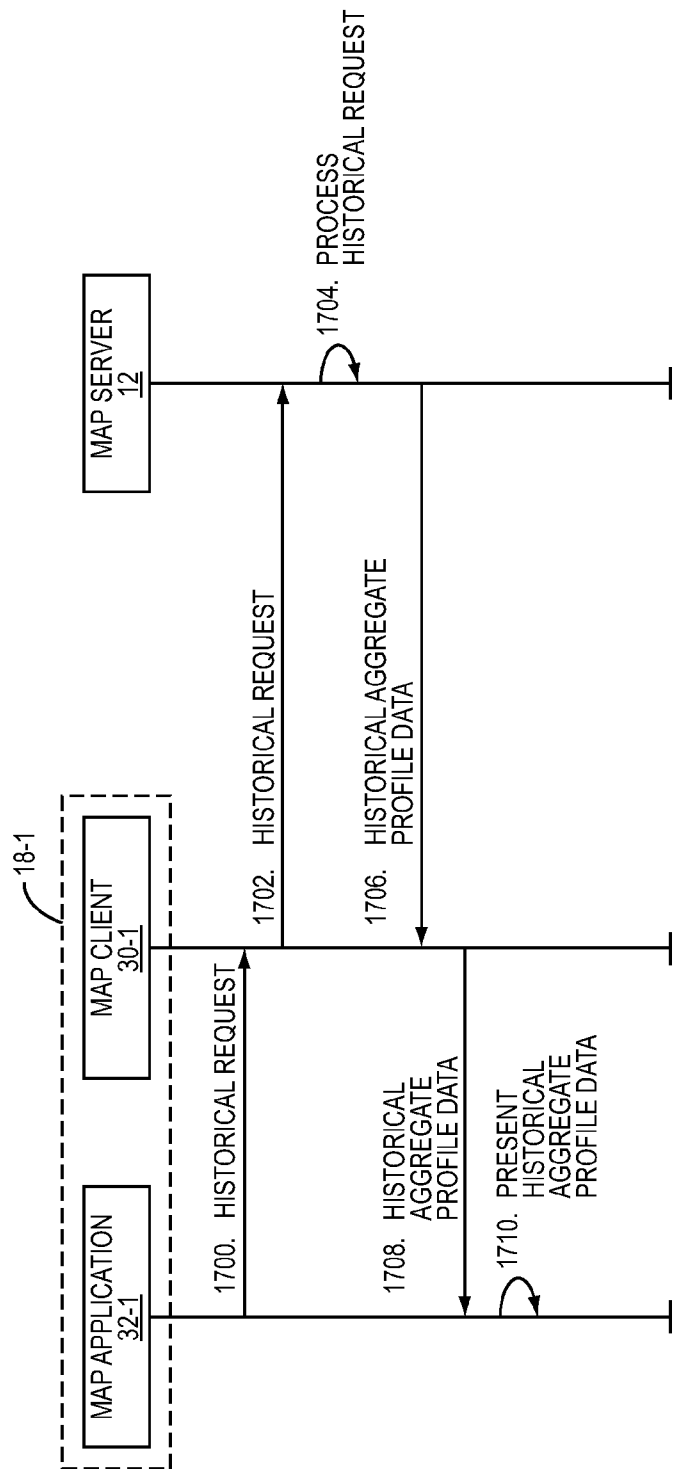


FIG. 15

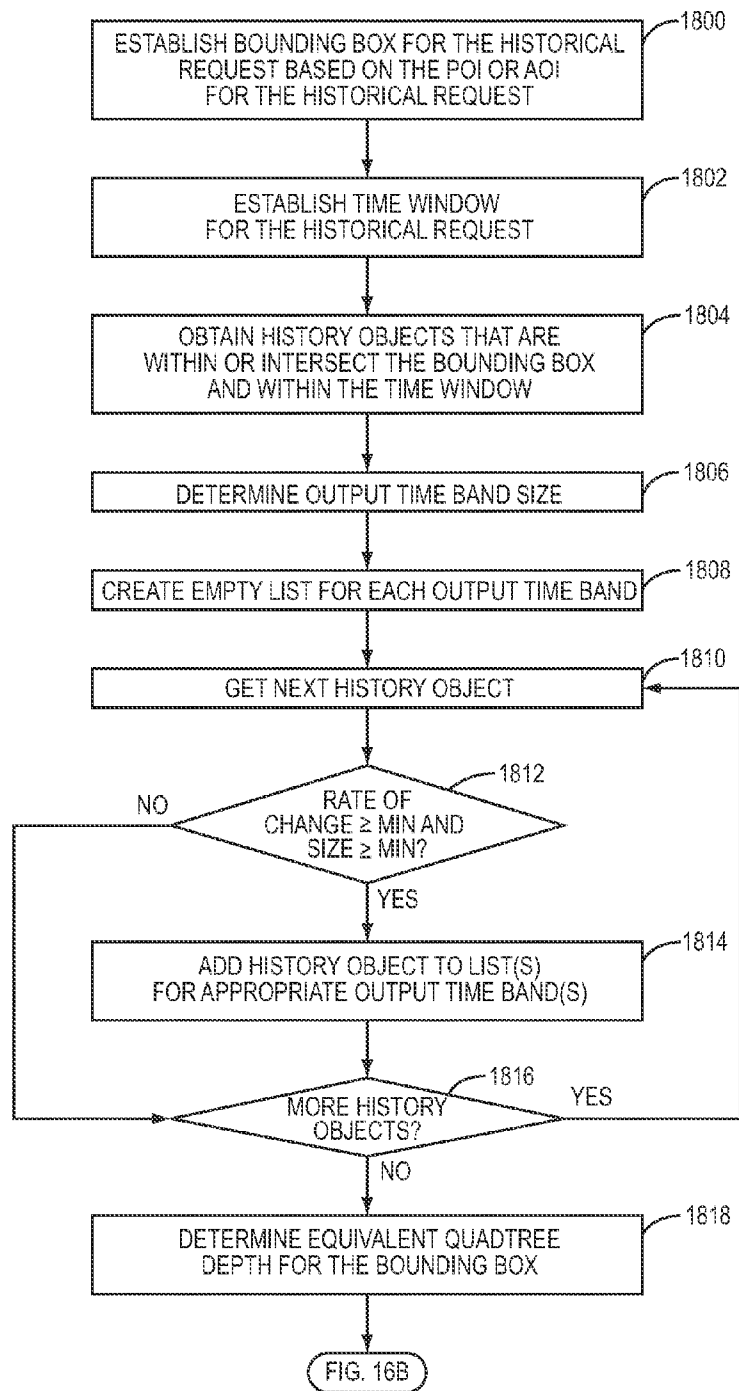
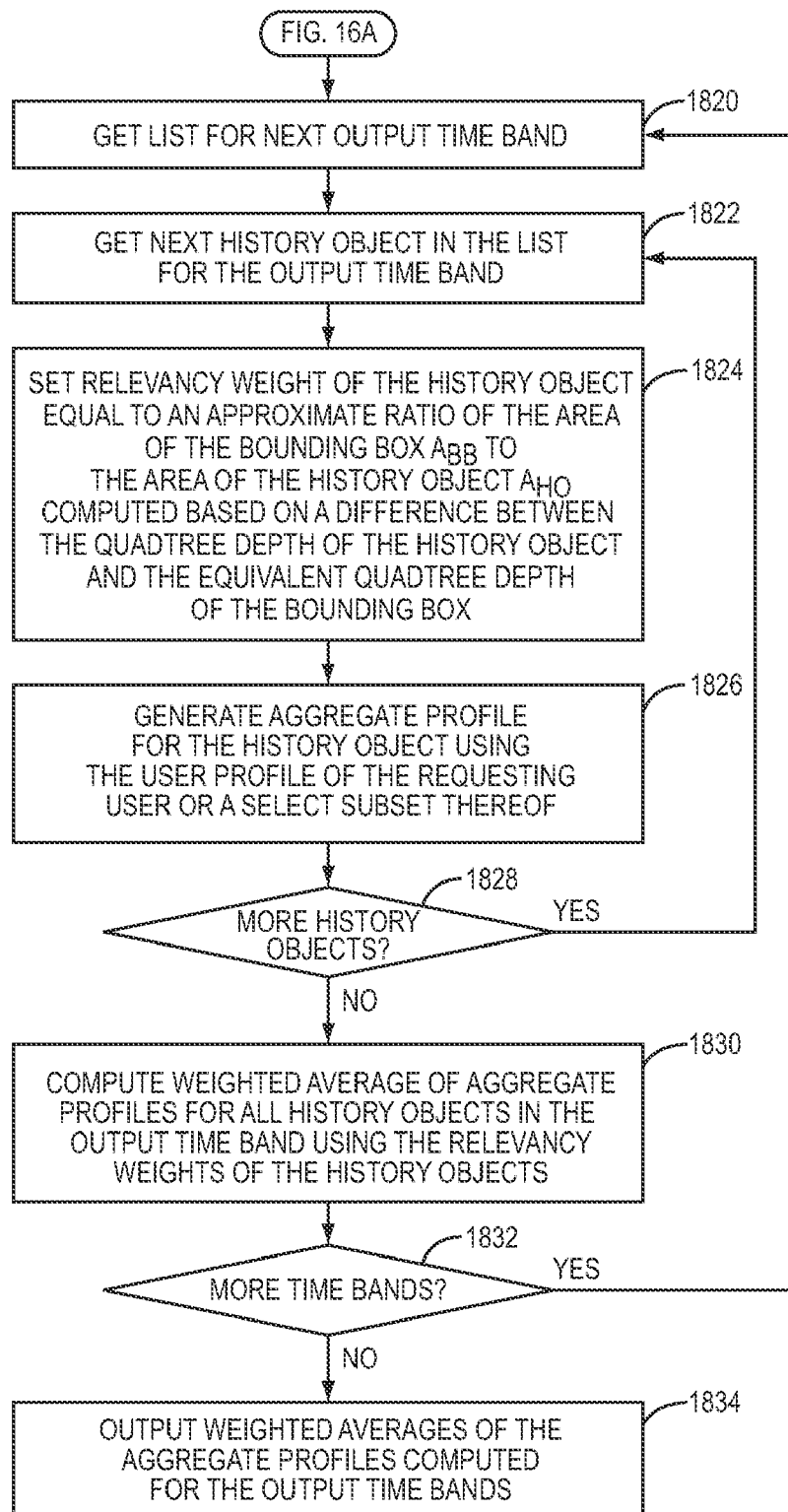
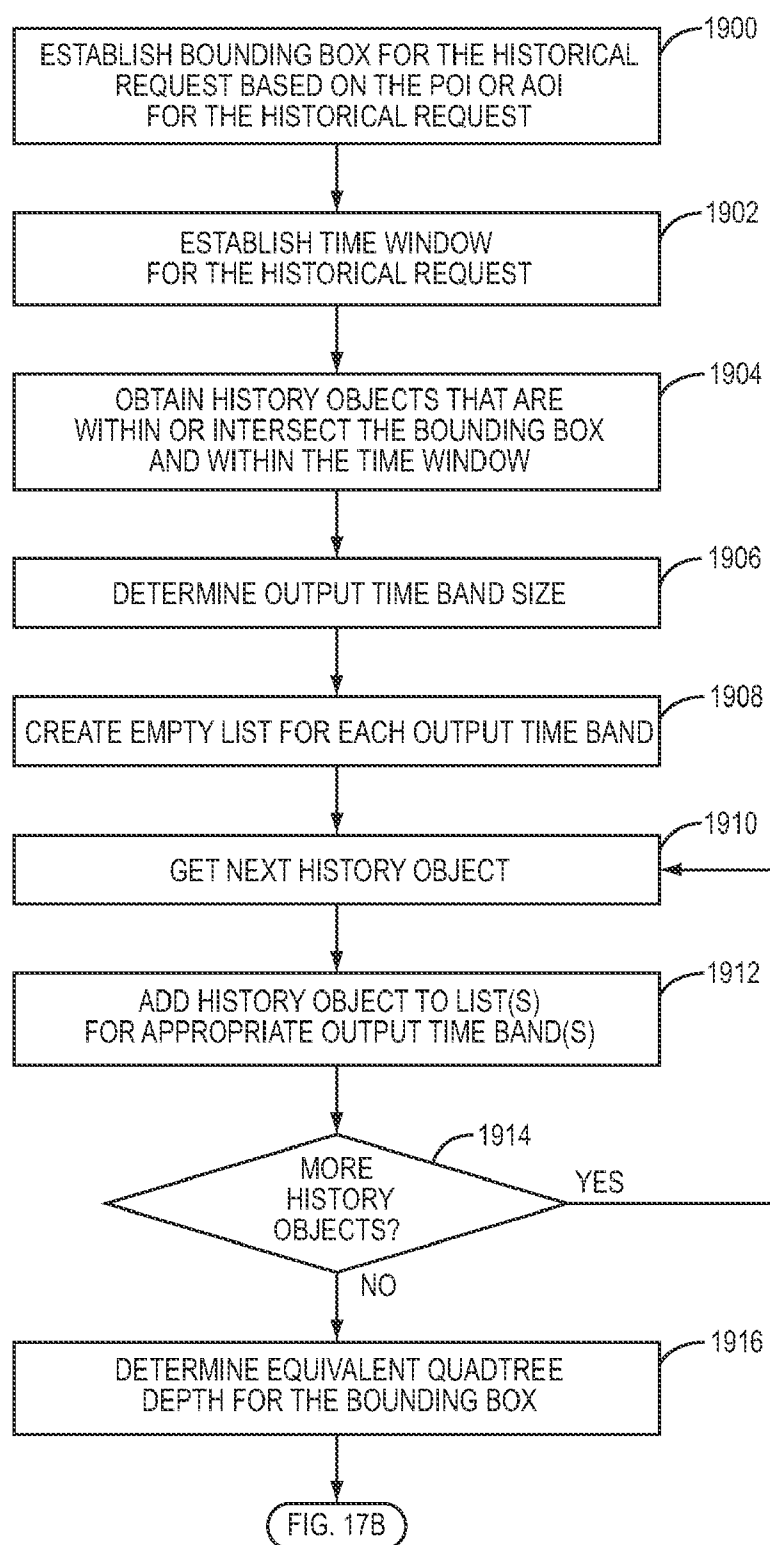
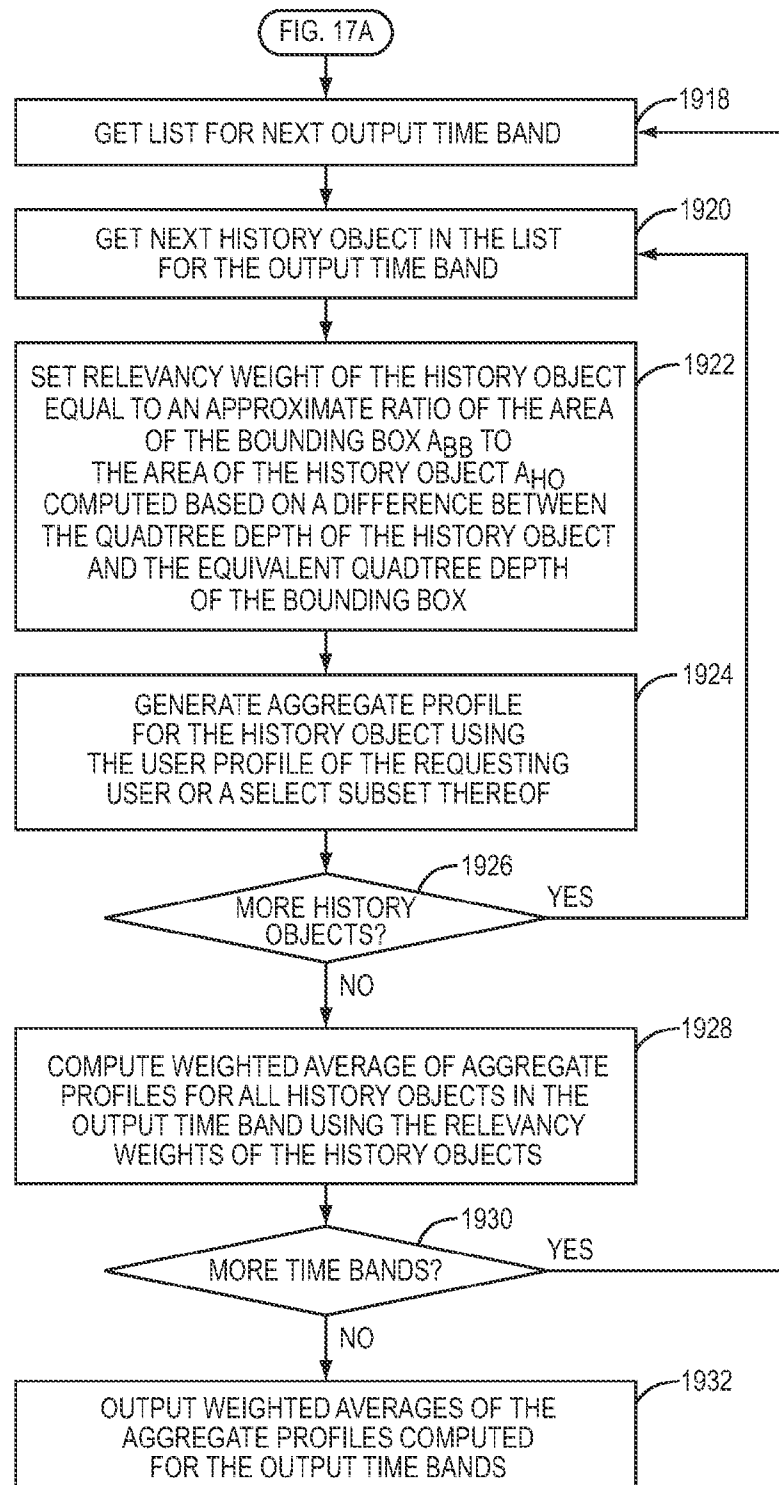


FIG. 16A

**FIG. 16B**

**FIG. 17A**

**FIG. 17B**

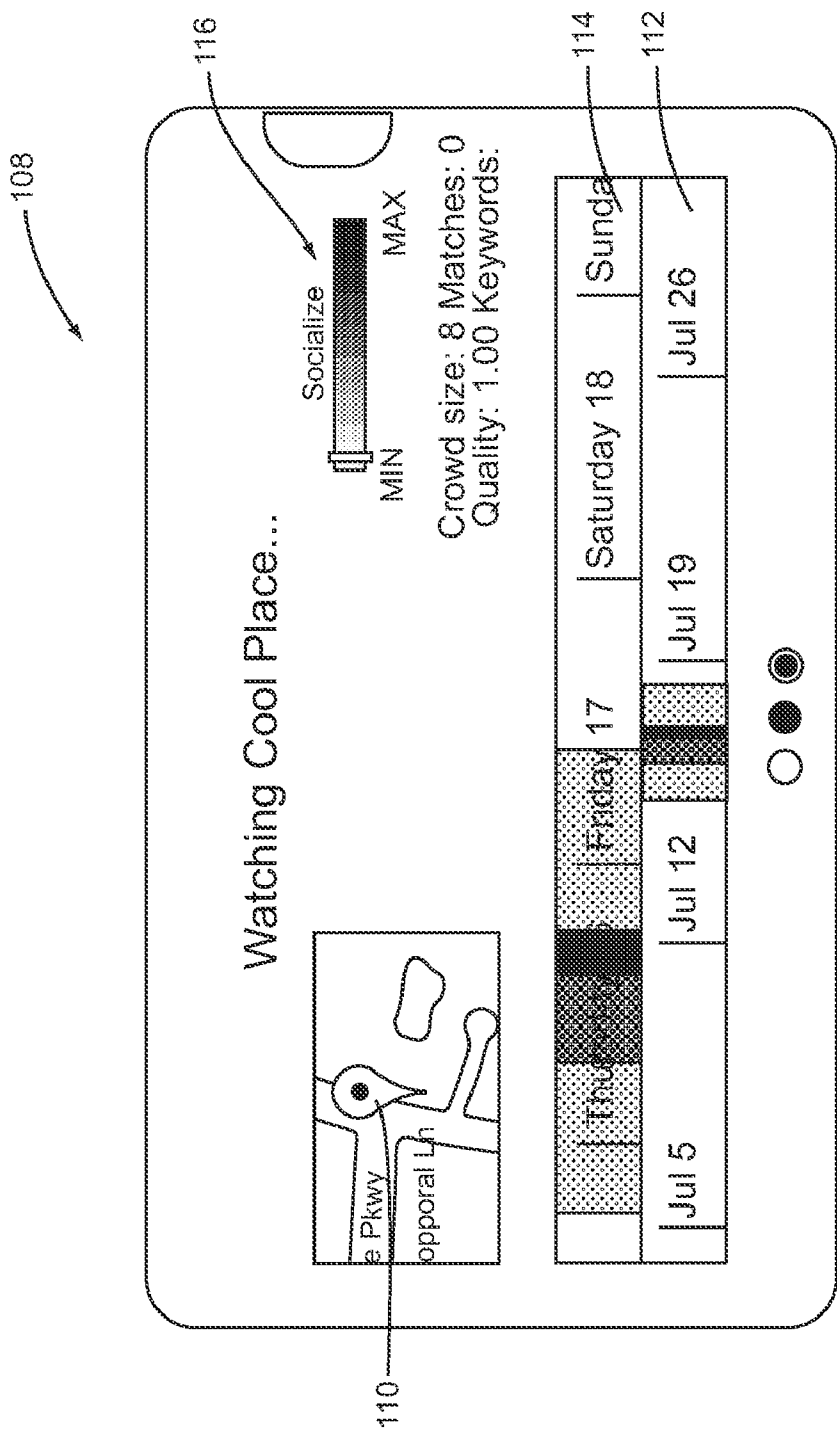
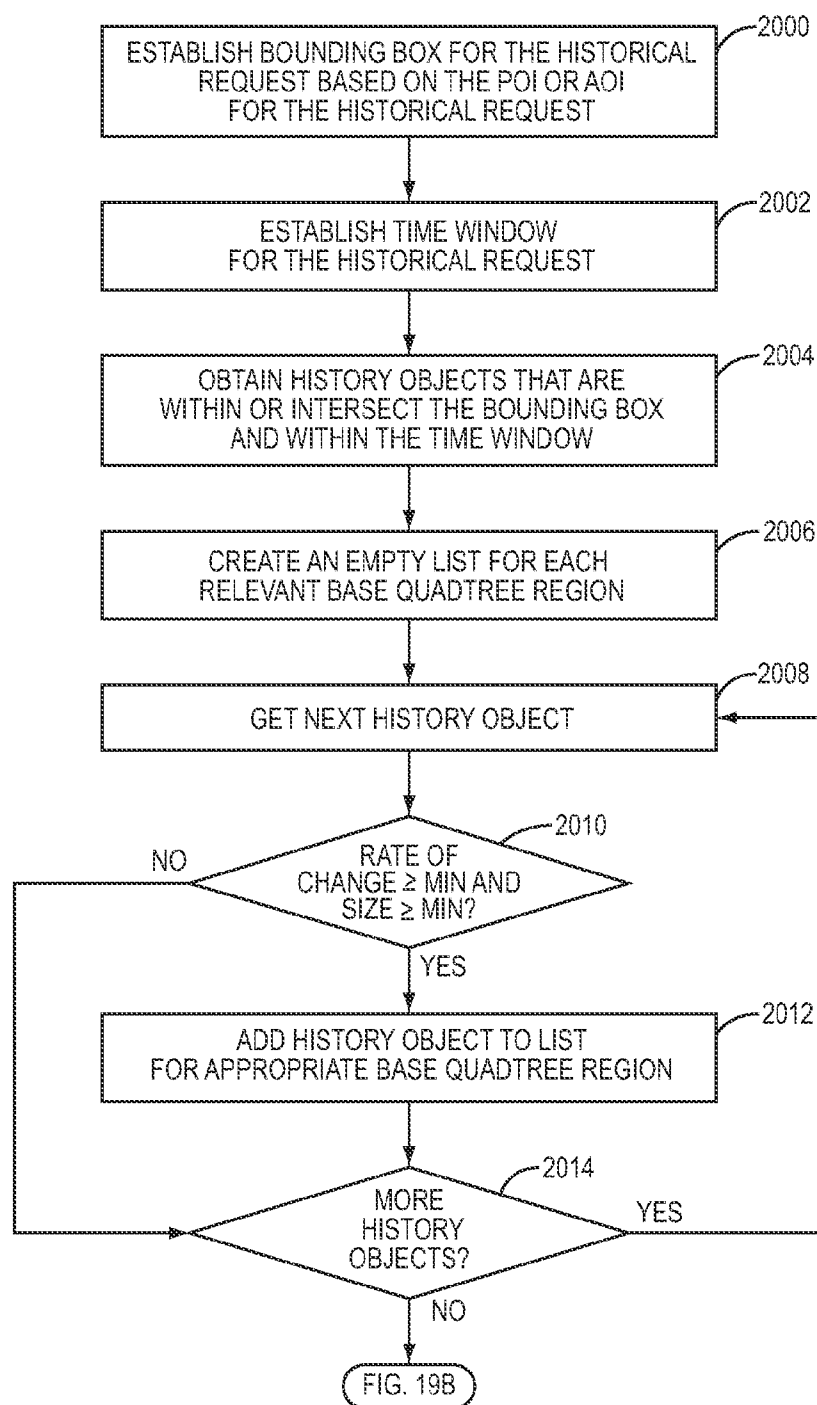


FIG. 18

**FIG. 19A**

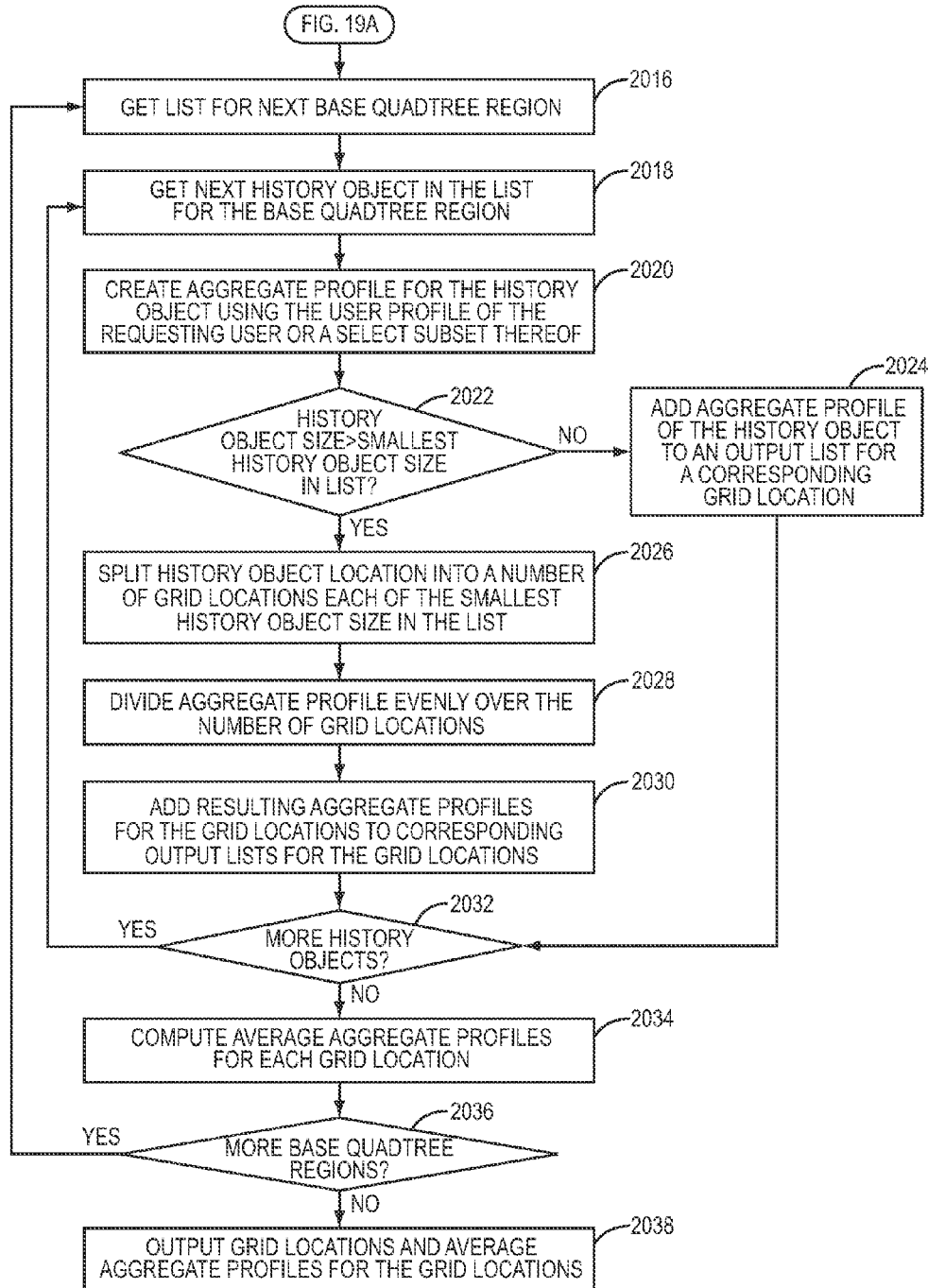
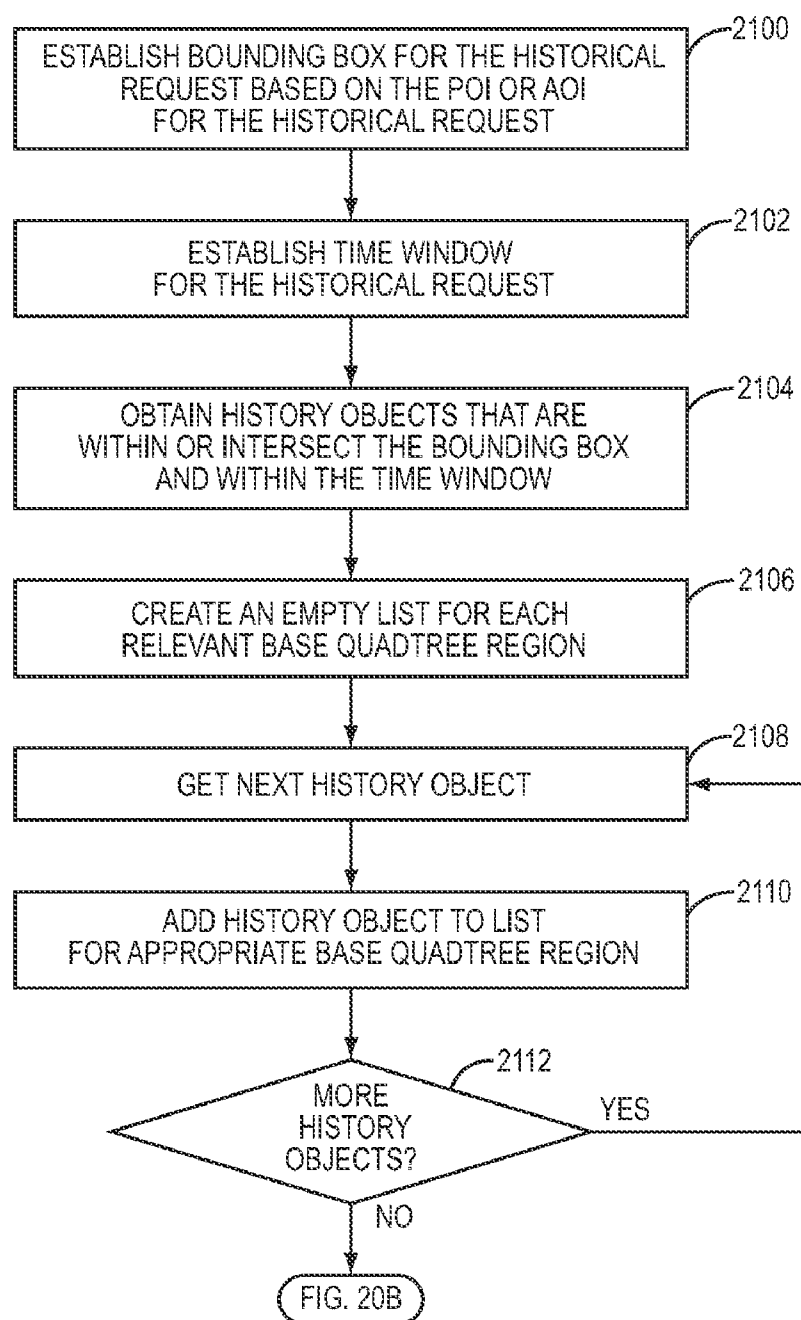


FIG. 19B

**FIG. 20A**

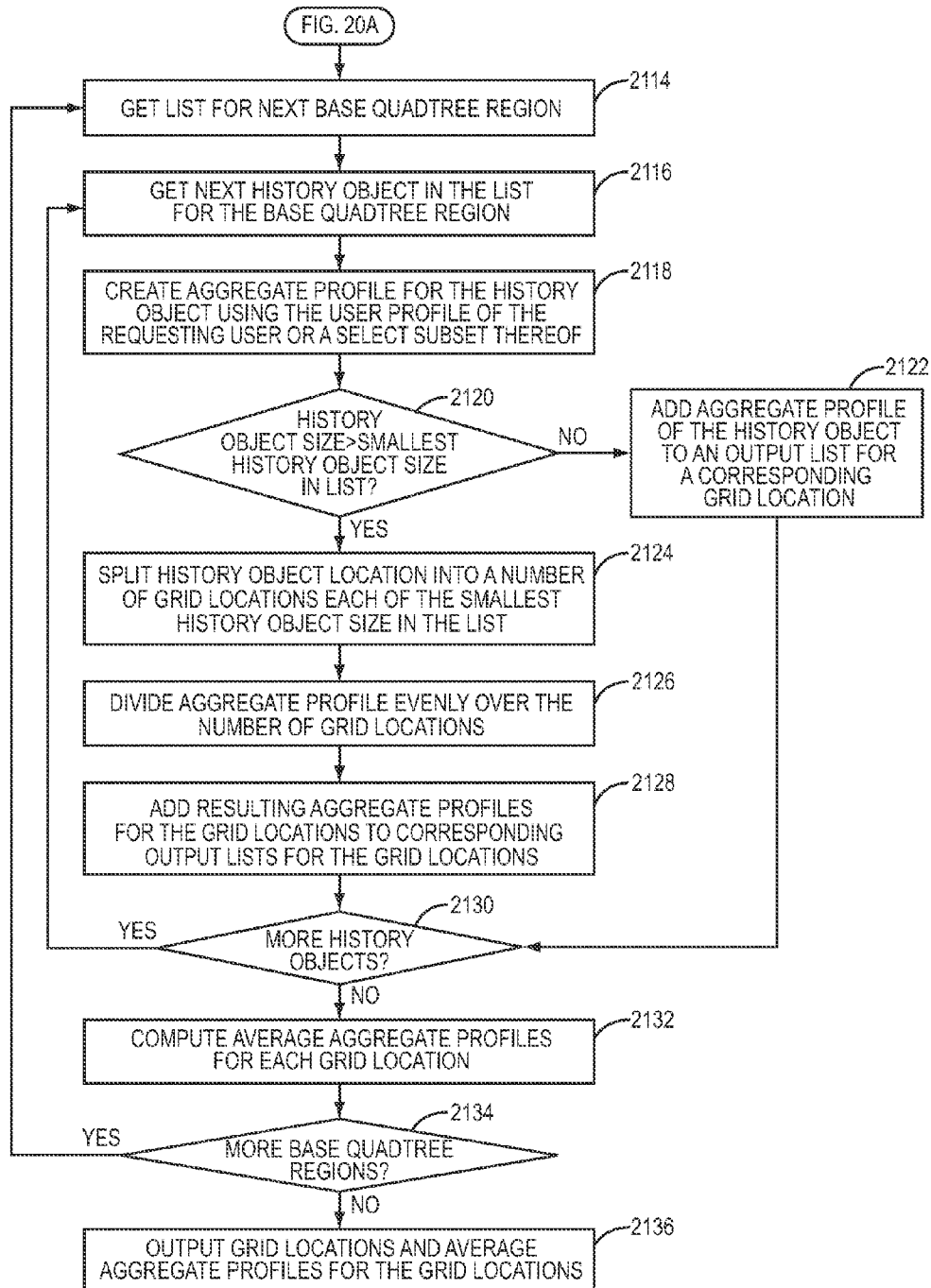


FIG. 20B

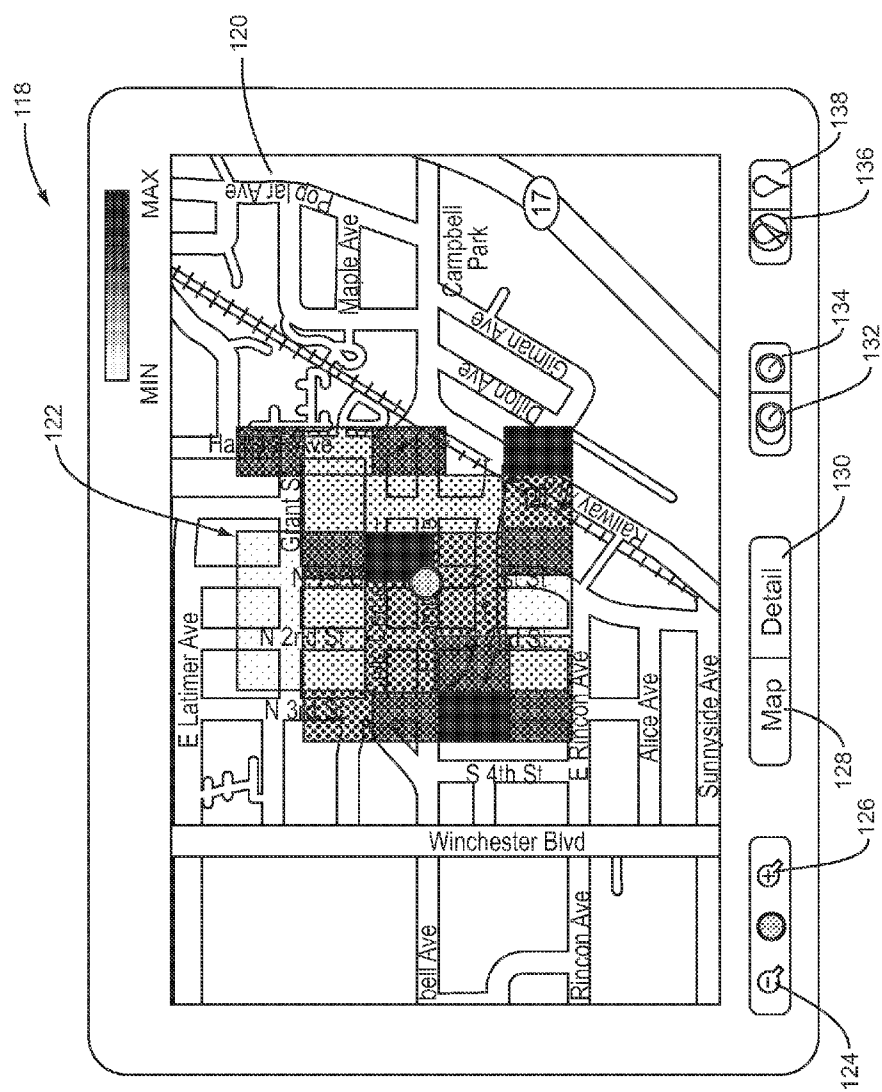


FIG. 21

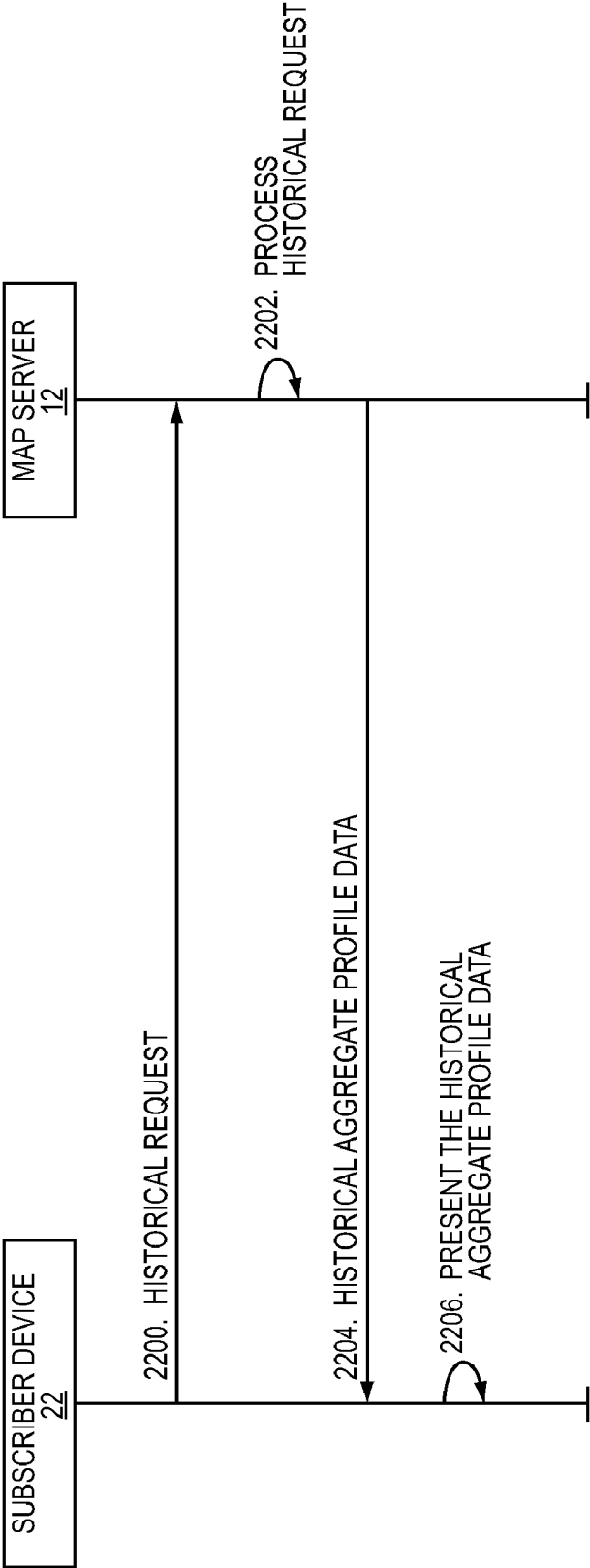
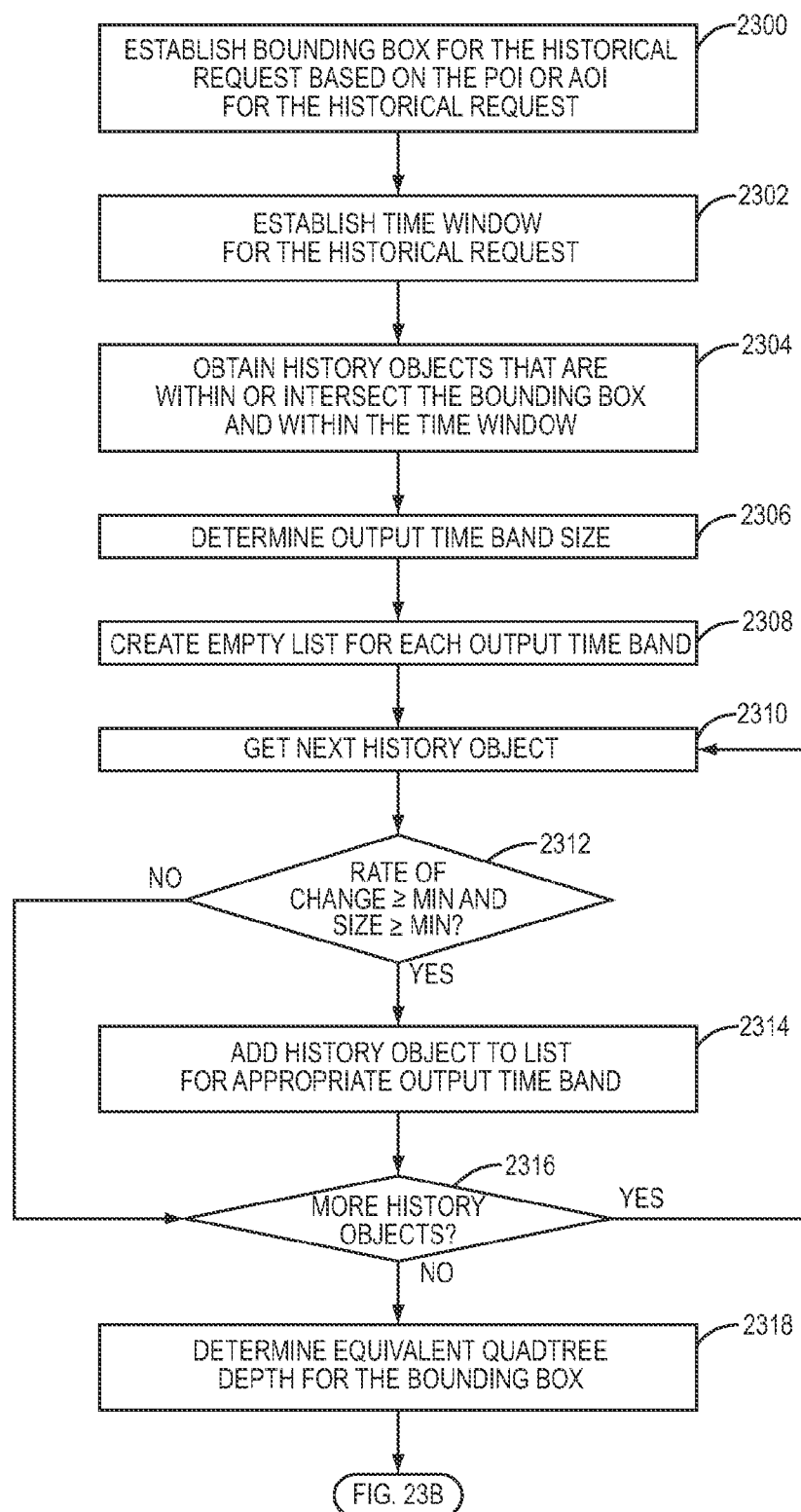
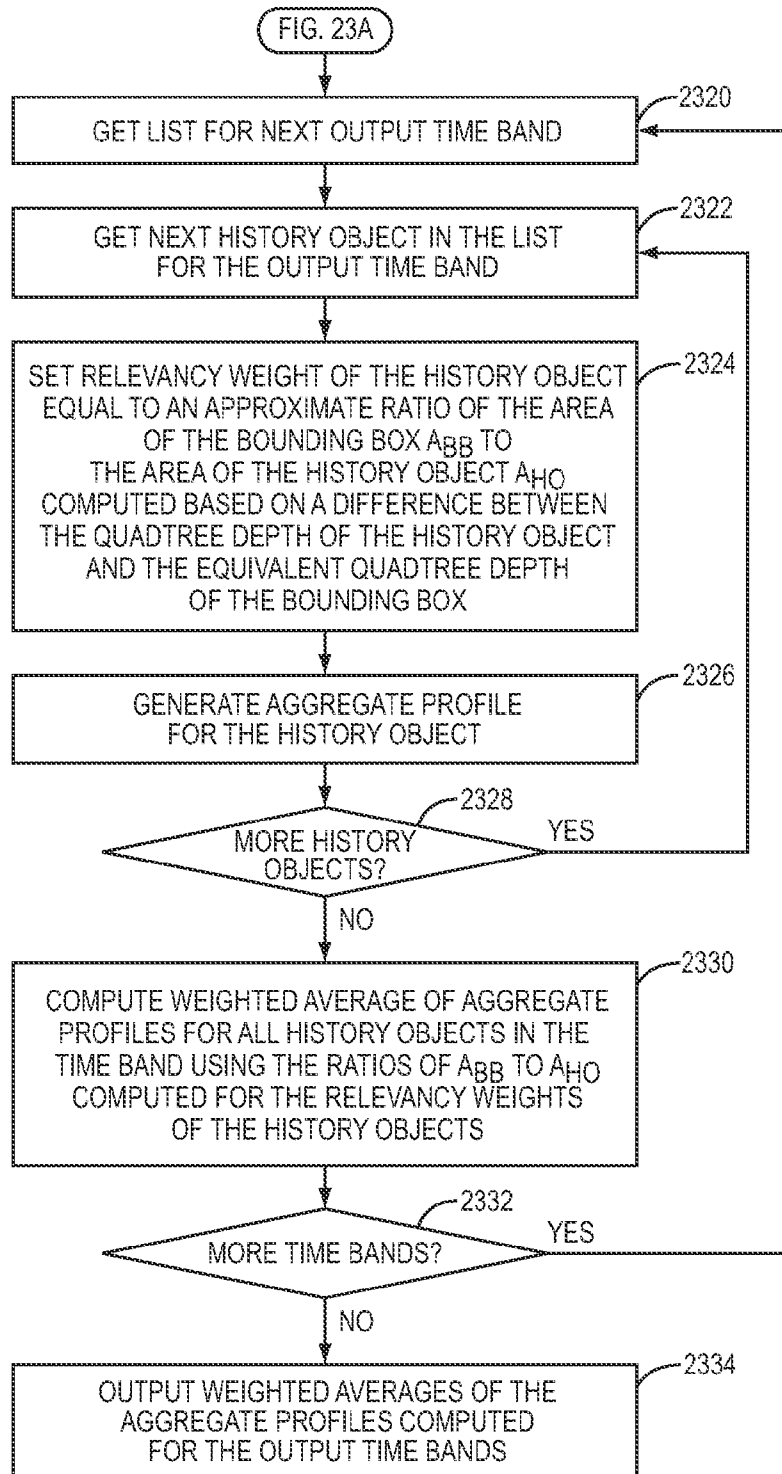
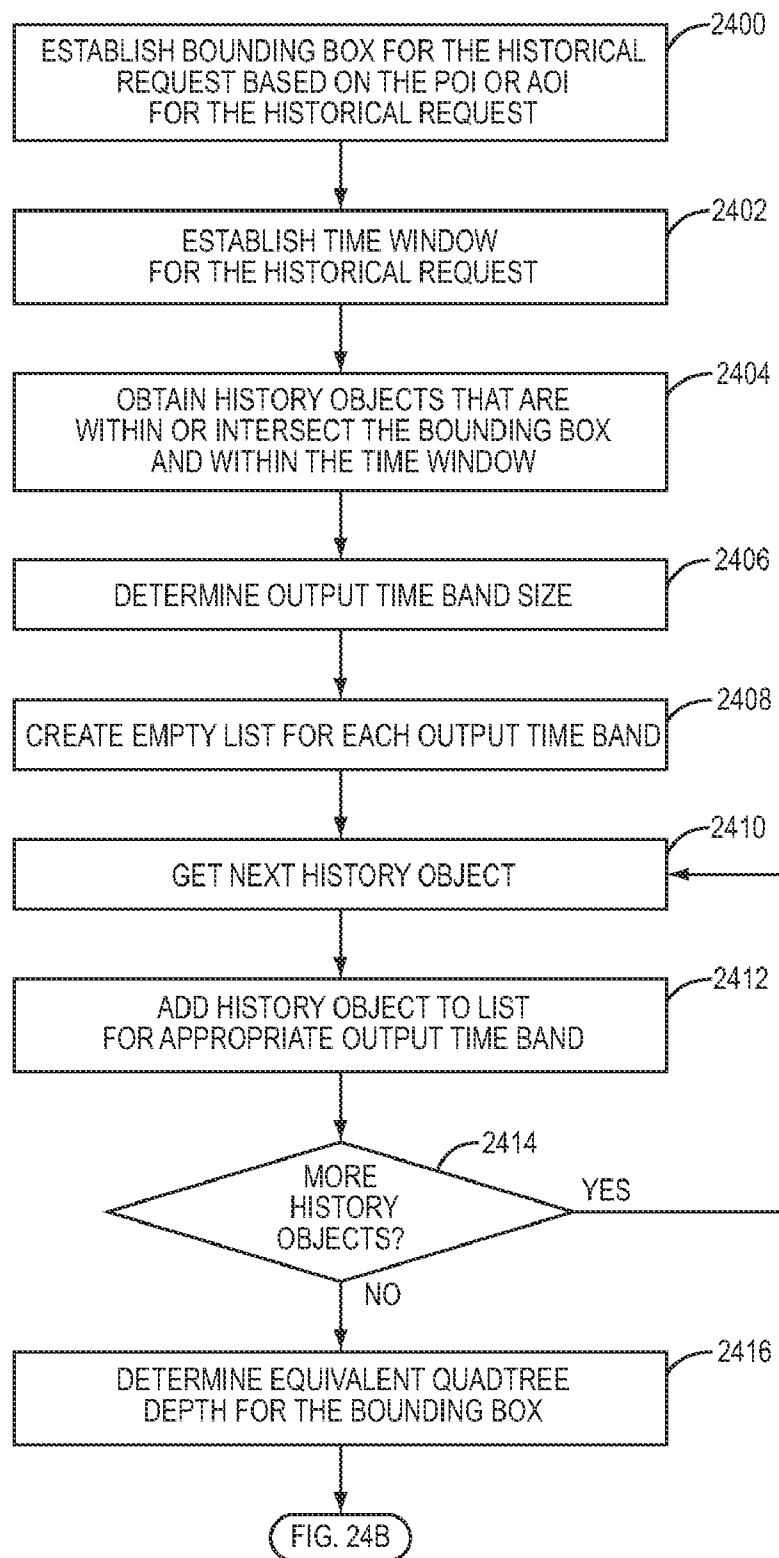
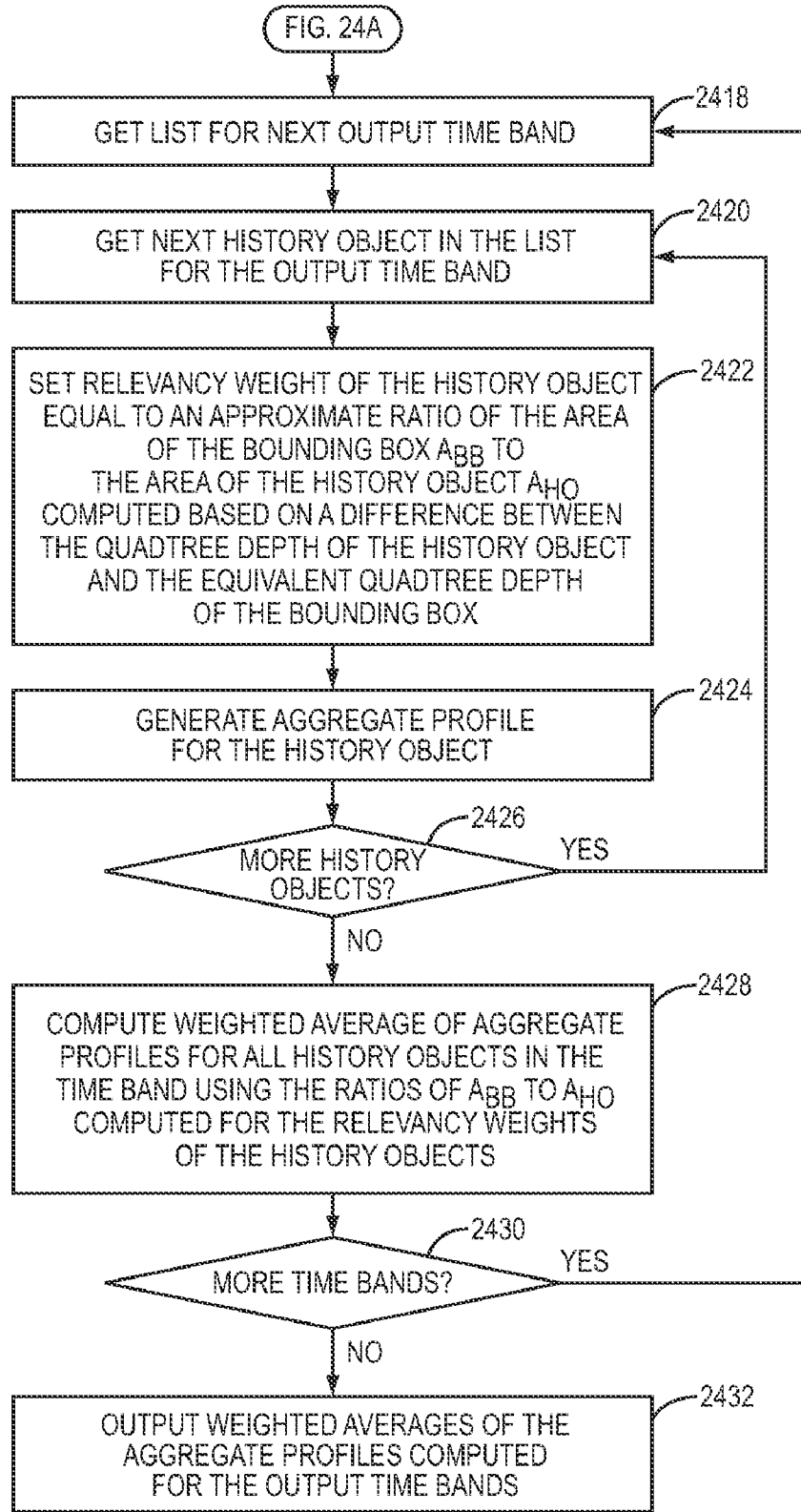


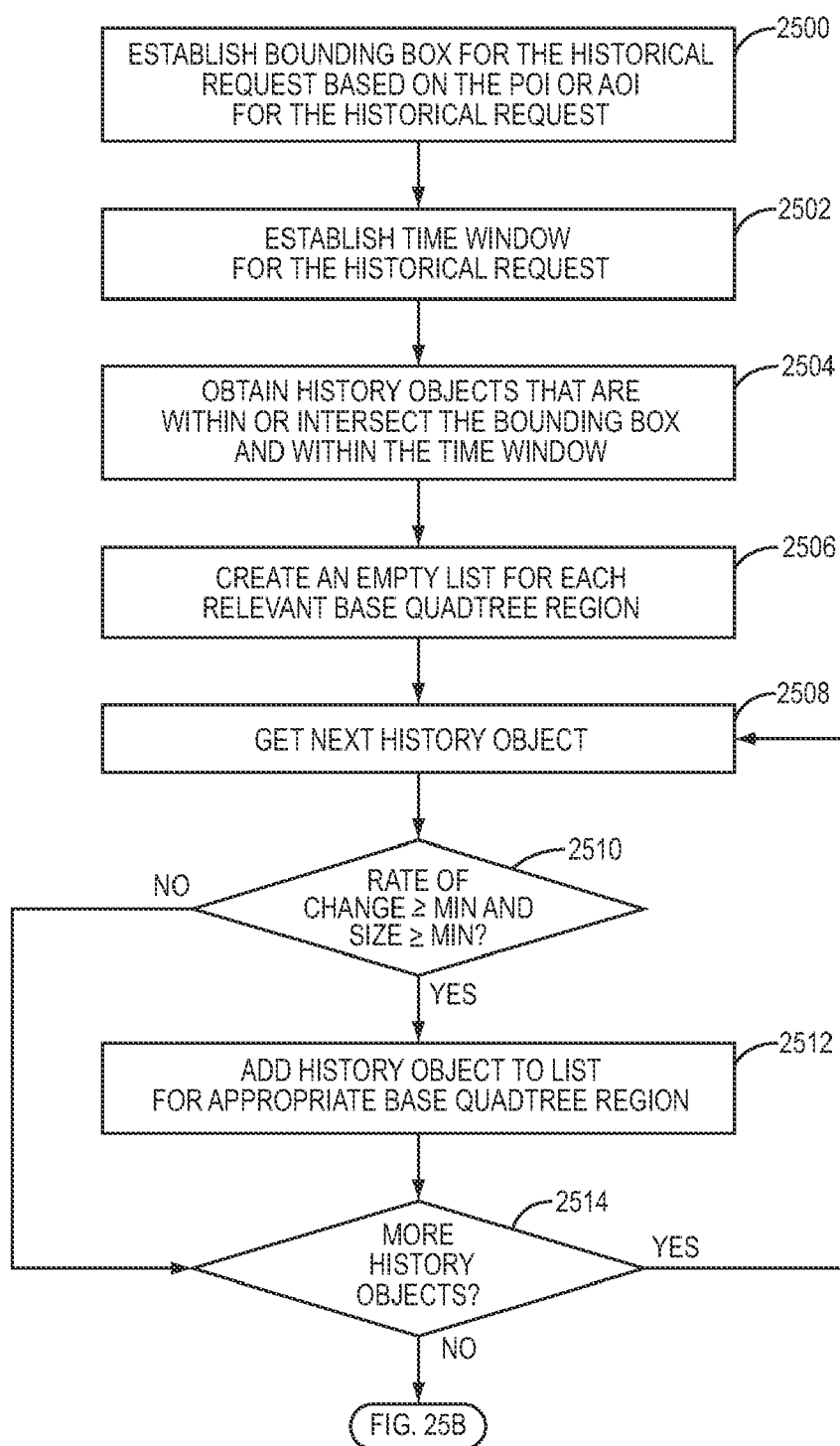
FIG. 22

**FIG. 23A**

**FIG. 23B**

**FIG. 24A**

**FIG. 24B**

**FIG. 25A**

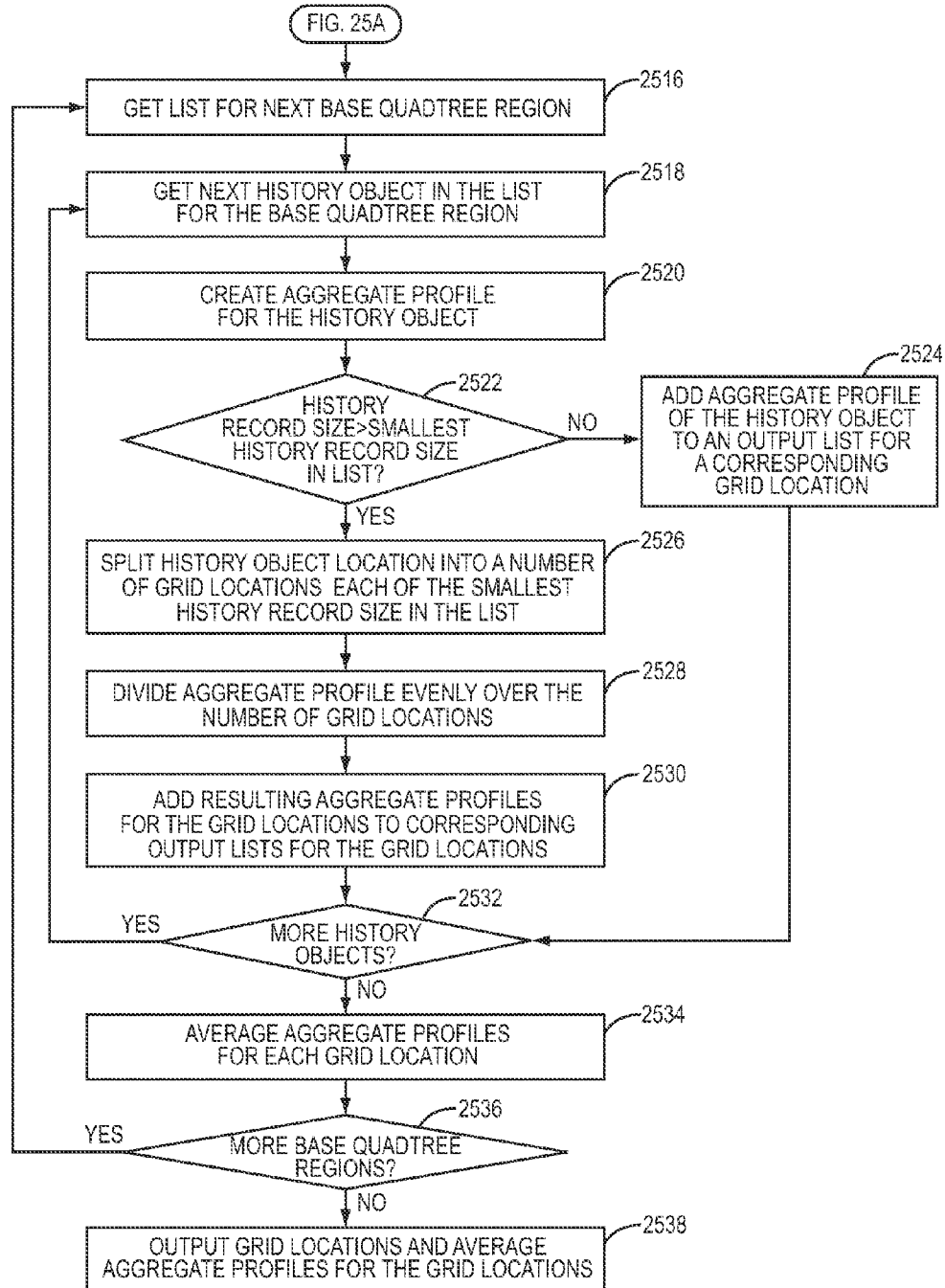
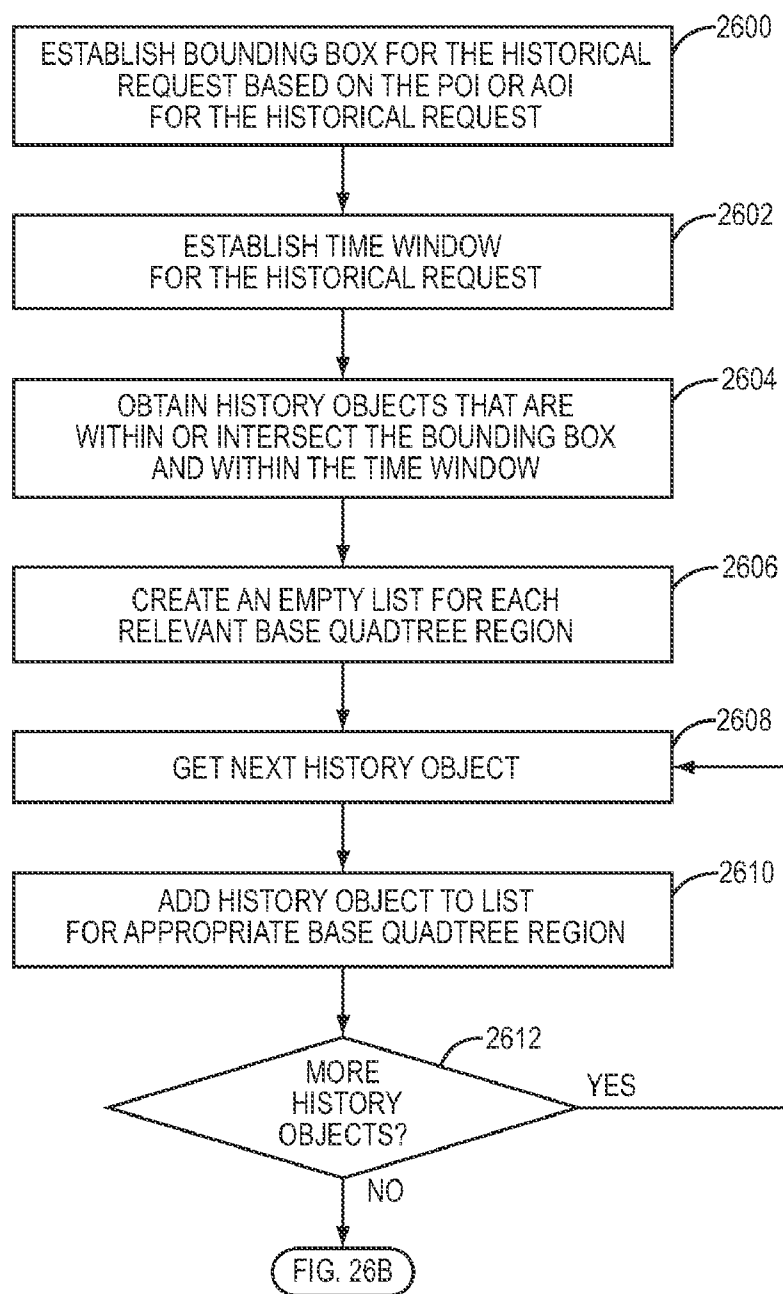


FIG. 25B

**FIG. 26A**

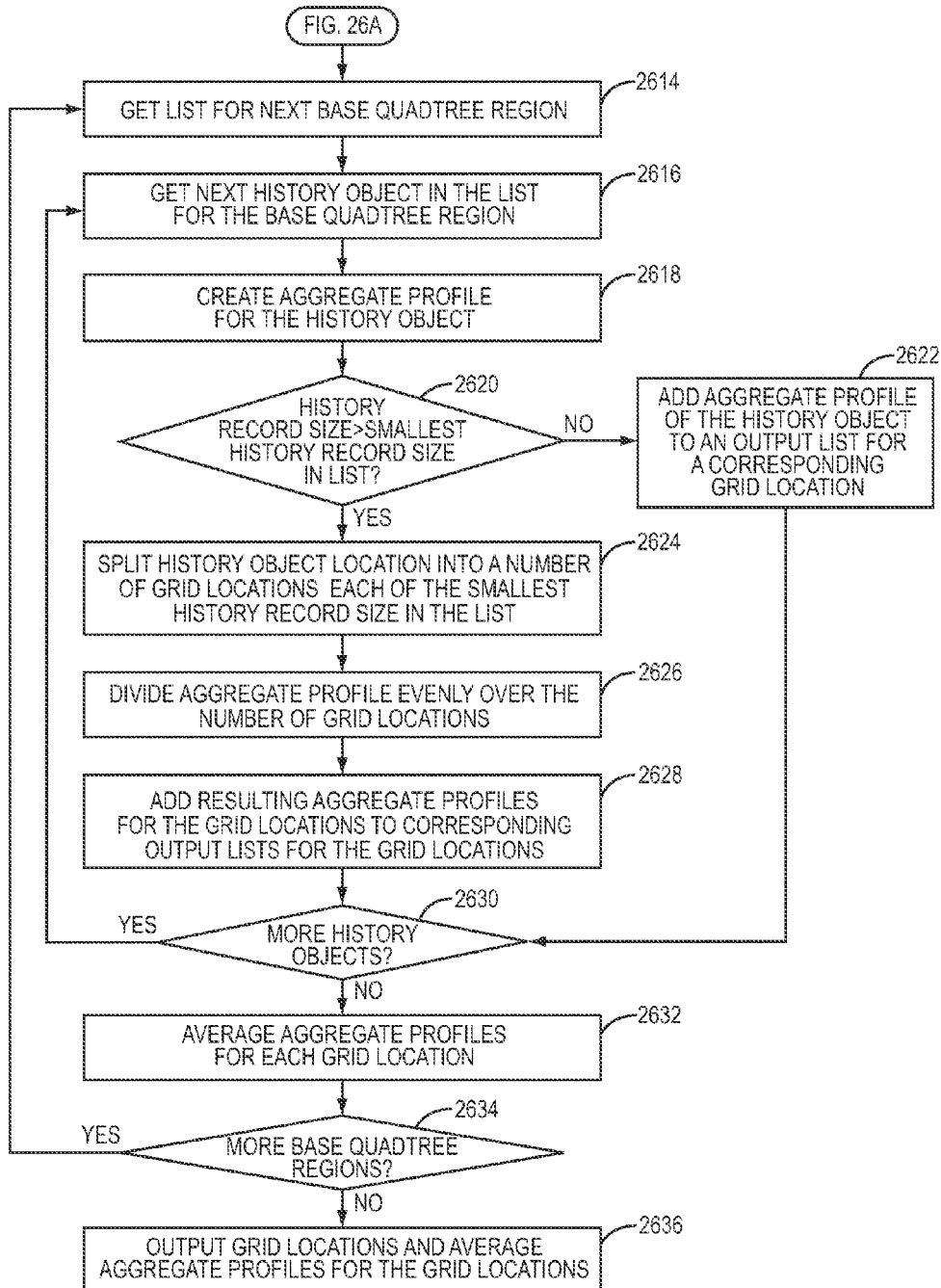


FIG. 26B

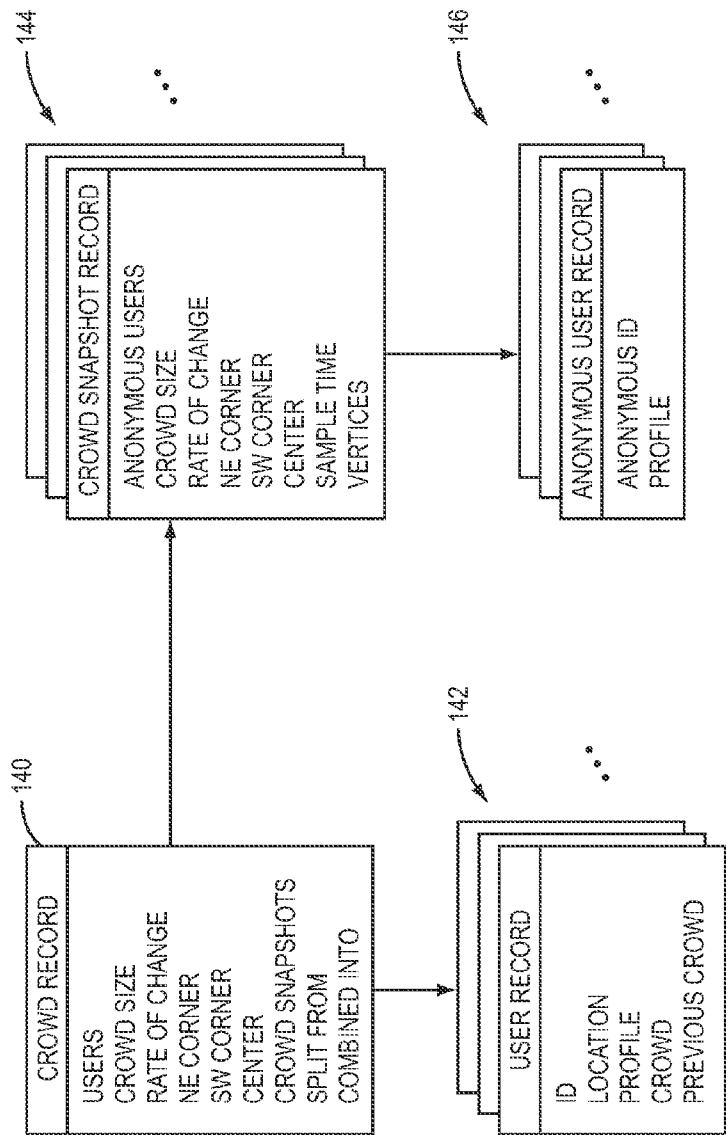


FIG. 27

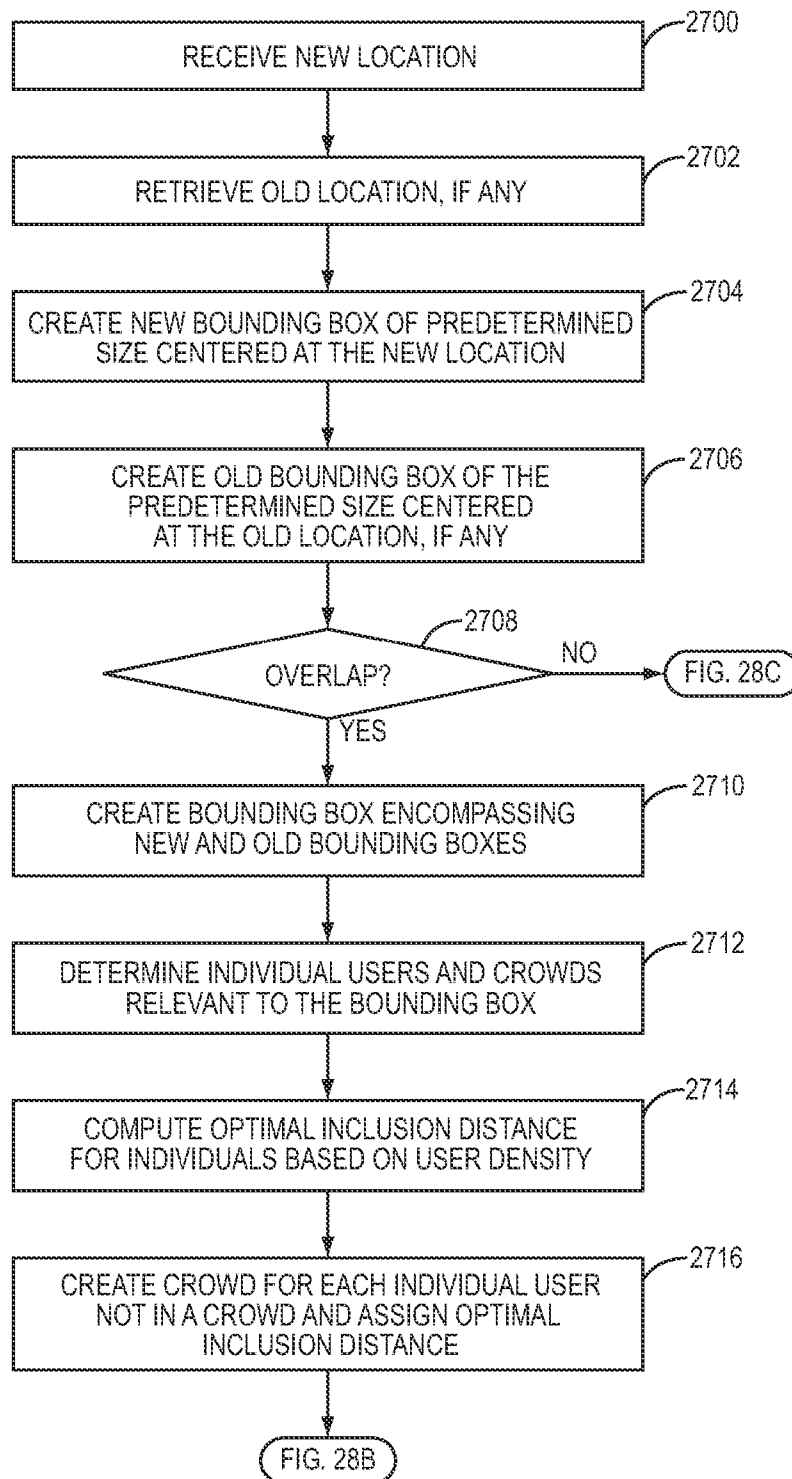


FIG. 28A

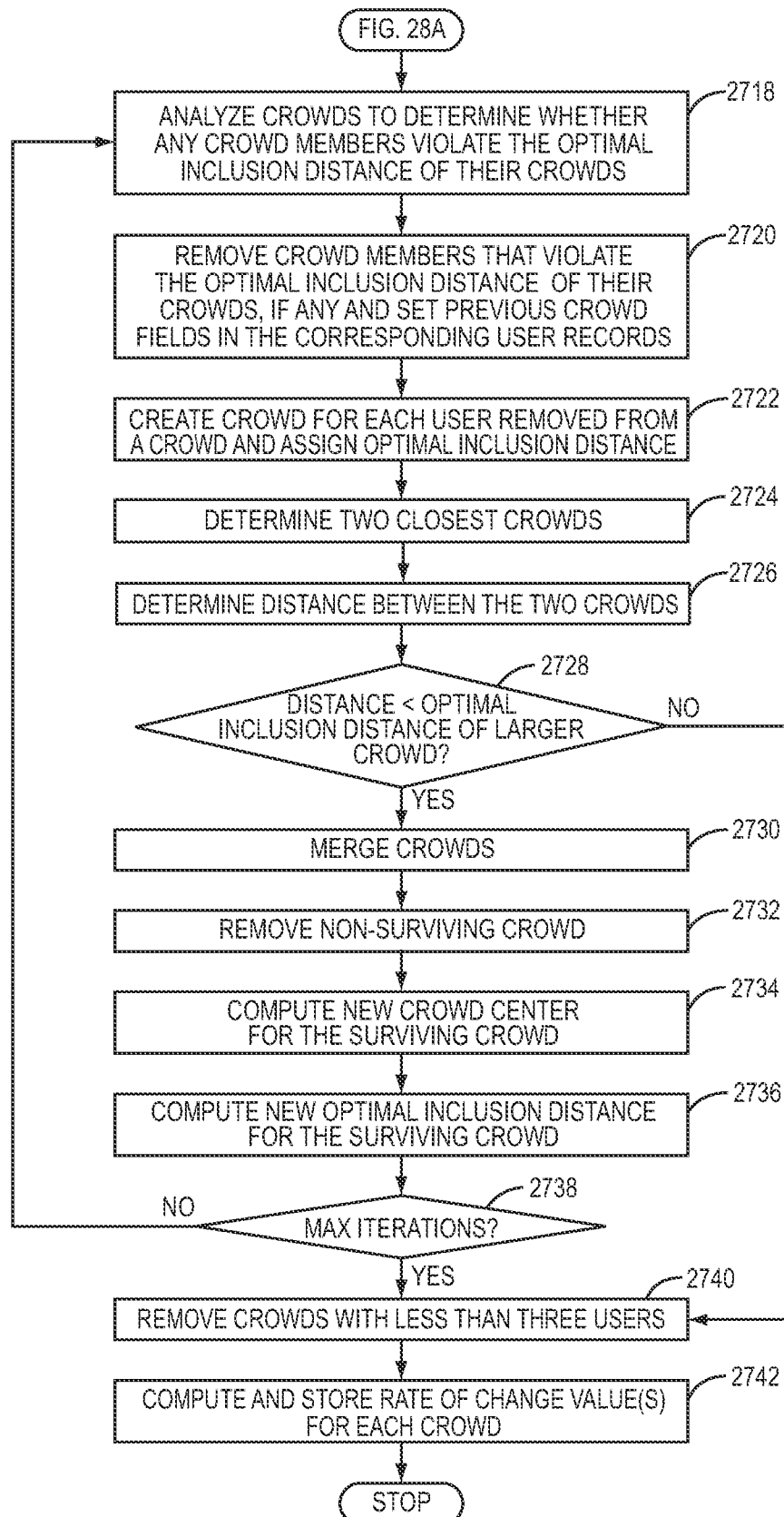


FIG. 28B

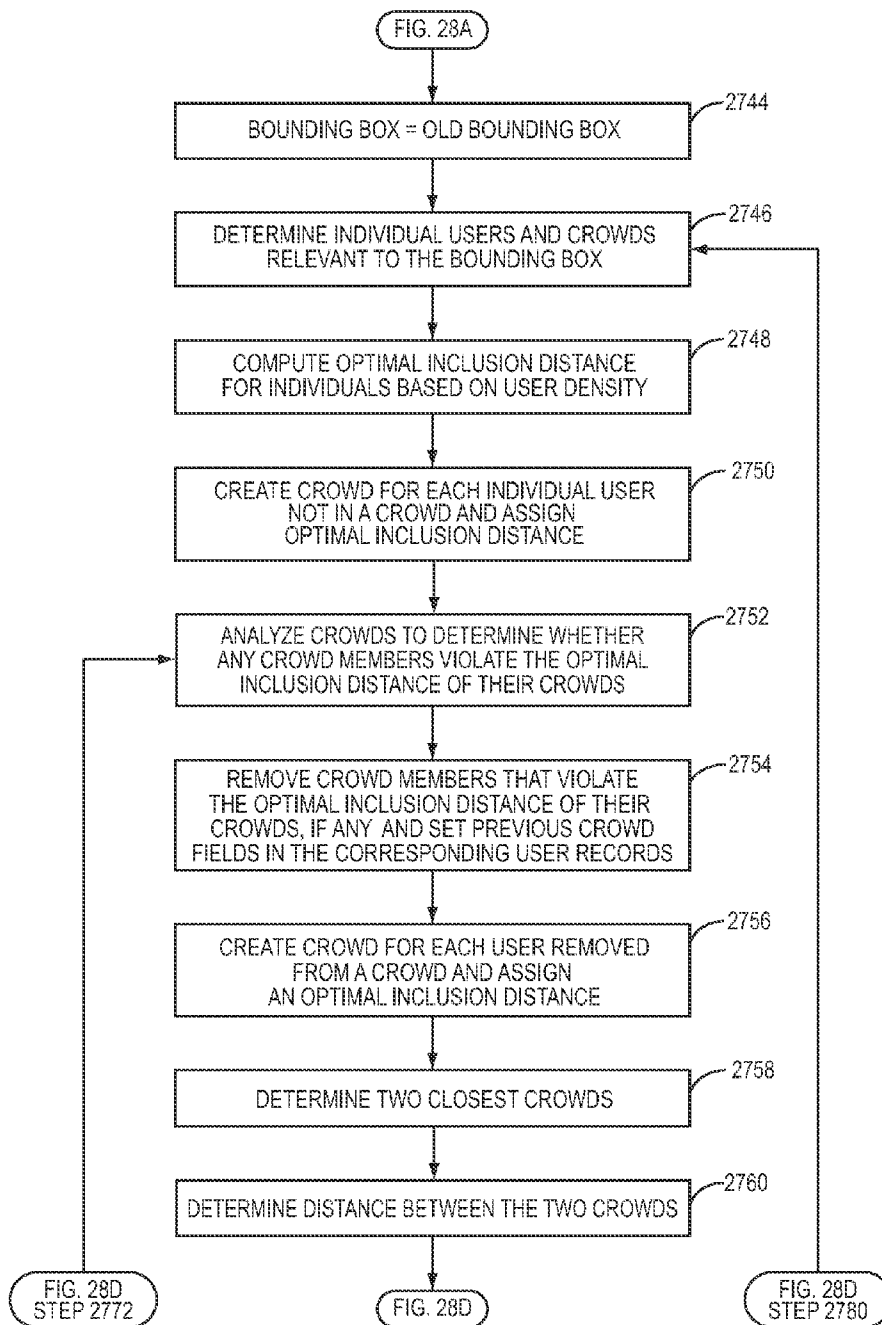


FIG. 28C

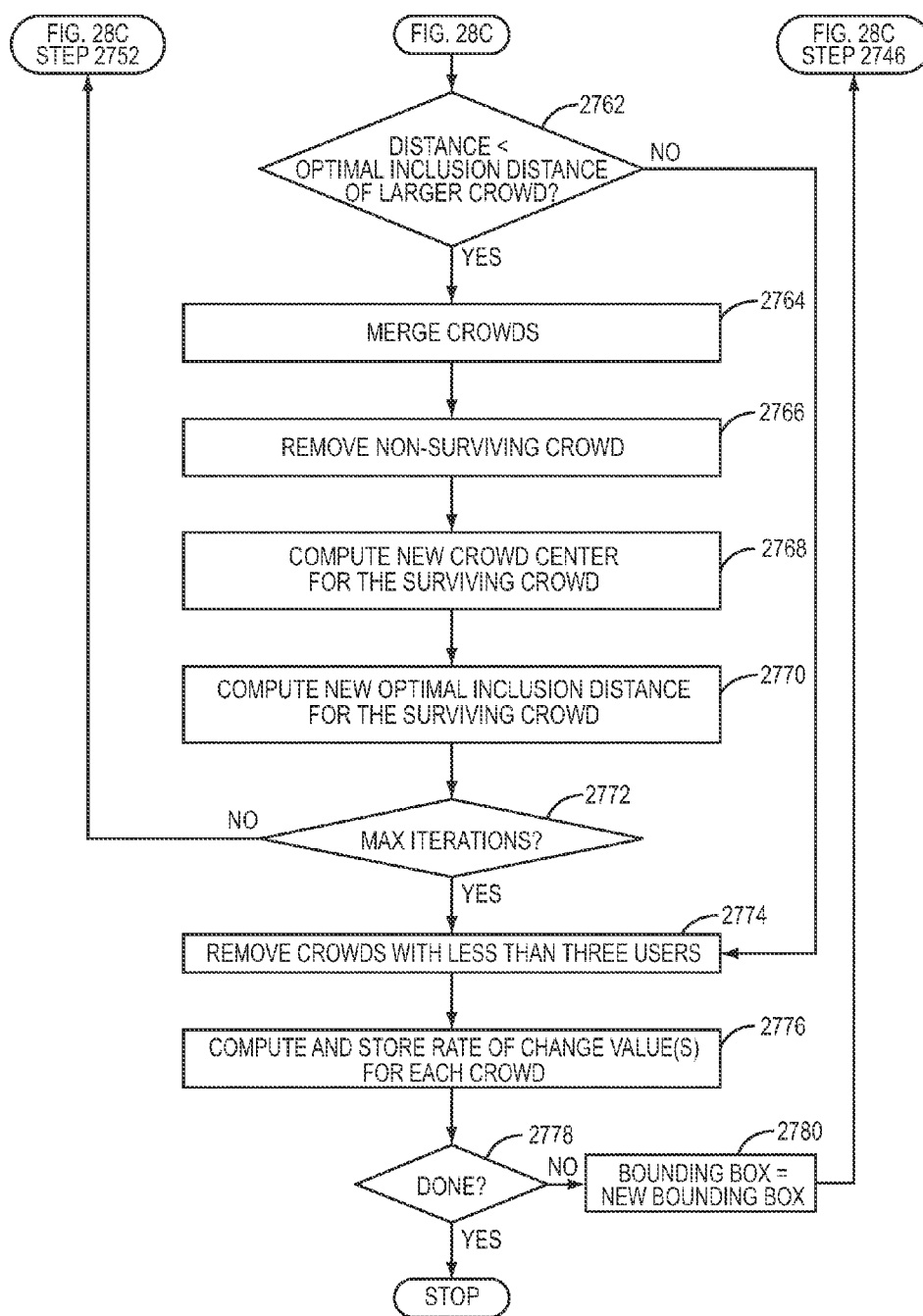


FIG. 28D

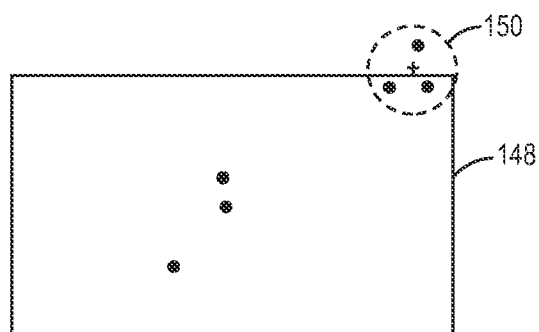


FIG. 29A

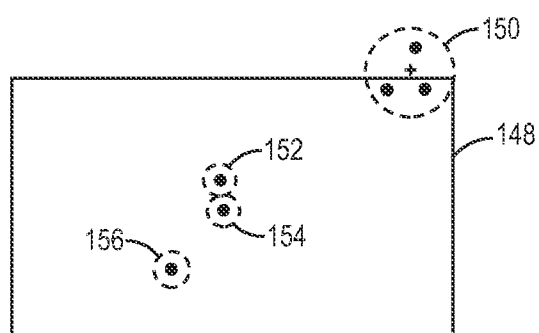


FIG. 29B

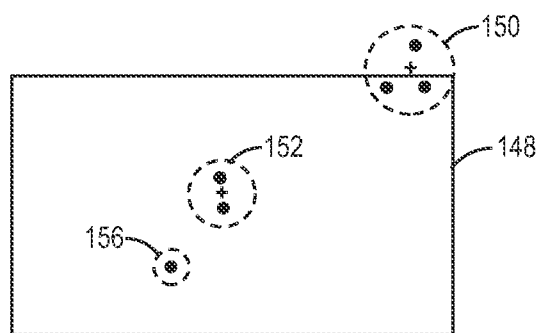


FIG. 29C

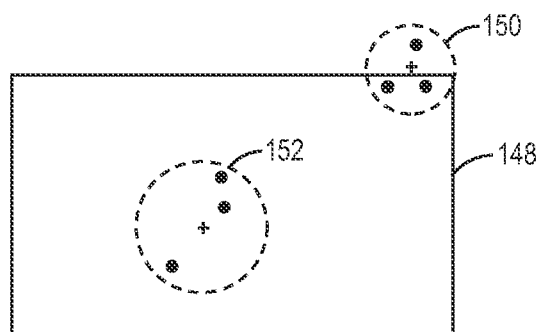


FIG. 29D

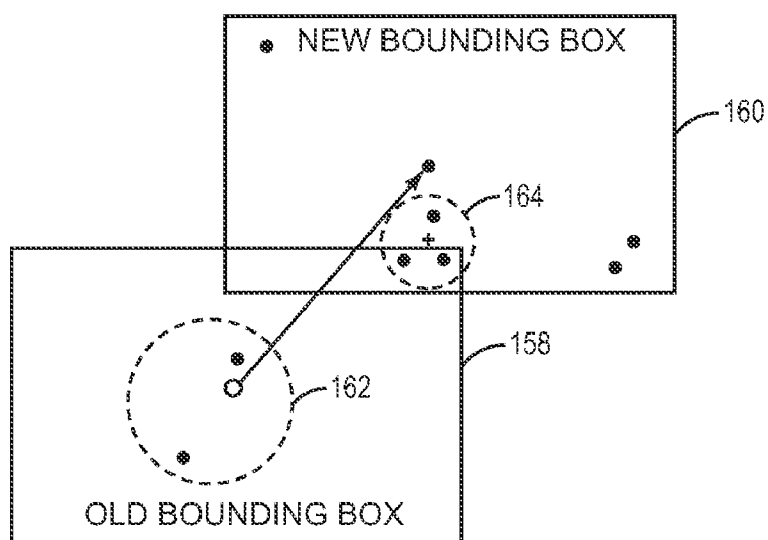


FIG. 30A

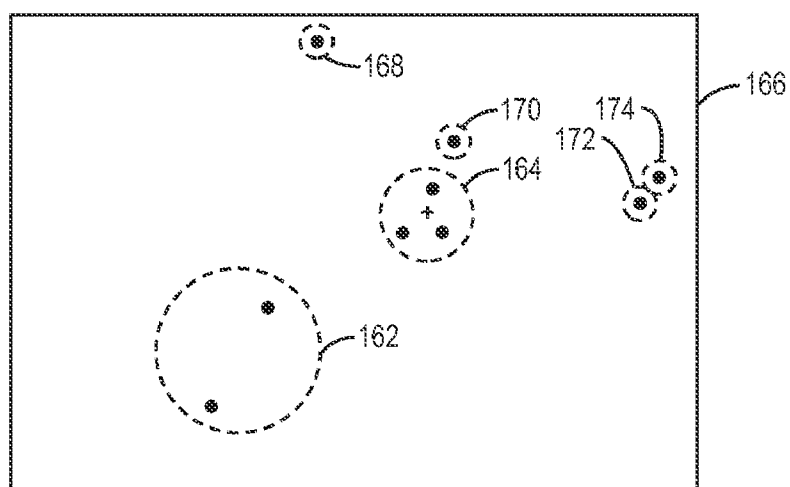


FIG. 30B

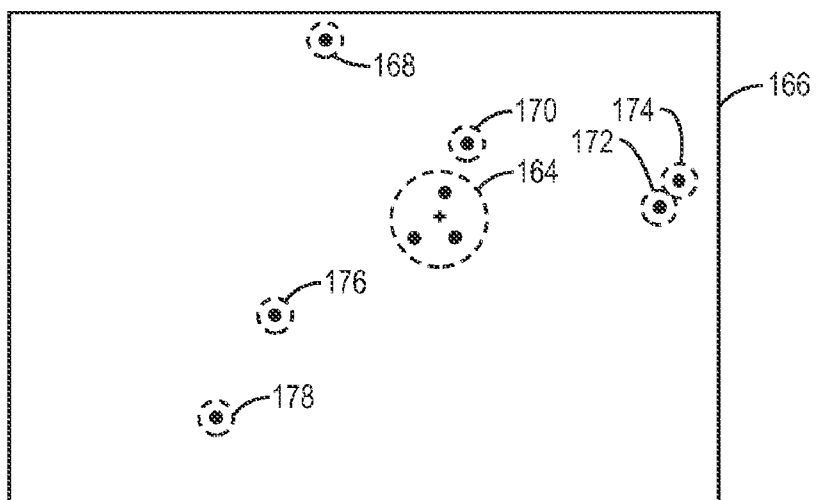


FIG. 30C

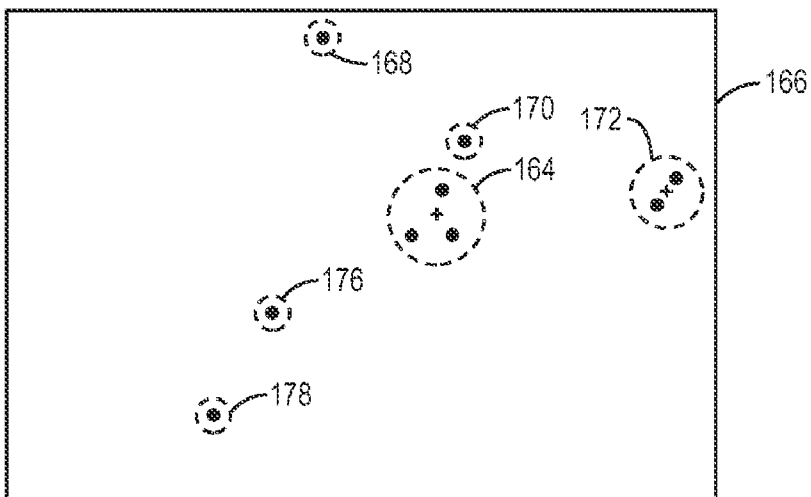


FIG. 30D

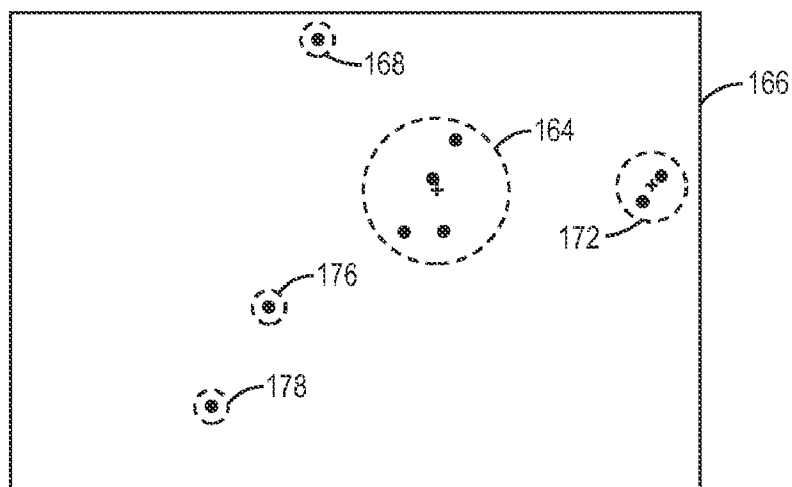


FIG. 30E

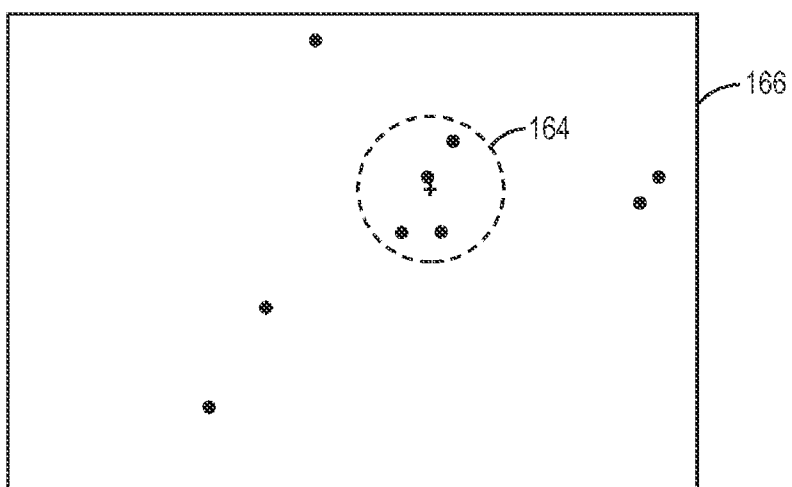


FIG. 30F

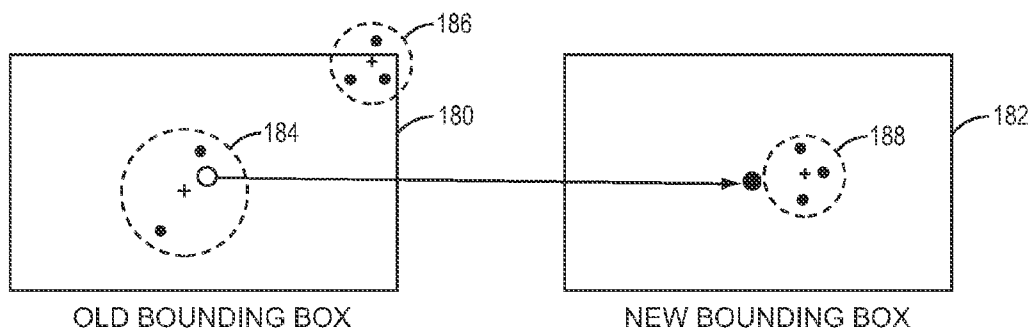


FIG. 31A

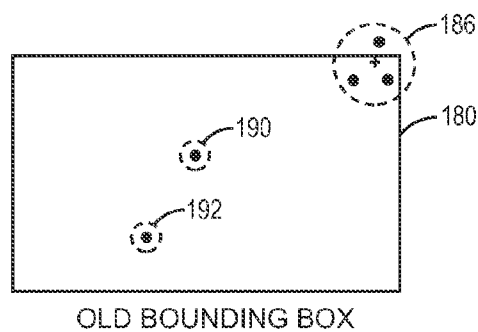


FIG. 31B

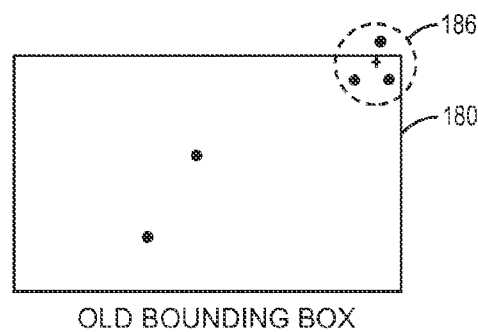
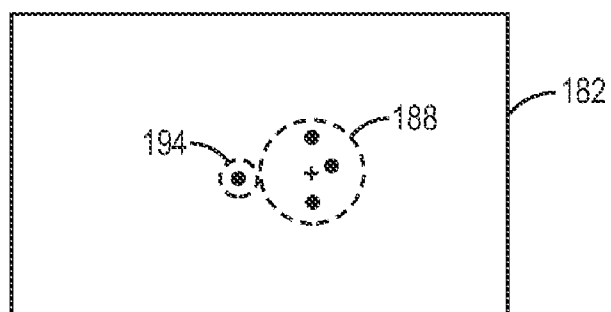
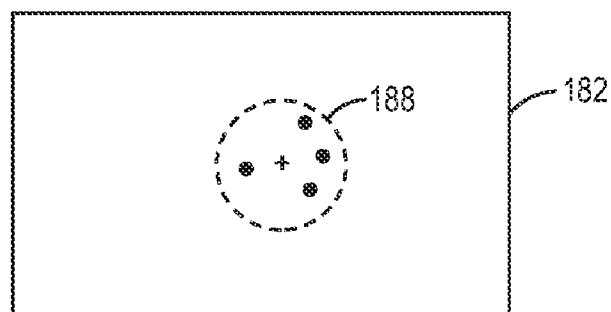


FIG. 31C



NEW BOUNDING BOX

FIG. 31D



NEW BOUNDING BOX

FIG. 31E

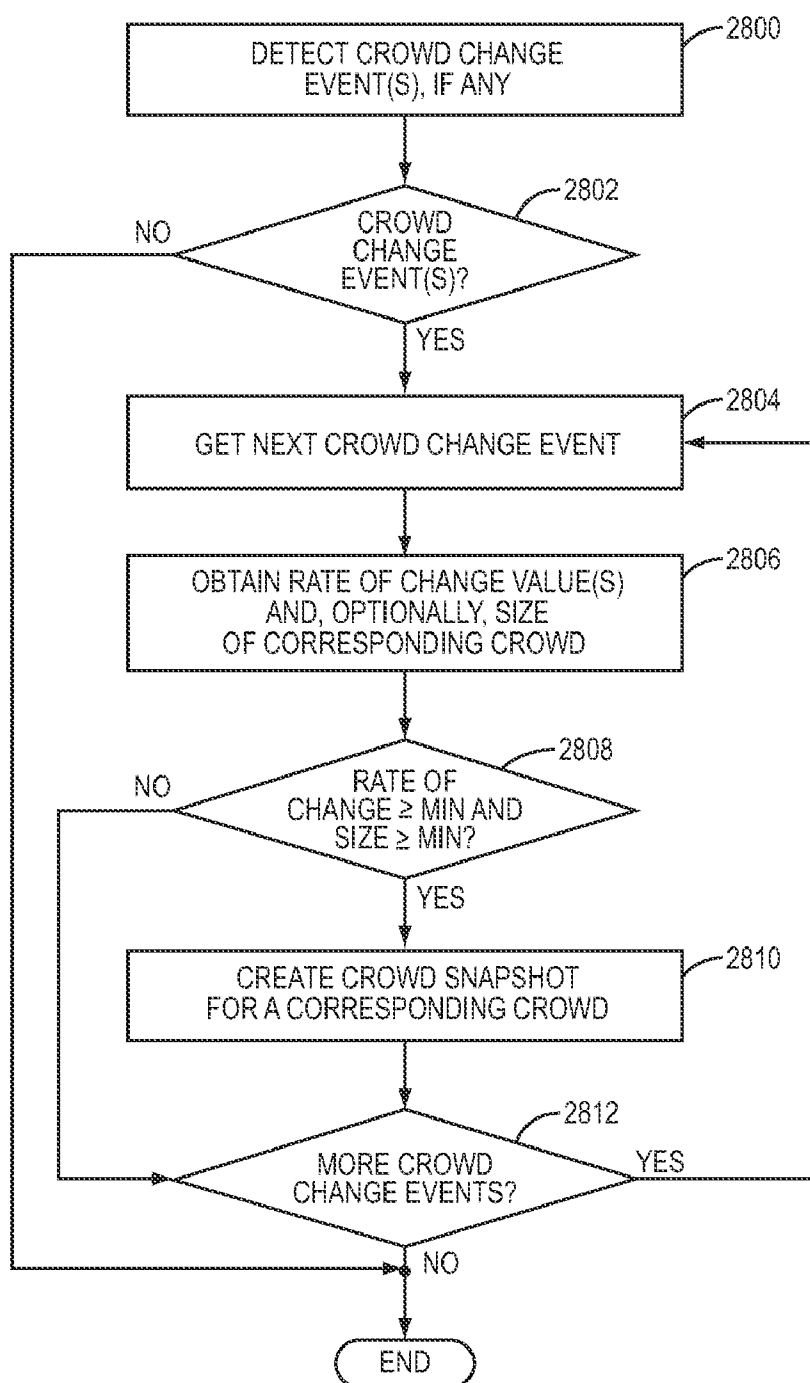
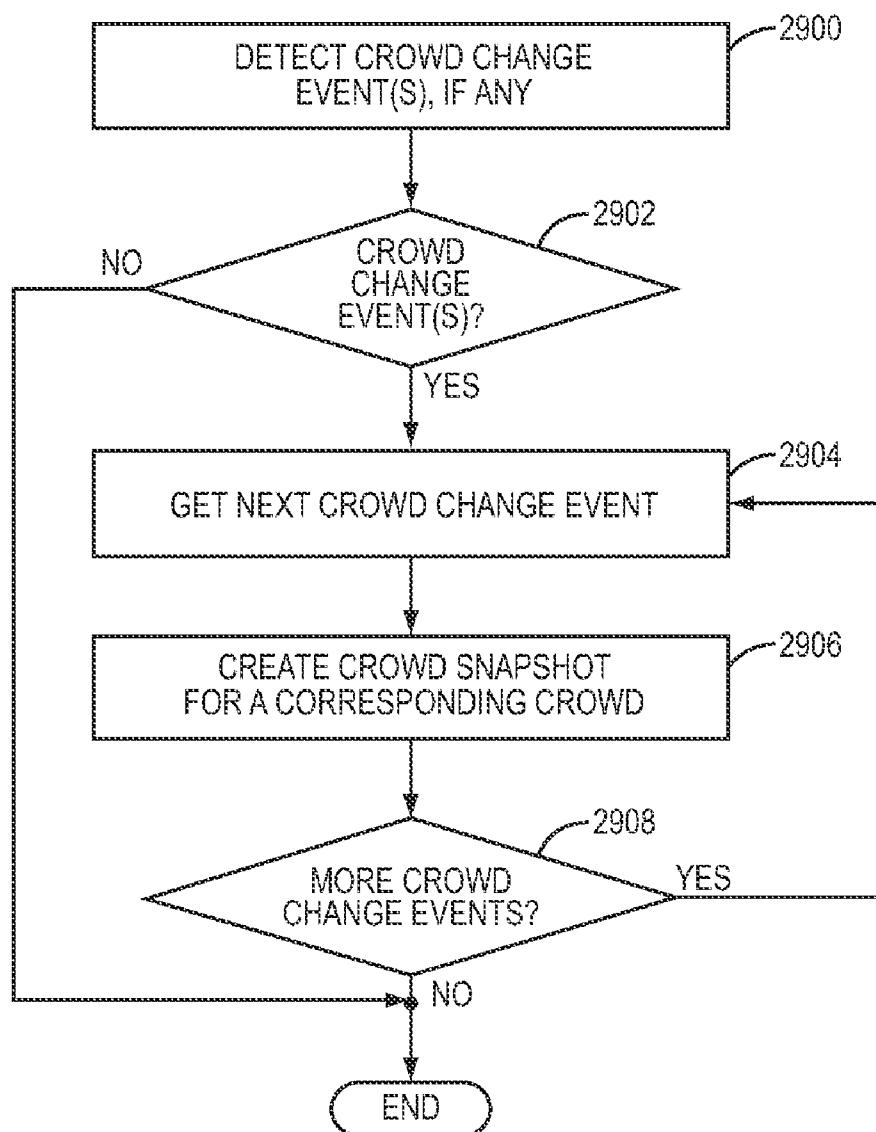


FIG. 32

**FIG. 33**

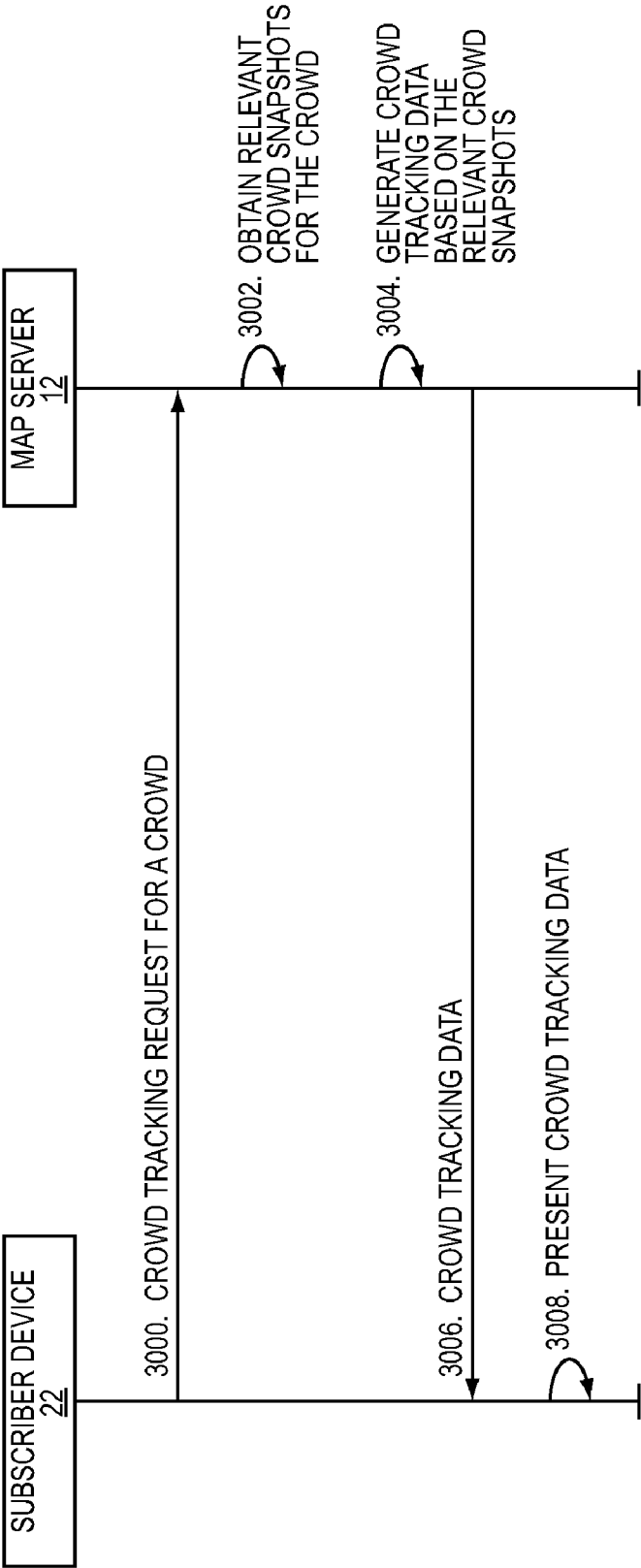


FIG. 34

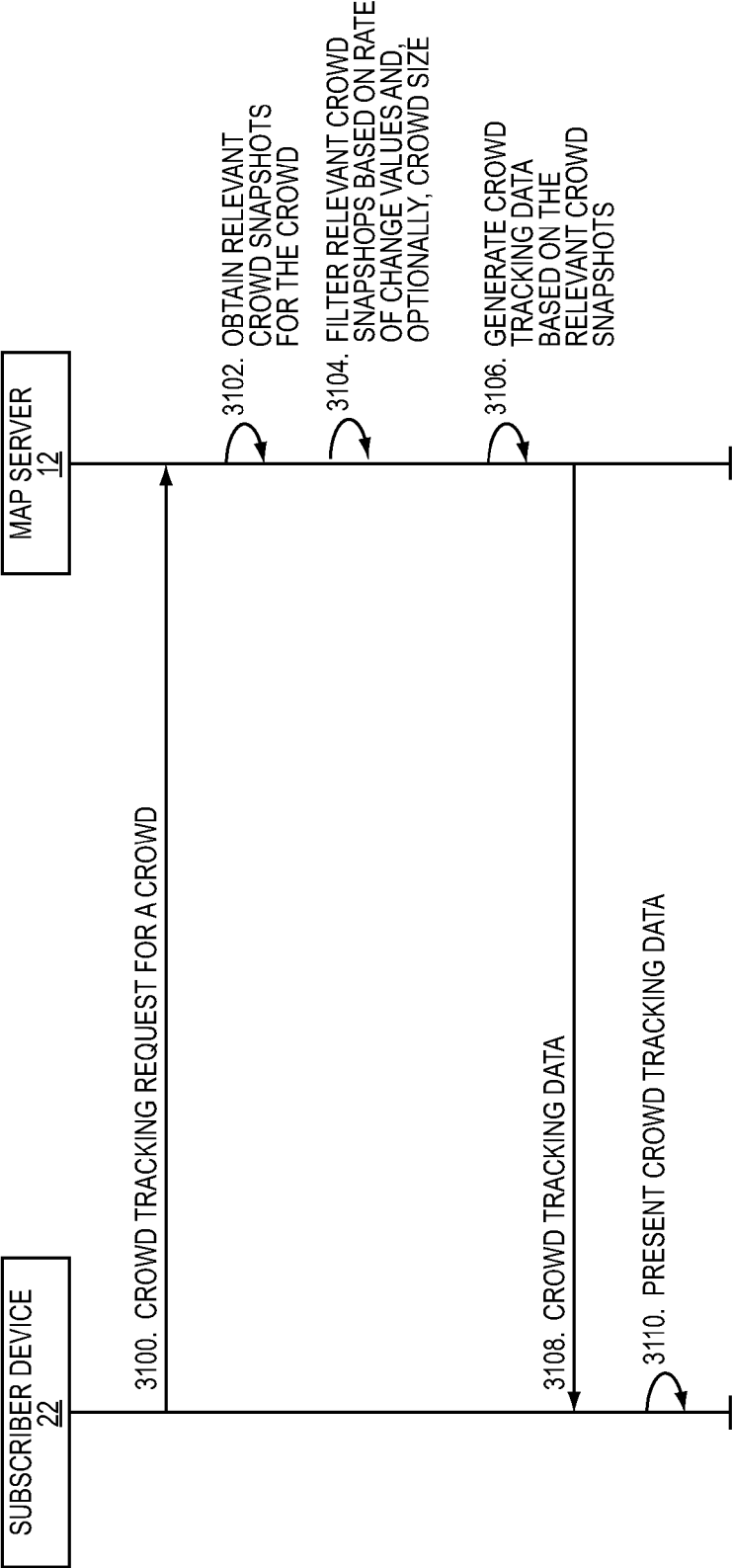
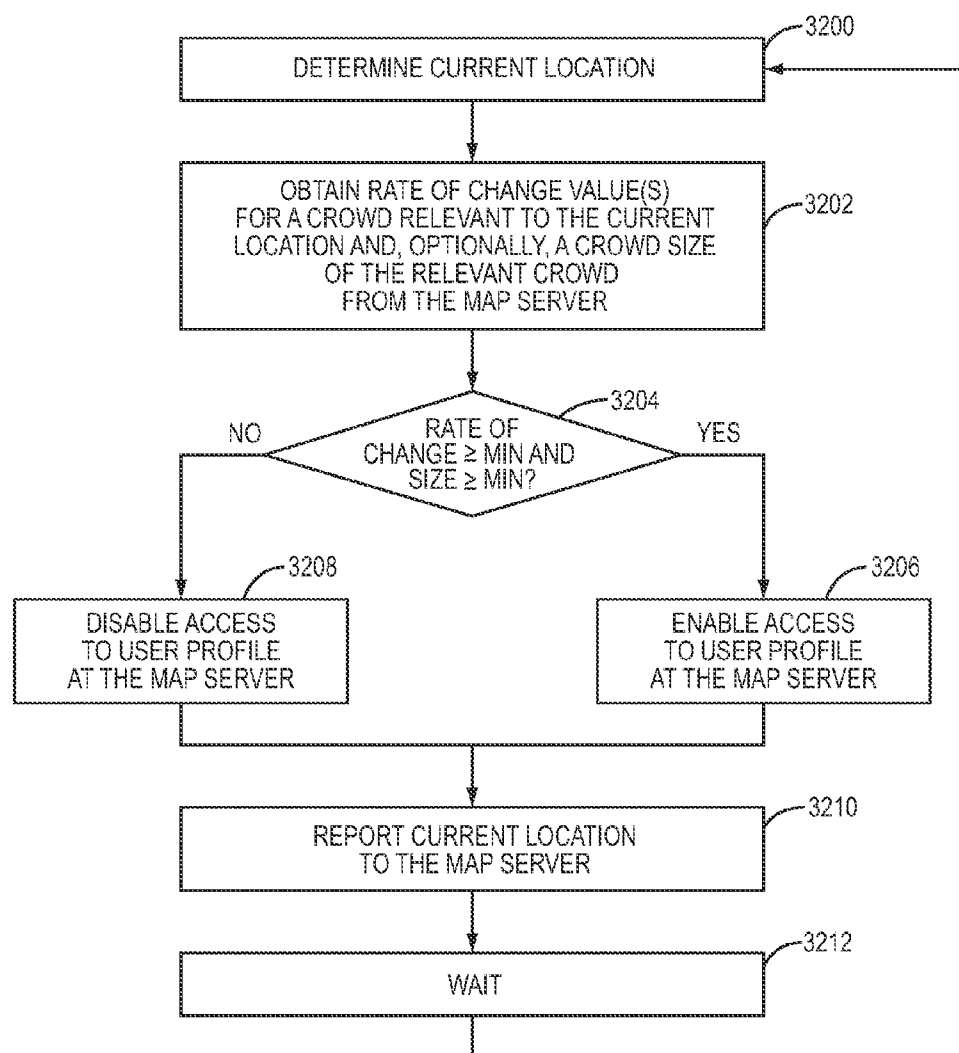


FIG. 35

**FIG. 36**

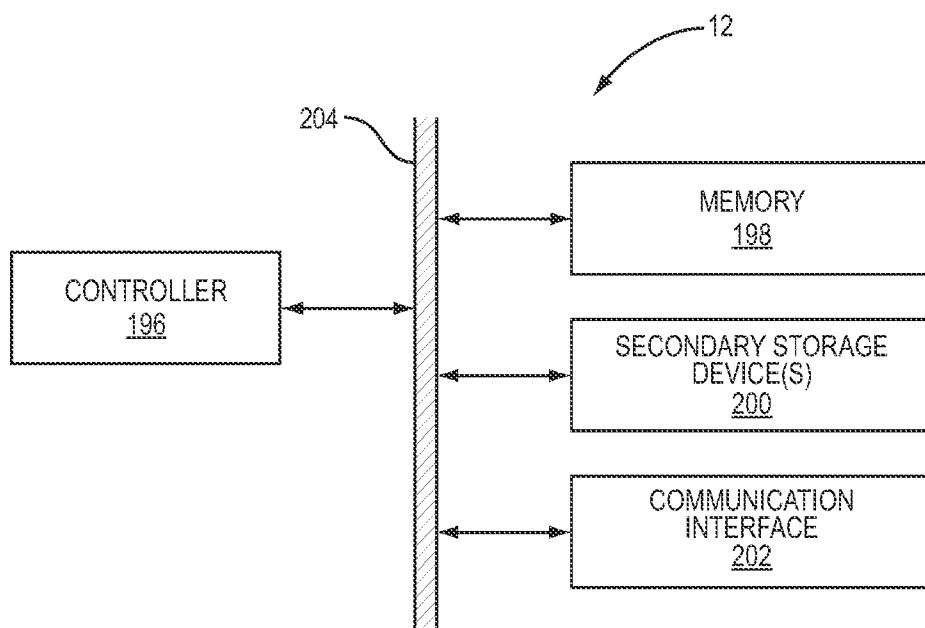


FIG. 37

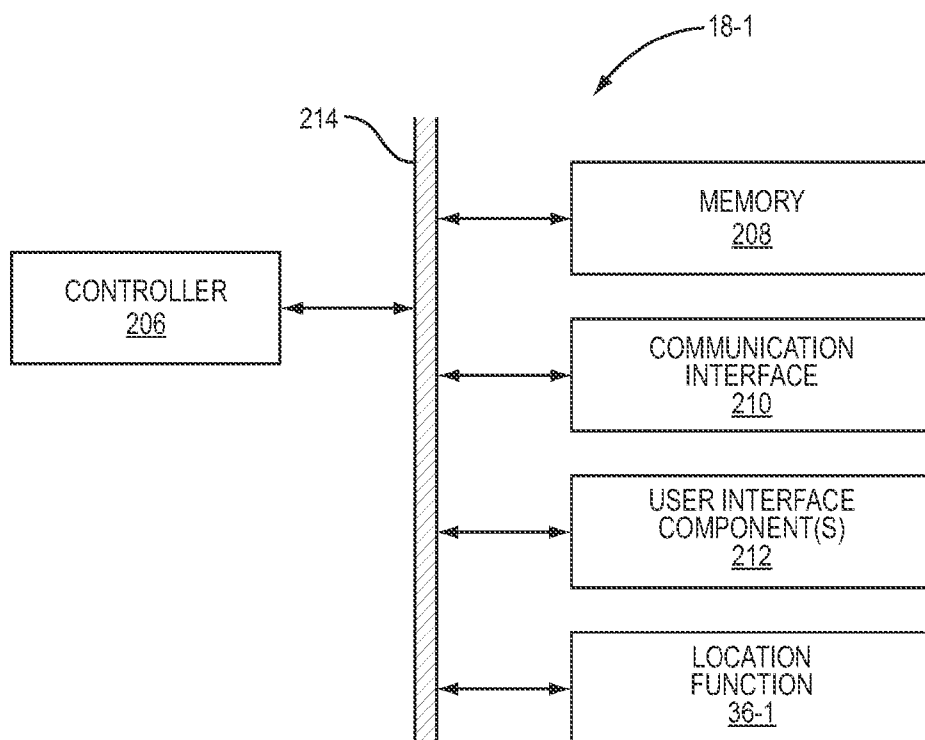


FIG. 38

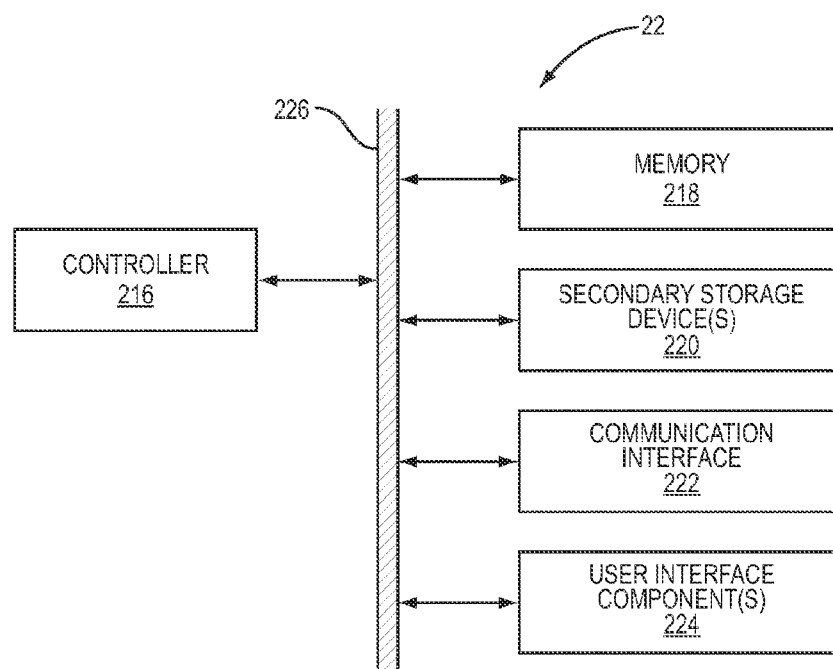


FIG. 39

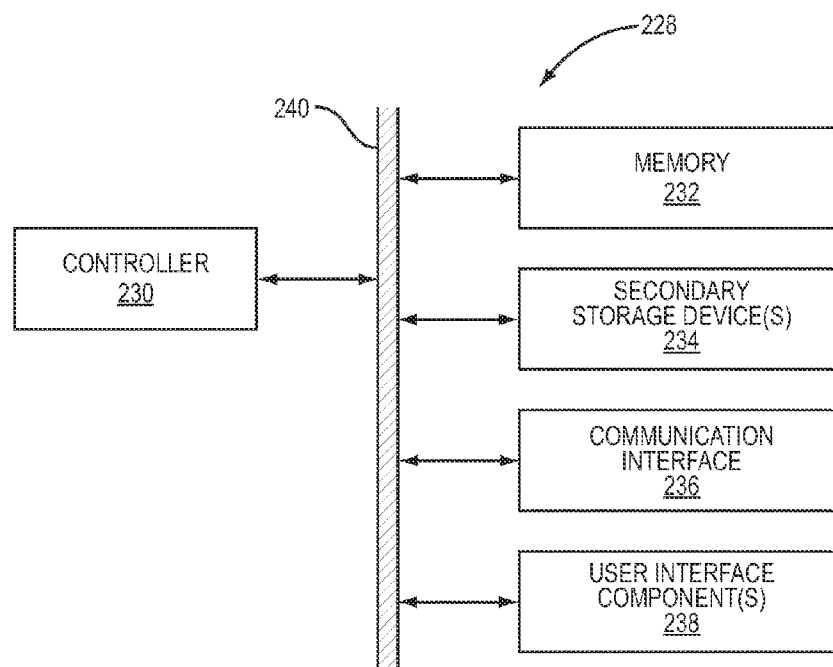


FIG. 40

SYSTEM AND METHOD FOR PREVENTION OF INDIRECT USER TRACKING THROUGH AGGREGATE PROFILE DATA

RELATED APPLICATIONS

[0001] This application claims the benefit of provisional patent application Ser. No. 61/173,625, filed Apr. 29, 2009, the disclosure of which is hereby incorporated herein by reference in its entirety.

FIELD OF THE DISCLOSURE

[0002] The present disclosure relates to a mobile aggregate profiling system and more particularly relates to systems and methods for preventing indirect user tracking through aggregate profile data.

BACKGROUND

[0003] Mobile aggregate profiling systems are becoming increasingly popular. One such system is described in commonly owned and assigned U.S. patent application Ser. No. 12/645,535 entitled MAINTAINING A HISTORICAL RECORD OF ANONYMIZED USER PROFILE DATA BY LOCATION FOR USERS IN A MOBILE ENVIRONMENT, U.S. patent application Ser. No. 12/645,532 entitled FORMING CROWDS AND PROVIDING ACCESS TO CROWD DATA IN A MOBILE ENVIRONMENT, U.S. patent application Ser. No. 12/645,539 entitled ANONYMOUS CROWD TRACKING, U.S. patent application Ser. No. 12/645,544 entitled MODIFYING A USER'S CONTRIBUTION TO AN AGGREGATE PROFILE BASED ON TIME BETWEEN LOCATION UPDATES AND EXTERNAL EVENTS, U.S. patent application Ser. No. 12/645,546 entitled CROWD FORMATION FOR MOBILE DEVICE USERS, U.S. patent application Ser. No. 12/645,556 entitled SERVING A REQUEST FOR DATA FROM A HISTORICAL RECORD OF ANONYMIZED USER PROFILE DATA IN A MOBILE ENVIRONMENT, and U.S. patent application Ser. No. 12/645,560 entitled HANDLING CROWD REQUESTS FOR LARGE GEOGRAPHIC AREAS, all of which were filed on Dec. 23, 2009 and are hereby incorporated herein by reference in their entireties.

[0004] One issue with such systems is that a user may be indirectly tracked by monitoring changes in aggregate profile data for nearby crowds or locations to detect changes that are likely attributable to a single user or small group of users that one is desiring to track. This is particularly an issue for crowds or locations having a small number of users. As such, there is a need for systems and methods to prevent such indirect user tracking in a mobile aggregate profiling system.

SUMMARY

[0005] Systems and methods are provided for preventing indirect user tracking in a mobile aggregate profiling system. In one embodiment, a Mobile Aggregate Profile (MAP) server controls whether an aggregate profile for a group of users is accessible from the MAP server based on a rate of change of one or more characteristics, such as population, of the group of users. More specifically, in one embodiment, the MAP server monitors crowds of users and stores historical information regarding the crowds, which are referred to as crowd snapshots of the crowds. In one embodiment, the MAP server controls whether a crowd snapshot of a crowd is to be created and stored by the MAP server based on a rate of

change of one or more characteristics, such as population, of the crowd at that time. In another embodiment, the MAP server controls whether to store profile data in a crowd snapshot of a crowd based on a rate of change of a characteristic, such as population, of the crowd at that time. The profile data may be user profiles of users in the crowd at that time, anonymized versions of the user profiles of the users in the crowd at that time, or an aggregate profile of the crowd at that time. In yet another embodiment, rather than controlling the creation and storage of crowd snapshots or profile data in crowd snapshots, the MAP server controls whether profile data from crowd snapshots are to be utilized when responding to requests based on a rate of change of one or more characteristics, such as population, of the corresponding crowds at the time of creating and storing the crowd snapshots.

[0006] In another embodiment, the MAP server maintains a historical record of anonymized user profile data by location, where each history object in the historical record stores anonymized user profile data for a group of users located in a corresponding geographic area during a corresponding period of time. In one embodiment, the MAP server controls whether to create and store a history object for a group of users located within a particular geographic area during a particular period of time based on a rate of change of one or more characteristics, such as population, of the group of users. In another embodiment, the MAP server controls whether to include profile data in a history object for a group of users located within a particular geographic area during a particular period of time based on a rate of change of one or more characteristics, such as population, of the group of users. The profile data may be user profiles of users in the group of users located in the particular geographic area during the particular period of time, anonymized versions of the user profiles of the users in the group of users located in the particular geographic area during the particular period of time, or an aggregate profile for the particular geographic area for the particular period of time. In yet another embodiment, rather than controlling the creation and storage of historical records or profile data in historical objects, the MAP server controls whether profile data from history objects are to be utilized when responding to historical requests based on a rate of change of one or more characteristics, such as population, of the corresponding groups of users.

[0007] In another embodiment, mobile devices of users of the aggregate profiling system control whether user profiles of the users contribute to aggregate profiles of corresponding crowds of users based on a rate of change of a characteristic, such as population, of the corresponding crowds. More specifically, in one embodiment, a mobile device of a user determines a current location of the mobile device and then obtains a rate of change of one or more characteristics, such as population, of a relevant crowd that is relevant to the current location from a mobile aggregate profile server that operates to form crowds of users and provide access to aggregate profile data for the crowds of users. The mobile device then determines whether to enable access to a user profile of a user of the mobile device at the mobile aggregate profile server based on the rate of change of the one or more characteristics of the relevant crowd of users. If a determination is made to enable access to the user profile, the mobile device enables access to the user profile of the user of the mobile device at the mobile aggregate profile server.

[0008] Those skilled in the art will appreciate the scope of the present invention and realize additional aspects thereof

after reading the following detailed description of the preferred embodiments in association with the accompanying drawing figures.

BRIEF DESCRIPTION OF THE DRAWING FIGURES

[0009] The accompanying drawing figures incorporated in and forming a part of this specification illustrate several aspects of the invention, and together with the description serve to explain the principles of the invention.

[0010] FIG. 1 illustrates a Mobile Aggregate Profile (MAP) system according to one embodiment of the present disclosure;

[0011] FIG. 2 is a block diagram of the MAP server of FIG. 1 according to one embodiment of the present disclosure;

[0012] FIG. 3 is a block diagram of the MAP client of one of the mobile devices of FIG. 1 according to one embodiment of the present disclosure;

[0013] FIG. 4 illustrates the operation of the system of FIG. 1 to provide user profiles and current locations of the users of the mobile devices to the MAP server according to one embodiment of the present disclosure;

[0014] FIG. 5 illustrates the operation of the system of FIG. 1 to provide user profiles and current locations of the users of the mobile devices to the MAP server according to another embodiment of the present disclosure;

[0015] FIGS. 6 and 7 graphically illustrate bucketization of users according to location for purposes of maintaining a historical record of anonymized user profile data by location according to one embodiment of the present disclosure;

[0016] FIG. 8 is a flow chart illustrating the operation of a foreground bucketization process performed by the MAP server to maintain the lists of users for location buckets for purposes of maintaining a historical record of anonymized user profile data by location according to one embodiment of the present disclosure;

[0017] FIG. 9 is a flow chart illustrating the anonymization and storage process performed by the MAP server for the location buckets in order to maintain a historical record of anonymized user profile data by location according to one embodiment of the present disclosure;

[0018] FIG. 10 graphically illustrates anonymization of a user record according to one embodiment of the present disclosure;

[0019] FIG. 11 is a flow chart for a storage process that may be used to store anonymized user profile data for location buckets regardless of rate of change values for one or more characteristics of corresponding groups of users according to one embodiment of the present disclosure;

[0020] FIG. 12 is a flow chart for a storage process that may be used to store anonymized user profile data for location buckets based on rate of change values for one or more characteristics of corresponding groups of users according to another embodiment of the present disclosure;

[0021] FIG. 13 is a flow chart illustrating a quadtree algorithm that may be used to process the location buckets for storage of the anonymized user profile data according to one embodiment of the present disclosure;

[0022] FIGS. 14A through 14E graphically illustrate the process of FIG. 13 for the generation of a quadtree data structure for one exemplary base quadtree region;

[0023] FIG. 15 illustrates the operation of the system of FIG. 1 wherein a mobile device is enabled to request and

receive historical data from the MAP server according to one embodiment of the present disclosure;

[0024] FIGS. 16A and 16B illustrate a flow chart for a process for generating historical data in a time context in response to a historical request from a mobile device such that profile data for groups of users having rate of change values for one or more characteristics do not contribute to the historical data according to one embodiment of the present disclosure;

[0025] FIGS. 17A and 17B illustrate a flow chart for a process for generating historical data in a time context in response to a historical request from a mobile device such that profile data for groups of users having rate of change values for one or more characteristics do not contribute to the historical data according to another embodiment of the present disclosure;

[0026] FIG. 18 is an exemplary Graphical User Interface (GUI) that may be provided by the MAP application of one of the mobile devices of FIG. 1 in order to present historical aggregate profile data in a time context according to one embodiment of the present disclosure;

[0027] FIGS. 19A and 19B illustrate a flow chart for a process for generating historical data in a geographic context in response to a historical request from a mobile device such that profile data for groups of users having rate of change values for one or more characteristics do not contribute to the historical data according to one embodiment of the present disclosure;

[0028] FIGS. 20A and 20B illustrate a flow chart for a process for generating historical data in a geographic context in response to a historical request from a mobile device such that profile data for groups of users having rate of change values for one or more characteristics do not contribute to the historical data according to another embodiment of the present disclosure;

[0029] FIG. 21 illustrates an exemplary GUI that may be provided by the MAP application of one of the mobile devices of FIG. 1 to present historical data in the geographic context according to one embodiment of the present disclosure;

[0030] FIG. 22 illustrates the operation of the system of FIG. 1 wherein the subscriber device is enabled to request and receive historical data from the MAP server according to one embodiment of the present disclosure;

[0031] FIGS. 23A and 23B illustrate a process for generating historical data in a time context in response to a historical request from a subscriber device such that profile data for groups of users having rate of change values for one or more characteristics do not contribute to the historical data according to one embodiment of the present disclosure;

[0032] FIGS. 24A and 24B illustrate a process for generating historical data in a time context in response to a historical request from a subscriber device such that profile data for groups of users having rate of change values for one or more characteristics do not contribute to the historical data according to another embodiment of the present disclosure;

[0033] FIGS. 25A and 25B illustrate a process for generating historical data in a geographic context in response to a historical request from a subscriber device such that profile data for groups of users having rate of change values for one or more characteristics do not contribute to the historical data according to one embodiment of the present disclosure;

[0034] FIGS. 26A and 26B illustrate a process for generating historical data in a geographic context in response to a historical request from a subscriber device such that profile

data for groups of users having rate of change values for one or more characteristics do not contribute to the historical data according to another embodiment of the present disclosure;

[0035] FIG. 27 illustrates exemplary data records that may be used to represent crowds, users, crowd snapshots, and anonymous users according to one embodiment of the present disclosure;

[0036] FIGS. 28A through 28D illustrate one embodiment of a spatial crowd formation process that may be used to enable crowd tracking according to one embodiment of the present disclosure;

[0037] FIGS. 29A through 29D graphically illustrate the crowd formation process of FIGS. 28A through 28D for a scenario where the crowd formation process is triggered by a location update for a user having no old location;

[0038] FIGS. 30A through 30F graphically illustrate the crowd formation process of FIGS. 28A through 28D for a scenario where the new and old bounding boxes overlap;

[0039] FIGS. 31A through 31E graphically illustrate the crowd formation process of FIGS. 28A through 28D in a scenario where the new and old bounding boxes do not overlap;

[0040] FIG. 32 illustrates a process for creating crowd snapshots such that crowd snapshots are stored based on rate of change values for one or more characteristics of corresponding crowds according to one embodiment of the present disclosure;

[0041] FIG. 33 illustrates a process for creating crowd snapshots such that crowd snapshots are stored regardless of rate of change values for one or more characteristics of corresponding crowds according to one embodiment of the present disclosure;

[0042] FIG. 34 illustrates the operation of the MAP server of FIG. 1 to serve a request for crowd tracking data for a crowd based on crowd snapshots stored for that crowd according to the process of FIG. 32 according to one embodiment of the present disclosure;

[0043] FIG. 35 illustrates the operation of the MAP server of FIG. 1 to serve a request for crowd tracking data for a crowd based on crowd snapshots stored for that crowd according to the process of FIG. 33 according to one embodiment of the present disclosure;

[0044] FIG. 36 is a flow chart illustrating a process for controlling whether aggregate profile data for a group of users is accessible from the MAP server according to another embodiment of the present disclosure;

[0045] FIG. 37 is a block diagram of the MAP server of FIG. 1 according to one embodiment of the present disclosure;

[0046] FIG. 38 is a block diagram of one of the mobile devices of FIG. 1 according to one embodiment of the present disclosure;

[0047] FIG. 39 is a block diagram of the subscriber device of FIG. 1 according to one embodiment of the present disclosure; and

[0048] FIG. 40 is a block diagram of a computing device operating to host the third-party service of FIG. 1 according to one embodiment of the present disclosure.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0049] The embodiments set forth below represent the necessary information to enable those skilled in the art to practice the invention and illustrate the best mode of practicing the

invention. Upon reading the following description in light of the accompanying drawing figures, those skilled in the art will understand the concepts of the invention and will recognize applications of these concepts not particularly addressed herein. It should be understood that these concepts and applications fall within the scope of the disclosure and the accompanying claims.

[0050] FIG. 1 illustrates a Mobile Aggregate Profile (MAP) system 10 (hereinafter “system 10”) according to one embodiment of the present disclosure. In this embodiment, the system 10 includes a MAP server 12, one or more profile servers 14, a location server 16, a number of mobile devices 18-1 through 18-N having associated users 20-1 through 20-N, a subscriber device 22 having an associated subscriber 24, and a third-party service 26 communicatively coupled via a network 28. The network 28 may be any type of network or any combination of networks. Specifically, the network 28 may include wired components, wireless components, or both wired and wireless components. In one exemplary embodiment, the network 28 is a distributed public network such as the Internet, where the mobile devices 18-1 through 18-N are enabled to connect to the network 28 via local wireless connections (e.g., WiFi or IEEE 802.11 connections) or wireless telecommunications connections (e.g., 3G or 4G telecommunications connections such as GSM, LTE, W-CDMA, or WiMAX connections).

[0051] As discussed below in detail, the MAP server 12 operates to obtain current locations, including location updates, and user profiles of the users 20-1 through 20-N of the mobile devices 18-1 through 18-N. The current locations of the users 20-1 through 20-N can be expressed as positional geographic coordinates such as latitude-longitude pairs, and a height vector (if applicable), or any other similar information capable of identifying a given physical point in space in a two-dimensional or three-dimensional coordinate system. Using the current locations and user profiles of the users 20-1 through 20-N, the MAP server 12 is enabled to provide a number of features such as, but not limited to, maintaining a historical record of anonymized user profile data by location, generating aggregate profile data over time for a Point of Interest (POI) or Area of Interest (AOI) using the historical record of anonymized user profile data, identifying crowds of users using current locations and/or user profiles of the users 20-1 through 20-N, generating aggregate profiles for crowds of users at a POI or in an AOI using the current user profiles of users in the crowds, and crowd tracking. While not essential, for additional information regarding the MAP server 12, the interested reader is directed to U.S. patent application Ser. No. 12/645,535 entitled MAINTAINING A HISTORICAL RECORD OF ANONYMIZED USER PROFILE DATA BY LOCATION FOR USERS IN A MOBILE ENVIRONMENT, U.S. patent application Ser. No. 12/645,532 entitled FORMING CROWDS AND PROVIDING ACCESS TO CROWD DATA IN A MOBILE ENVIRONMENT, U.S. patent application Ser. No. 12/645,539 entitled ANONYMOUS CROWD TRACKING, U.S. Patent Application Ser. No. 12/645,544 entitled MODIFYING A USER'S CONTRIBUTION TO AN AGGREGATE PROFILE BASED ON TIME BETWEEN LOCATION UPDATES AND EXTERNAL EVENTS, U.S. patent application Ser. No. 12/645,546 entitled CROWD FORMATION FOR MOBILE DEVICE USERS, U.S. patent application Ser. No. 12/645,556 entitled SERVING A REQUEST FOR DATA FROM A HISTORICAL RECORD OF ANONYMIZED USER PROFILE DATA IN A MOBILE

ENVIRONMENT, and U.S. patent application Ser. No. 12/645,560 entitled HANDLING CROWD REQUESTS FOR LARGE GEOGRAPHIC AREAS, all of which were filed on Dec. 23, 2009 and have been incorporated herein by reference in their entireties. Note that while the MAP server 12 is illustrated as a single server for simplicity and ease of discussion, it should be appreciated that the MAP server 12 may be implemented as a single physical server or multiple physical servers operating in a collaborative manner for purposes of redundancy and/or load sharing.

[0052] In general, the one or more profile servers 14 operate to store user profiles for a number of persons including the users 20-1 through 20-N of the mobile devices 18-1 through 18-N. For example, the one or more profile servers 14 may be servers providing social network services such as the Facebook® social networking service, the MySpace® social networking service, the LinkedIn® social networking service, or the like. As discussed below, in one embodiment, the MAP server 12 is enabled to directly or indirectly obtain the user profiles of the users 20-1 through 20-N of the mobile devices 18-1 through 18-N using the one or more profile servers 14. The location server 16 generally operates to receive location updates from the mobile devices 18-1 through 18-N and make the location updates available to entities such as, for instance, the MAP server 12. In one exemplary embodiment, the location server 16 is a server operating to provide Yahoo!'s FireEagle service.

[0053] The mobile devices 18-1 through 18-N may be mobile smart phones, portable media player devices, mobile gaming devices, or the like. Some exemplary mobile devices that may be programmed or otherwise configured to operate as the mobile devices 18-1 through 18-N are the Apple® iPhone®, the Palm Pre™, the Samsung Rogue™, the BlackBerry® Storm™, and the Apple® iPod Touch® device. However, this list of exemplary mobile devices is not exhaustive and is not intended to limit the scope of the present disclosure.

[0054] The mobile devices 18-1 through 18-N include MAP clients 30-1 through 30-N, MAP applications 32-1 through 32-N, third-party applications 34-1 through 34-N, and location functions 36-1 through 36-N, respectively. Using the mobile device 18-1 as an example, the MAP client 30-1 is preferably implemented in software. In general, in the preferred embodiment, the MAP client 30-1 is a middleware layer operating to interface an application layer (i.e., the MAP application 32-1 and the third-party applications 34-1) to the MAP server 12. More specifically, the MAP client 30-1 enables the MAP application 32-1 and the third-party applications 34-1 to request and receive data from the MAP server 12. In addition, the MAP client 30-1 enables applications, such as the MAP application 32-1 and the third-party applications 34-1, to access data from the MAP server 12. Note that while the MAP client 30-1 is implemented separately from the MAP application 32-1 and the third-party applications 34-1 in this embodiment, one of ordinary skill in the art will appreciate that the MAP client 30-1, or functions of the MAP client 30-1, may alternatively be implemented in the MAP application 32-1 and the third party applications 34-1.

[0055] The MAP application 32-1 is also preferably implemented in software. The MAP application 32-1 generally provides a user interface component between the user 20-1 and the MAP server 12. More specifically, among other things, the MAP application 32-1 may enable the user 20-1 to initiate historical requests for historical data or crowd requests for crowd data (e.g., aggregate profile data and/or

crowd characteristics data) from the MAP server 12 for a POI or AOI. The MAP application 32-1 may also enable the user 20-1 to configure various settings. For example, the MAP application 32-1 may enable the user 20-1 to select a desired social networking service (e.g., Facebook®, MySpace®, LinkedIn®, etc.) from which to obtain the user profile of the user 20-1 and provide any necessary credentials (e.g., username and password) needed to access the user profile from the social networking service.

[0056] The third-party applications 34-1 are preferably implemented in software. The third-party applications 34-1 operate to access the MAP server 12 via the MAP client 30-1. The third-party applications 34-1 may utilize data obtained from the MAP server 12 in any desired manner. As an example, one of the third party applications 34-1 may be a gaming application that utilizes historical aggregate profile data to notify the user 20-1 of POIs or AOIs where persons having an interest in the game have historically congregated.

[0057] The location function 36-1 may be implemented in hardware, software, or a combination thereof. In general, the location function 36-1 operates to determine or otherwise obtain the location of the mobile device 18-1. For example, the location function 36-1 may be or include a Global Positioning System (GPS) receiver.

[0058] The subscriber device 22 is a physical device such as a personal computer, a mobile computer (e.g., a notebook computer, a netbook computer, a tablet computer, etc.), a mobile smart phone, or the like. The subscriber 24 associated with the subscriber device 22 is a person or entity. In general, the subscriber device 22 enables the subscriber 24 to access the MAP server 12 via a web browser 38 to obtain various types of data, preferably for a fee. For example, the subscriber 24 may pay a fee to have access to historical aggregate profile data for one or more POIs and/or one or more AOIs, pay a fee to have access to crowd data such as aggregate profiles for crowds located at one or more POIs and/or located in one or more AOIs, pay a fee to track crowds, or the like. Note that the web browser 38 is exemplary. In another embodiment, the subscriber device 22 is enabled to access the MAP server 12 via a custom application.

[0059] Lastly, the third-party service 26 is a service that has access to data from the MAP server 12 such as a historical aggregate profile data for one or more POIs or one or more AOIs, crowd data such as aggregate profiles for one or more crowds at one or more POIs or within one or more AOIs, or crowd tracking data. Based on the data from the MAP server 12, the third-party service 26 operates to provide a service such as, for example, targeted advertising. For example, the third-party service 26 may obtain anonymous aggregate profile data for one or more crowds located at a POI and then provide targeted advertising to known users located at the POI based on the anonymous aggregate profile data. Note that while targeted advertising is mentioned as an exemplary third-party service 26, other types of third-party services 26 may additionally or alternatively be provided. Other types of third-party services 26 that may be provided will be apparent to one of ordinary skill in the art upon reading this disclosure.

[0060] Before proceeding, it should be noted that while the system 10 of FIG. 1 illustrates an embodiment where the one or more profile servers 14 and the location server 16 are separate from the MAP server 12, the present disclosure is not limited thereto. In an alternative embodiment, the functionality of the one or more profile servers 14 and/or the location server 16 may be implemented within the MAP server 12.

[0061] FIG. 2 is a block diagram of the MAP server 12 of FIG. 1 according to one embodiment of the present disclosure. As illustrated, the MAP server 12 includes an application layer 40, a business logic layer 42, and a persistence layer 44. The application layer 40 includes a user web application 46, a mobile client/server protocol component 48, and one or more data Application Programming Interfaces (APIs) 50. The user web application 46 is preferably implemented in software and operates to provide a web interface for users, such as the subscriber 24, to access the MAP server 12 via a web browser. The mobile client/server protocol component 48 is preferably implemented in software and operates to provide an interface between the MAP server 12 and the MAP clients 30-1 through 30-N hosted by the mobile devices 18-1 through 18-N. The data APIs 50 enable third-party services, such as the third-party service 26, to access the MAP server 12.

[0062] The business logic layer 42 includes a profile manager 52, a location manager 54, a history manager 56, a crowd analyzer 58, and an aggregation engine 60, each of which is preferably implemented in software. In one embodiment, the profile manager 52 operates to obtain the user profiles of the users 20-1 through 20-N directly or indirectly from the one or more profile servers 14 and store the user profiles in the persistence layer 44. In another embodiment, the profile manager 52 operates to obtain the user profiles of the users 20-1 through 20-N from the mobile devices 18-1 through 18-N of the users 20-1 through 20-N. The location manager 54 operates to obtain the current locations of the users 20-1 through 20-N including location updates. The current locations of the users 20-1 through 20-N may be obtained directly from the mobile devices 18-1 through 18-N and/or obtained from the location server 16.

[0063] The history manager 56 generally operates to maintain a historical record of anonymized user profile data by location. The crowd analyzer 58 operates to form crowds of users. In one embodiment, the crowd analyzer 58 utilizes a spatial crowd formation algorithm. However, the present disclosure is not limited thereto. In addition, the crowd analyzer 58 may also operate to track crowds. The aggregation engine 60 generally operates to provide aggregate profile data in response to requests from the mobile devices 18-1 through 18-N, the subscriber device 22, and the third-party service 26. The aggregate profile data may be historical aggregate profile data for one or more POIs or one or more AOIs or aggregate profile data for crowd(s) currently at one or more POIs or within one or more AOIs.

[0064] The persistence layer 44 includes an object mapping layer 62 and a datastore 64. The object mapping layer 62 is preferably implemented in software. The datastore 64 is preferably a relational database, which is implemented in a combination of hardware (i.e., physical data storage hardware) and software (i.e., relational database software). In this embodiment, the business logic layer 42 is implemented in an object-oriented programming language such as, for example, Java. As such, the object mapping layer 62 operates to map objects used in the business logic layer 42 to relational database entities stored in the datastore 64. Note that, in one embodiment, data is stored in the datastore 64 in a Resource Description Framework (RDF) compatible format.

[0065] In an alternative embodiment, rather than being a relational database, the datastore 64 may be implemented as an RDF datastore. More specifically, the RDF datastore may be compatible with RDF technology adopted by Semantic

Web activities. Namely, the RDF datastore may use the Friend-Of-A-Friend (FOAF) vocabulary for describing people, their social networks, and their interests. In this embodiment, the MAP server 12 may be designed to accept raw FOAF files describing persons, their friends, and their interests. These FOAF files are currently output by some social networking services such as LiveJournal™ and Facebook®. The MAP server 12 may then persist RDF descriptions of the users 20-1 through 20-N as a proprietary extension of the FOAF vocabulary that includes additional properties desired for the MAP system 10.

[0066] FIG. 3 illustrates the MAP client 30-1 of FIG. 1 in more detail according to one embodiment of the present disclosure. This discussion is equally applicable to the other MAP clients 30-2 through 30-N. As illustrated, in this embodiment, the MAP client 30-1 includes a MAP access API 66, a MAP middleware component 68, and a mobile client/server protocol component 70. The MAP access API 66 is implemented in software and provides an interface by which the MAP client 30-1 and the third-party applications 34-1 are enabled to access the MAP client 30-1. The MAP middleware component 68 is implemented in software and performs the operations needed for the MAP client 30-1 to operate as an interface between the MAP application 32-1 and the third-party applications 34-1 at the mobile device 18-1 and the MAP server 12. The mobile client/server protocol component 70 enables communication between the MAP client 30-1 and the MAP server 12 via a defined protocol.

[0067] FIG. 4 illustrates the operation of the system 10 of FIG. 1 to provide the user profile of the user 20-1 of the mobile device 18-1 to the MAP server 12 according to one embodiment of the present disclosure. This discussion is equally applicable to user profiles of the other users 20-2 through 20-N of the other mobile devices 18-2 through 18-N. First, an authentication process is performed (step 1000). For authentication, in this embodiment, the mobile device 18-1 authenticates with the profile server 14 (step 1000A) and the MAP server 12 (step 1000B). In addition, the MAP server 12 authenticates with the profile server 14 (step 1000C). Preferably, authentication is performed using OpenID or similar technology. However, authentication may alternatively be performed using separate credentials (e.g., username and password) of the user 20-1 for access to the MAP server 12 and the profile server 14. Assuming that authentication is successful, the profile server 14 returns an authentication succeeded message to the MAP server 12 (step 1000D), and the profile server 14 returns an authentication succeeded message to the MAP client 30-1 of the mobile device 18-1 (step 1000E).

[0068] At some point after authentication is complete, a user profile process is performed such that a user profile of the user 20-1 is obtained from the profile server 14 and delivered to the MAP server 12 (step 1002). In this embodiment, the MAP client 30-1 of the mobile device 18-1 sends a profile request to the profile server 14 (step 1002A). In response, the profile server 14 returns the user profile of the user 20-1 to the mobile device 18-1 (step 1002B). The MAP client 30-1 of the mobile device 18-1 then sends the user profile of the user 20-1 to the MAP server 12 (step 1002C). Note that while in this embodiment the MAP client 30-1 sends the complete user profile of the user 20-1 to the MAP server 12, in an alternative embodiment, the MAP client 30-1 may filter the user profile of the user 20-1 according to criteria specified by the user 20-1. For example, the user profile of the user 20-1 may

include demographic information, general interests, music interests, and movie interests, and the user 20-1 may specify that the demographic information or some subset thereof is to be filtered, or removed, before sending the user profile to the MAP server 12.

[0069] Upon receiving the user profile of the user 20-1 from the MAP client 30-1 of the mobile device 18-1, the profile manager 52 of the MAP server 12 processes the user profile (step 1002D). More specifically, in the preferred embodiment, the profile manager 52 includes social network handlers for the social network services supported by the MAP server 12. Thus, for example, if the MAP server 12 supports user profiles from the Facebook® social networking service, the MySpace® social networking service, and the LinkedIn® social networking service, the profile manager 52 may include a handler for the Facebook® social network, a handler for the MySpace® social network, and a handler for the LinkedIn® social network. The social network handlers process user profiles to generate user profiles for the MAP server 12 that include lists of keywords for each of a number of profile categories. The profile categories may be the same for each of the social network handlers or different for each of the social network handlers. Thus, for this example assume that the user profile of the user 20-1 is from the Facebook® social networking service. The profile manager 52 uses the handler for the Facebook® social network to process the user profile of the user 20-1 to map the user profile of the user 20-1 from the Facebook® social networking service to a user profile for the MAP server 12 including lists of keywords for a number of predefined profile categories. For example, for the handler for the Facebook® social network, the profile categories may be a demographic profile category, a social interaction profile category, a general interests profile category, a music interests profile category, and a movie interests profile category. As such, the user profile of the user 20-1 from the Facebook® social networking service may be processed by the handler for the Facebook® social network to create a list of keywords such as, for example, liberal, High School Graduate, 35-44, College Graduate, etc. for the demographic profile category; a list of keywords such as Seeking Friendship for the social interaction profile category; a list of keywords such as politics, technology, photography, books, etc. for the general interests profile category; a list of keywords including music genres, artist names, album names, or the like for the music interests profile category; and a list of keywords including movie titles, actor or actress names, director names, movie genres, or the like for the movie interests profile category. In one embodiment, the profile manager 52 may use natural language processing or semantic analysis. For example, if the Facebook® user profile of the user 20-1 states that the user 20-1 is 20 years old, semantic analysis may result in the keyword of 18-24 years old being stored in the user profile of the user 20-1 for the MAP server 12.

[0070] After processing the user profile of the user 20-1, the profile manager 52 of the MAP server 12 stores the resulting user profile for the user 20-1 (step 1002E). More specifically, in one embodiment, the MAP server 12 stores user records for the users 20-1 through 20-N in the datastore 64 (FIG. 2). The user profile of the user 20-1 is stored in the user record of the user 20-1. The user record of the user 20-1 includes a unique identifier of the user 20-1, the user profile of the user 20-1, and, as discussed below, a current location of the user 20-1. Note that the user profile of the user 20-1 may be updated as desired. For example, in one embodiment, the user profile of

the user 20-1 is updated by repeating step 1002 each time the user 20-1 activates the MAP application 32-1.

[0071] Note that while the discussion herein focuses on an embodiment where the user profiles of the users 20-1 through 20-N are obtained from the one or more profile servers 14, the user profiles of the users 20-1 through 20-N may be obtained in any desired manner. For example, in one alternative embodiment, the user 20-1 may identify one or more favorite websites. The profile manager 52 of the MAP server 12 may then crawl the one or more favorite websites of the user 20-1 to obtain keywords appearing in the one or more favorite websites of the user 20-1. These keywords may then be stored as the user profile of the user 20-1.

[0072] At some point, a process is performed such that a current location of the mobile device 18-1 and thus a current location of the user 20-1 is obtained by the MAP server 12 (step 1004). In this embodiment, the MAP application 32-1 of the mobile device 18-1 obtains the current location of the mobile device 18-1 from the location function 36-1 of the mobile device 18-1. The MAP application 32-1 then provides the current location of the mobile device 18-1 to the MAP client 30-1, and the MAP client 30-1 then provides the current location of the mobile device 18-1 to the MAP server 12 (step 1004A). Note that step 1004A may be repeated periodically or in response to a change in the current location of the mobile device 18-1 in order for the MAP application 32-1 to provide location updates for the user 20-1 to the MAP server 12.

[0073] In response to receiving the current location of the mobile device 18-1, the location manager 54 of the MAP server 12 stores the current location of the mobile device 18-1 as the current location of the user 20-1 (step 1004B). More specifically, in one embodiment, the current location of the user 20-1 is stored in the user record of the user 20-1 maintained in the datastore 64 of the MAP server 12. Note that in the preferred embodiment only the current location of the user 20-1 is stored in the user record of the user 20-1. In this manner, the MAP server 12 maintains privacy for the user 20-1 since the MAP server 12 does not maintain a historical record of the location of the user 20-1. As discussed below in detail, historical data maintained by the MAP server 12 is preferably anonymized in order to maintain the privacy of the users 20-1 through 20-N.

[0074] In addition to storing the current location of the user 20-1, the location manager 54 sends the current location of the user 20-1 to the location server 16 (step 1004C). In this embodiment, by providing location updates to the location server 16, the MAP server 12 in return receives location updates for the user 20-1 from the location server 16. This is particularly beneficial when the mobile device 18-1 does not permit background processes, which is the case for the Apple® iPhone®. As such, if the mobile device 18-1 is an Apple® iPhone® or similar device that does not permit background processes, the MAP application 32-1 will not be able to provide location updates for the user 20-1 to the MAP server 12 unless the MAP application 32-1 is active.

[0075] Therefore, when the MAP application 32-1 is not active, other applications running on the mobile device 18-1 (or some other device of the user 20-1) may directly or indirectly provide location updates to the location server 16 for the user 20-1. This is illustrated in step 1006 where the location server 16 receives a location update for the user 20-1 directly or indirectly from another application running on the mobile device 18-1 or an application running on another device of the user 20-1 (step 1006A). The location server 16

then provides the location update for the user 20-1 to the MAP server 12 (step 1006B). In response, the location manager 54 updates and stores the current location of the user 20-1 in the user record of the user 20-1 (step 1006C). In this manner, the MAP server 12 is enabled to obtain location updates for the user 20-1 even when the MAP application 32-1 is not active at the mobile device 18-1.

[0076] FIG. 5 illustrates the operation of the system 10 of FIG. 1 to provide the user profile of the user 20-1 of the mobile device 18-1 to the MAP server 12 according to another embodiment of the present disclosure. This discussion is equally applicable to user profiles of the other users 20-2 through 20-N of the other mobile devices 18-2 through 18-N. First, an authentication process is performed (step 1100). For authentication, in this embodiment, the mobile device 18-1 authenticates with the MAP server 12 (step 1100A), and the MAP server 12 authenticates with the profile server 14 (step 1100B). Preferably, authentication is performed using Open ID or similar technology. However, authentication may alternatively be performed using separate credentials (e.g., username and password) of the user 20-1 for access to the MAP server 12 and the profile server 14. Assuming that authentication is successful, the profile server 14 returns an authentication succeeded message to the MAP server 12 (step 1100C), and the MAP server 12 returns an authentication succeeded message to the MAP client 30-1 of the mobile device 18-1 (step 1100D).

[0077] At some point after authentication is complete, a user profile process is performed such that a user profile of the user 20-1 is obtained from the profile server 14 and delivered to the MAP server 12 (step 1102). In this embodiment, the profile manager 52 of the MAP server 12 sends a profile request to the profile server 14 (step 1102A). In response, the profile server 14 returns the user profile of the user 20-1 to the profile manager 52 of the MAP server 12 (step 1102B). Note that while in this embodiment the profile server 14 returns the complete user profile of the user 20-1 to the MAP server 12, in an alternative embodiment, the profile server 14 may return a filtered version of the user profile of the user 20-1 to the MAP server 12. The profile server 14 may filter the user profile of the user 20-1 according to criteria specified by the user 20-1. For example, the user profile of the user 20-1 may include demographic information, general interests, music interests, and movie interests, and the user 20-1 may specify that the demographic information or some subset thereof is to be filtered, or removed, before sending the user profile to the MAP server 12.

[0078] Upon receiving the user profile of the user 20-1, the profile manager 52 of the MAP server 12 processes the user profile (step 1102C). More specifically, as discussed above, in the preferred embodiment, the profile manager 52 includes social network handlers for the social network services supported by the MAP server 12. The social network handlers process user profiles to generate user profiles for the MAP server 12 that include lists of keywords for each of a number of profile categories. The profile categories may be the same for each of the social network handlers or different for each of the social network handlers.

[0079] After processing the user profile of the user 20-1, the profile manager 52 of the MAP server 12 stores the resulting user profile for the user 20-1 (step 1102D). More specifically, in one embodiment, the MAP server 12 stores user records for the users 20-1 through 20-N in the datastore 64 (FIG. 2). The user profile of the user 20-1 is stored in the user record of the

user 20-1. The user record of the user 20-1 includes a unique identifier of the user 20-1, the user profile of the user 20-1, and, as discussed below, a current location of the user 20-1. Note that the user profile of the user 20-1 may be updated as desired. For example, in one embodiment, the user profile of the user 20-1 is updated by repeating step 1102 each time the user 20-1 activates the MAP application 32-1.

[0080] Note that while the discussion herein focuses on an embodiment where the user profiles of the users 20-1 through 20-N are obtained from the one or more profile servers 14, the user profiles of the users 20-1 through 20-N may be obtained in any desired manner. For example, in one alternative embodiment, the user 20-1 may identify one or more favorite websites. The profile manager 52 of the MAP server 12 may then crawl the one or more favorite websites of the user 20-1 to obtain keywords appearing in the one or more favorite websites of the user 20-1. These keywords may then be stored as the user profile of the user 20-1.

[0081] At some point, a process is performed such that a current location of the mobile device 18-1 and thus a current location of the user 20-1 is obtained by the MAP server 12 (step 1104). In this embodiment, the MAP application 32-1 of the mobile device 18-1 obtains the current location of the mobile device 18-1 from the location function 36-1 of the mobile device 18-1. The MAP application 32-1 then provides the current location of the user 20-1 of the mobile device 18-1 to the location server 16 (step 1104A). Note that step 1104A may be repeated periodically or in response to changes in the location of the mobile device 18-1 in order to provide location updates for the user 20-1 to the MAP server 12. The location server 16 then provides the current location of the user 20-1 to the MAP server 12 (step 1104B). The location server 16 may provide the current location of the user 20-1 to the MAP server 12 automatically in response to receiving the current location of the user 20-1 from the mobile device 18-1 or in response to a request from the MAP server 12.

[0082] In response to receiving the current location of the mobile device 18-1, the location manager 54 of the MAP server 12 stores the current location of the mobile device 18-1 as the current location of the user 20-1 (step 1104C). More specifically, in one embodiment, the current location of the user 20-1 is stored in the user record of the user 20-1 maintained in the datastore 64 of the MAP server 12. Note that in the preferred embodiment only the current location of the user 20-1 is stored in the user record of the user 20-1. In this manner, the MAP server 12 maintains privacy for the user 20-1 since the MAP server 12 does not maintain a historical record of the location of the user 20-1. As discussed below in detail, historical data maintained by the MAP server 12 is preferably anonymized in order to maintain the privacy of the users 20-1 through 20-N.

[0083] As discussed above, the use of the location server 16 is particularly beneficial when the mobile device 18-1 does not permit background processes, which is the case for the Apple® iPhone®. As such, if the mobile device 18-1 is an Apple® iPhone® or similar device that does not permit background processes, the MAP application 32-1 will not provide location updates for the user 20-1 to the location server 16 unless the MAP application 32-1 is active. However, other applications running on the mobile device 18-1 (or some other device of the user 20-1) may provide location updates to the location server 16 for the user 20-1 when the MAP application 32-1 is not active. This is illustrated in step 1106 where the location server 16 receives a location update for the user

20-1 from another application running on the mobile device **18-1** or an application running on another device of the user **20-1** (step **1106A**). The location server **16** then provides the location update for the user **20-1** to the MAP server **12** (step **1106B**). In response, the location manager **54** updates and stores the current location of the user **20-1** in the user record of the user **20-1** (step **1106C**). In this manner, the MAP server **12** is enabled to obtain location updates for the user **20-1** even when the MAP application **32-1** is not active at the mobile device **18-1**.

[0084] Using the current locations of the users **20-1** through **20-N** and the user profiles of the users **20-1** through **20-N**, the MAP server **12** can provide a number of features. A first feature that may be provided by the MAP server **12** is historical storage of anonymized user profile data by location. This historical storage of anonymized user profile data by location is performed by the history manager **56** of the MAP server **12**. More specifically, as illustrated in FIG. 6, in the preferred embodiment, the history manager **56** maintains lists of users located in a number of geographic regions or areas, which are referred to herein as "location buckets." Preferably, the location buckets are defined by floor (latitude, longitude) to a desired resolution. The higher the resolution, the smaller the size of the location buckets. For example, in one embodiment, the location buckets are defined by floor (latitude, longitude) to a resolution of $1/10,000^{th}$ of a degree such that the lower left-hand corners of the squares illustrated in FIG. 6 are defined by the floor (latitude, longitude) values at a resolution of $1/10,000^{th}$ of a degree. In the example of FIG. 6, users are represented as dots, and location buckets **72** through **88** have lists of 1, 3, 2, 1, 1, 2, 1, 2, and 3 users, respectively.

[0085] As discussed below in detail, at a predetermined time interval such as, for example, 15 minutes, the history manager **56** makes a copy of the lists of users in the location buckets, anonymizes the user profiles of the users in the lists to provide anonymized user profile data for the corresponding location buckets, and stores the anonymized user profile data in a number of history objects. In one embodiment, a history object is stored for each location bucket having at least one user. In another embodiment, a rate of change of one or more characteristics, such as population, of a group of users located within each location bucket is maintained. The rate of change of a characteristic of a group of users located within a location bucket is also referred to herein as a rate of change of a characteristic of the location bucket. A history object is stored for each location bucket if the rate of change of the one or more characteristics of the corresponding group of users satisfies one or more corresponding predetermined criterion and, optionally, if the size of the corresponding group of users is at least a predetermined minimum size.

[0086] In yet another embodiment, a quadtree algorithm is used to efficiently create and store history objects for geographic regions (i.e., groups of one or more adjoining location buckets). More specifically, in one embodiment, a history object is created and stored for each of a number of geographic regions resulting from the quadtree algorithm that has at least one user. In another embodiment, a history object is created and stored for each of a number of geographic regions resulting from the quadtree algorithm formed by one or more location buckets having a combined rate of change of one or more characteristics, such as population, that satisfies a predetermined criterion and, optionally, having at least a predetermined minimum number of users.

[0087] FIG. 7 graphically illustrates a scenario where a user moves from one location bucket to another, namely, from the location bucket **74** to the location bucket **76**. As discussed below in detail, assuming that the movement occurs during the time interval between persistence of the historical data by the history manager **56**, the user is included on both the list for the location bucket **74** and the list for the location bucket **76**. However, the user is flagged or otherwise marked as inactive for the location bucket **74** and active for the location bucket **76**. As discussed below, after making a copy of the lists for the location buckets to be used to persist the historical data, users flagged as inactive are removed from the lists of users for the location buckets. Thus, in sum, once a user moves from the location bucket **74** to the location bucket **76**, the user remains in the list for the location bucket **74** until the predetermined time interval has expired and the anonymized user profile data is persisted. The user is then removed from the list for the location bucket **74**.

[0088] FIG. 8 is a flow chart illustrating the operation of a foreground "bucketization" process performed by the history manager **56** to maintain the lists of users for location buckets according to one embodiment of the present disclosure. First, the history manager **56** receives a location update for a user (step **1200**). For this discussion, assume that the location update is received for the user **20-1**. The history manager **56** then determines a location bucket corresponding to the updated location (i.e., the current location) of the user **20-1** (step **1202**). In the preferred embodiment, the location of the user **20-1** is expressed as latitude and longitude coordinates, and the history manager **56** determines the location bucket by determining floor values of the latitude and longitude coordinates, which can be written as floor(latitude, longitude) at a desired resolution. As an example, if the latitude and longitude coordinates for the location of the user **20-1** are 32.24267381553987 and -111.9249213502935, respectively, and the floor values are to be computed to a resolution of $1/10,000^{th}$ of a degree, then the floor values for the latitude and longitude coordinates are 32.2426 and -111.9249. The floor values for the latitude and longitude coordinates correspond to a particular location bucket.

[0089] After determining the location bucket for the location of the user **20-1**, the history manager **56** determines whether the user **20-1** is new to the location bucket (step **1204**). In other words, the history manager **56** determines whether the user **20-1** is already on the list of users for the location bucket. If the user **20-1** is new to the location bucket, the history manager **56** creates an entry for the user **20-1** in the list of users for the location bucket (step **1206**). In addition, the history manager **56** updates a rate of change of a population of the location bucket to reflect the addition of the user **20-1** to the location bucket (step **1208**). More specifically, the rate of change of the population of the location bucket preferably includes a rate of user ingress and/or a rate of user egress. Here, the rate of change of the population of the location bucket is updated to reflect the user's entry into that location bucket by updating the rate of user ingress for the location bucket. The rate of user ingress may be, for example, a running average of the number of users that enter the location bucket over time or a moving average of the number of users that enter the location bucket over a defined period of time such as, for example, the last hour, the last day, or the like. As a more specific example, the rate of user ingress into the location bucket may be expressed as a number of users per hour, a number of users per day, or the like. In another

embodiment, the rate of change of the population of the location bucket may be a net rate of change of the population of the location bucket taking into consideration both users that leave the location bucket and users that enter the location bucket. The net rate of change may be expressed as a rate of change of a total number of users in the location bucket for a predetermined period amount of time such as, for example, the last hour, the last day, or the like.

[0090] Note that while in this embodiment a rate of change of the population of the location bucket is updated, the present disclosure is not limited thereto. The rate of change of one or more other characteristics of the location bucket may additionally or alternatively be maintained and updated. For example, in addition to or as an alternative to the rate of change of the population of the location bucket, the history manager **56** may compute and store the rate of change of one or more components of an aggregate profile of a group of users within the location bucket, which is also referred to herein as an aggregate profile of the location bucket. For example, rate of change values for the number of occurrences of one or more interests in the aggregate profile of the location bucket and/or a ratio of the number of occurrences of one or more interests in the aggregate profile to the total number of users in the location bucket may be computed and maintained by the history manager **56**.

[0091] Returning to step **1204**, if the user **20-1** is not new to the location bucket, the history manager **56** updates the entry for the user **20-1** in the list of users for the location bucket (step **1210**). At this point, whether proceeding from step **1208** or **1210**, the user **20-1** is flagged as active in the list of users for the location bucket (step **1212**). The history manager **56** then determines whether the user **20-1** has moved from another location bucket (step **1214**). More specifically, the history manager **56** determines whether the user **20-1** is included in the list of users for another location bucket and is currently flagged as active in that list. If the user **20-1** has not moved from another location bucket, the process proceeds to step **1220**. If the user **20-1** has moved from another location bucket, the history manager **56** flags the user **20-1** as inactive in the list of users for the other location bucket from which the user **20-1** has moved (step **1216**). In addition, the history manager **56** updates a rate of change of a population of the location bucket that the user **20-1** left to reflect the user's leaving (step **1218**). More specifically, the rate of change of the population of the location bucket preferably includes a rate of user ingress and a rate of user egress. Here, the rate of change of the population of the location bucket that the user **20-1** left is updated to reflect the user's leaving that location bucket by updating the rate of user egress for the location bucket that the user **20-1** left. The rate of user egress may be, for example, a running average of the number of users that leave the location bucket or a moving average of the number of users that leave the location bucket over a defined period of time such as, for example, the last hour, the last day, or the like. As a more specific example, the rate of user egress from the location bucket may be expressed as a number of users per hour, a number of users per day, or the like. In another embodiment, the rate of change of the population of the location bucket taking into consideration users that leave the location bucket and users that enter the location bucket. The net rate of change may be expressed as a rate of

change of a total number of users in the location bucket for a predetermined amount of time such as, for example, the last hour, the last day, or the like.

[0092] Again, note that while in this embodiment, only a rate of change of the population of the location bucket is updated, the present disclosure is not limited thereto. The rate of change of one or more other characteristics of the location bucket may additionally or alternatively be maintained. For example, in addition to or as an alternative to the rate of change of the population of the location bucket, the history manager **56** may compute and store the rate of change of one or more components of an aggregate profile of a group of users within the location bucket, which is also referred to herein as an aggregate profile of the location bucket. For example, rate of change values for the number of occurrences of one or more interests in the aggregate profile of the location bucket and/or a ratio of the number of occurrences of one or more interests in the aggregate profile to the total number of users in the location bucket may be computed and maintained by the history manager **56**.

[0093] At this point, whether proceeding from step **1214** or **1218**, the history manager **56** determines whether it is time to persist (step **1220**). More specifically, as mentioned above, the history manager **56** operates to persist history objects at a predetermined time interval such as, for example, every 15 minutes. Thus, the history manager **56** determines that it is time to persist if the predetermined time interval has expired. If it is not time to persist, the process returns to step **1200** and is repeated for a next received location update, which will typically be for another user. If it is time to persist, the history manager **56** creates a copy of the lists of users for the location buckets and the rates of change in the populations of the location buckets and passes the copy of the lists and the rates of change in the populations of the location buckets to an anonymization and storage process (step **1222**). In this embodiment, the anonymization and storage process is a separate process performed by the history manager **56**. The history manager **56** then removes inactive users from the lists of users for the location buckets (step **1224**). The process then returns to step **1200** and is repeated for a next received location update, which will typically be for another user.

[0094] FIG. 9 is a flow chart illustrating the anonymization and storage process performed by the history manager **56** at the predetermined time interval according to one embodiment of the present disclosure. First, the anonymization and storage process receives the copy of the lists of users for the location buckets and the rates of change in the populations of the location buckets passed to the anonymization and storage process by the bucketization process of FIG. 8 (step **1300**). Next, anonymization is performed for each of the location buckets having at least one user in order to provide anonymized user profile data for the location buckets (step **1302**). Anonymization prevents connecting information stored in the history objects stored by the history manager **56** back to the users **20-1** through **20-N** or at least substantially increases a difficulty of connecting information stored in the history objects stored by the history manager **56** back to the users **20-1** through **20-N**. Lastly, the anonymized user profile data for the location buckets is stored in a number of history objects (step **1304**). In one embodiment, a separate history object is stored for each of the location buckets, where the history object of a location bucket includes the anonymized user profile data for the location bucket and the rate of change of the population of the location bucket received in step **1300**,

which is referred to herein as the rate of change of the population of the location bucket at the time the history object is stored. In another embodiment, as discussed below, a quadtree algorithm is used to efficiently store the anonymized user profile data in a number of history objects such that each history object stores the anonymized user profile data for one or more location buckets.

[0095] FIG. 10 graphically illustrates one embodiment of the anonymization process of step 1302 of FIG. 9. In this embodiment, anonymization is performed by creating anonymous user records for the users in the lists of users for the location buckets. The anonymous user records are not connected back to the users 20-1 through 20-N. More specifically, as illustrated in FIG. 10, each user in the lists of users for the location buckets has a corresponding user record 90. The user record 90 includes a unique user identifier (ID) for the user, the current location of the user, and the user profile of the user. The user profile includes keywords for each of a number of profile categories, which are stored in corresponding profile category records 92-1 through 92-M. Each of the profile category records 92-1 through 92-M includes a user ID for the corresponding user which may be the same user ID used in the user record 90, a category ID, and a list of keywords for the profile category.

[0096] For anonymization, an anonymous user record 94 is created from the user record 90. In the anonymous user record 94, the user ID is replaced with a new user ID that is not connected back to the user, which is also referred to herein as an anonymous user ID. This new user ID is different than any other user ID used for anonymous user records created from the user record of the user for any previous or subsequent time periods. In this manner, anonymous user records for a single user created over time cannot be linked to one another.

[0097] In addition, anonymous profile category records 96-1 through 96-M are created for the profile category records 92-1 through 92-M. In the anonymous profile category records 96-1 through 96-M, the user ID is replaced with a new user ID, which may be the same new user ID included in the anonymous user record 94. The anonymous profile category records 96-1 through 96-M include the same category IDs and lists of keywords as the corresponding profile category records 92-1 through 92-M. Note that the location of the user is not stored in the anonymous user record 94. With respect to location, it is sufficient that the anonymous user record 94 is linked to a location bucket.

[0098] In another embodiment, the history manager 56 performs anonymization in a manner similar to that described above with respect to FIG. 10. However, in this embodiment, the profile category records for the group of users in a location bucket, or the group of users in a number of location buckets representing a node in a quadtree data structure (see below), may be selectively randomized among the anonymous user records of those users. In other words, each anonymous user record would have a user profile including a selectively randomized set of profile category records (including keywords) from a cumulative list of profile category records for all of the users in the group.

[0099] In yet another embodiment, rather than creating anonymous user records 94 for the users in the lists maintained for the location buckets, the history manager 56 may perform anonymization by storing an aggregate user profile for each location bucket, or each group of location buckets representing a node in a quadtree data structure (see below). The aggregate user profile may include a list of all keywords

and potentially the number of occurrences of each keyword in the user profiles of the corresponding group of users and/or a ratio of a number of occurrences to a total number of users for the corresponding group of users for each keyword in the user profiles of the corresponding group of users. In this manner, the data stored by the history manager 56 is not connected back to the users 20-1 through 20-N.

[0100] FIG. 11 is a flow chart illustrating the storing step (step 1304) of FIG. 9 in more detail according to one embodiment of the present disclosure. First, the history manager 56 processes the location buckets using a quadtree algorithm to produce a quadtree data structure, where each node of the quadtree data structure includes one or more of the location buckets having a combined number of users that is at most a predefined maximum number of users (step 1400). The history manager 56 then gets a next node in the quadtree data structure (step 1402), and then determines whether a size of the node, or the number of users in the node, is greater than or equal to 1 (step 1404). If not, the process proceeds to step 1410. Otherwise, the history manager 56 combines the rate of change of population values for the location buckets in the node to provide a rate of change of a population for the node (step 1406). While the rate of change of population values for the location buckets may be combined in any desired manner, in one embodiment, the rate of change of population values for the location buckets are combined by summing the rate of change of population values. In another embodiment, the rate of change of population values for the location buckets are combined by averaging the rate of change of population values. The history manager 56 then stores a history object for the node where the history object includes the rate of change of population for the node from step 1406 (step 1408).

[0101] Each history object includes location information, timing information, profile data, the rate of change of population resulting from combining the rate of change value(s) for the corresponding location bucket(s), and quadtree data structure information. The location information included in the history object defines a combined geographic area of the location bucket(s) forming the corresponding node of the quadtree data structure. For example, the location information may be latitude and longitude coordinates for a northeast corner of the combined geographic area of the node of the quadtree data structure and a southwest corner of the combined geographic area for the node of the quadtree data structure. The timing information includes information defining a time window for the history object, which may be, for example, a start time for the corresponding time interval and an end time for the corresponding time interval. The profile data includes the anonymized user profile data for the users in the list(s) maintained for the location bucket(s) forming the node of the quadtree data structure for which the history object is stored. In addition, the data may include a total number of users in the location bucket(s) forming the corresponding node of the quadtree data structure. Lastly, the quadtree data structure information includes information defining a quadtree depth of the node in the quadtree data structure.

[0102] Lastly, the history manager 56 determines whether the last node in the quadtree data structure has been processed (step 1410). If not, the process returns to step 1402 and is repeated. Once the last node in the quadtree data structure has been processed, the process ends.

[0103] FIG. 12 is a flow chart illustrating the storing step (step 1304) of FIG. 9 in more detail according to another

embodiment of the present disclosure. In this embodiment, rather than storing history objects for all nodes in the quadtree data structure having at least one user, the history manager **56** only stores history objects for nodes in the quadtree data structure having rate of change of population values that satisfy a predetermined criterion. First, the history manager **56** processes the location buckets using a quadtree algorithm to produce a quadtree data structure, where each node of the quadtree data structure includes one or more of the location buckets having a combined number of users that is at most a predefined maximum number of users (step **1500**). The history manager **56** then gets a next node in the quadtree data structure (step **1502**), and combines the rates of change of population for the location buckets in the node to provide a rate of change of a population for the node (step **1504**). While the rates of change of population for the location buckets may be combined in any desired manner, in one embodiment, the rates of change of population for the location buckets are combined by summing the rates of change of population. In another embodiment, the rates of change of population for the location buckets are combined by averaging the rates of change of population.

[0104] Next, the history manager **56** determines whether the rate of change of population for the node is greater than or equal to a predetermined minimum rate of change (step **1506**). The predetermined minimum rate of change may be system-defined or user-defined depending on the particular implementation. In one embodiment, the predetermined minimum rate of change is static. In another embodiment, the predetermined minimum rate of change is dynamic. For example, the predetermined minimum rate of change may be defined as being a function of a predetermined minimum size (see step **1508** below) such that the predetermined minimum rate of change is inversely related to the predetermined minimum size. Note that in one embodiment, the rate of change of population for the node is expressed as a rate of user ingress and a rate of user egress. In this case, the rate of change of population is greater than or equal to the predetermined minimum rate of change if both the rate of user ingress and rate of user egress is greater than or equal to the same predetermined minimum rate of change value, if at least one of the rate of user ingress and rate of user egress is greater than or equal to the same predetermined minimum rate of change value, if both the rate of user ingress and rate of user egress are greater than or equal to corresponding minimum values, or if at least one of the user ingress and rate of user egress is greater than or equal to corresponding minimum values, depending on the particular implementation.

[0105] If the rate of change of the population for the node is not greater than or equal to the predetermined minimum rate of change, the process proceeds to step **1512**. Otherwise, in this embodiment, the history manager **56** determines whether a size of the node, which is the number of users in the location bucket(s) in the node, is greater than or equal to a predetermined minimum size (step **1508**). The predetermined minimum size may be system-defined or user-defined depending on the particular implementation. In one embodiment, the predetermined minimum size is static. In another embodiment, the predetermined minimum size is dynamic. For example, the predetermined minimum size may be defined as being a function of the predetermined minimum rate of change such that the predetermined minimum size is inversely related to the predetermined minimum rate of change. If the size of the node is not greater than or equal to

the predetermined minimum size, the process proceeds to step **1512**. Otherwise, the history manager **56** stores a history object for the node (step **1510**). The history manager **56** then determines whether the last node has been processed (step **1512**). If not, the process returns to step **1502** and is repeated. Once all nodes have been processed, the process ends.

[0106] Each history object includes location information, timing information, profile data, and quadtree data structure information. While not necessary, each history object may also include the rate of change of population computed for the corresponding node in step **1504**. The location information included in the history object defines a combined geographic area of the location bucket(s) forming the corresponding node of the quadtree data structure. For example, the location information may be latitude and longitude coordinates for a north-east corner of the combined geographic area of the node of the quadtree data structure and a southwest corner of the combined geographic area for the node of the quadtree data structure. The timing information includes information defining a time window for the history object, which may be, for example, a start time for the corresponding time interval and an end time for the corresponding time interval. The profile data includes the anonymized user profile data for the users in the list(s) maintained for the location bucket(s) forming the node of the quadtree data structure for which the history object is stored. In addition, the data may include a total number of users in the location bucket(s) forming the node of the quadtree data structure. Lastly, the quadtree data structure information includes information defining a quadtree depth of the node in the quadtree data structure.

[0107] Before proceeding, an alternative embodiment should be described. In the embodiment of FIG. **12**, history objects are stored only if the corresponding nodes in the quadtree data structure having rates of change of population values and, optionally, sizes that are greater than or equal to corresponding minimum values. However, in an alternative embodiment, history objects are stored for all nodes in the quadtree data structure having at least one user, but profile data is not stored in the history objects unless the rate of change of population values and, optionally, sizes of the corresponding nodes in the quadtree data structure are greater than or equal to the corresponding minimum values.

[0108] FIG. **13** is a flow chart illustrating a quadtree algorithm that may be used to process the location buckets to form the quadtree data structure in steps **1400** and **1500** of FIGS. **11** and **12**, respectively, according to one embodiment of the present disclosure. Initially, a geographic area served by the MAP server **12** is divided into a number of geographic regions, each including multiple location buckets. These geographic regions are also referred to herein as base quadtree regions. The geographic area served by the MAP server **12** may be, for example, a city, a state, a country, or the like. Further, the geographic area may be the only geographic area served by the MAP server **12** or one of a number of geographic areas served by the MAP server **12**. Preferably, the base quadtree regions have a size of $2^n \times 2^n$ location buckets, where n is an integer greater than or equal to 1.

[0109] In order to form the quadtree data structure, the history manager **56** determines whether there are any more base quadtree regions to process (step **1600**). If there are more base quadtree regions to process, the history manager **56** sets a current node to the next base quadtree region to process, which for the first iteration is the first base quadtree region (step **1602**). The history manager **56** then determines whether

the number of users in the current node is greater than a predefined maximum number of users and whether a current quadtree depth is less than a maximum quadtree depth (step 1604). In one embodiment, the maximum quadtree depth may be reached when the current node corresponds to a single location bucket. However, the maximum quadtree depth may be set such that the maximum quadtree depth is reached before the current node reaches a single location bucket.

[0110] If the number of users in the current node is greater than the predefined maximum number of users and the current quadtree depth is less than a maximum quadtree depth, the history manager 56 creates a number of child nodes for the current node (step 1606). More specifically, the history manager 56 creates a child node for each quadrant of the current node. The users in the current node are then assigned to the appropriate child nodes based on the location buckets in which the users are located (step 1608), and the current node is then set to the first child node (step 1610). At this point, the process returns to step 1604 and is repeated.

[0111] Once the number of users in the current node is not greater than the predefined maximum number of users or the maximum quadtree depth has been reached, the history manager 56 determines whether the current node has any more sibling nodes (step 1612). Sibling nodes are child nodes of the same parent node. If so, the history manager 56 sets the current node to the next sibling node of the current node (step 1614), and the process returns to step 1604 and is repeated. Once there are no more sibling nodes to process, the history manager 56 determines whether the current node has a parent node (step 1616). If so, since the parent node has already been processed, the history manager 56 determines whether the parent node has any sibling nodes that need to be processed (step 1618). If the parent node does not have any sibling nodes that need to be processed, the process returns to step 1600. If the parent node has any sibling nodes that need to be processed, the history manager 56 sets the next sibling node of the parent node to be processed as the current node (step 1620). From this point, the process returns to step 1604 and is repeated. Returning to step 1616, if the current node does not have a parent node, the process returns to step 1600 and is repeated until there are no more base quadtree regions to process. Once there are no more base quadtree regions to process, the finished quadtree data structure is returned to the process of FIG. 11 or 12 such that the history manager 56 can then store the history objects for nodes in the quadtree data structure as discussed above (step 1622).

[0112] FIGS. 14A through 14E graphically illustrate the process of FIG. 13 for the generation of the quadtree data structure for one exemplary base quadtree region 98. FIG. 14A illustrates the base quadtree region 98. As illustrated, the base quadtree region 98 is an 8x8 square of location buckets, where each of the small squares represents a location bucket. First, the history manager 56 determines whether the number of users in the base quadtree region 98 is greater than the predetermined maximum number of users. In this example, the predetermined maximum number of users is 3. However, the value of 3 is exemplary. Other larger values may be used for the predetermined maximum number of users such as, for example, 10, 25, 50, or the like. Since the number of users in the base quadtree region 98 is greater than 3, the history manager 56 divides the base quadtree region 98 into four child nodes 100-1 through 100-4, as illustrated in FIG. 14B.

[0113] Next, the history manager 56 determines whether the number of users in the child node 100-1 is greater than the

predetermined maximum, which again for this example is 3. Since the number of users in the child node 100-1 is greater than 3, the history manager 56 divides the child node 100-1 into four child nodes 102-1 through 102-4, as illustrated in FIG. 14C. The child nodes 102-1 through 102-4 are children of the child node 100-1. The history manager 56 then determines whether the number of users in the child node 102-1 is greater than the predetermined maximum number of users, which again is 3. Since there are more than 3 users in the child node 102-1, the history manager 56 further divides the child node 102-1 into four child nodes 104-1 through 104-N, as illustrated in FIG. 14D.

[0114] The history manager 56 then determines whether the number of users in the child node 104-1 is greater than the predetermined maximum number of users, which again is 3. Since the number of users in the child node 104-1 is not greater than the predetermined maximum number of users, the child node 104-1 is identified as a node for the finished quadtree data structure, and the history manager 56 proceeds to process the sibling nodes of the child node 104-1, which are the child nodes 104-2 through 104-4. Since the number of users in each of the child nodes 104-2 through 104-4 is less than the predetermined maximum number of users, the child nodes 104-2 through 104-4 are also identified as nodes for the finished quadtree data structure.

[0115] Once the history manager 56 has finished processing the child nodes 104-1 through 104-4, the history manager 56 identifies the parent node of the child nodes 104-1 through 104-4, which in this case is the child node 102-1. The history manager 56 then processes the sibling nodes of the child node 102-1, which are the child nodes 102-2 through 102-4. In this example, the number of users in each of the child nodes 102-2 through 102-4 is less than the predetermined maximum number of users. As such, the child nodes 102-2 through 102-4 are identified as nodes for the finished quadtree data structure.

[0116] Once the history manager 56 has finished processing the child nodes 102-1 through 102-4, the history manager 56 identifies the parent node of the child nodes 102-1 through 102-4, which in this case is the child node 100-1. The history manager 56 then processes the sibling nodes of the child node 100-1, which are the child nodes 100-2 through 100-4. More specifically, the history manager 56 determines that the child node 100-2 includes more than the predetermined maximum number of users and, as such, divides the child node 100-2 into four child nodes 106-1 through 106-4, as illustrated in FIG. 14E. Because the number of users in each of the child nodes 106-1 through 106-4 is not greater than the predetermined maximum number of users, the child nodes 106-1 through 106-4 are identified as nodes for the finished quadtree data structure. Then, the history manager 56 proceeds to process the child nodes 100-3 and 100-4. Since the number of users in each of the child nodes 100-3 and 100-4 is not greater than the predetermined maximum number of users, the child nodes 100-3 and 100-4 are identified as nodes for the finished quadtree data structure. Thus, at completion, the quadtree data structure for the base quadtree region 98 includes the child nodes 104-1 through 104-4, the child nodes 102-2 through 102-4, the child nodes 106-1 through 106-4, and the child nodes 100-3 and 100-4, as illustrated in FIG. 14E.

[0117] FIG. 15 illustrates the operation of the system 10 of FIG. 1 wherein a mobile device is enabled to request and receive historical data from the MAP server 12 according to one embodiment of the present disclosure. As illustrated, in this embodiment, the MAP application 32-1 of the mobile

device **18-1** sends a historical request to the MAP client **30-1** of the mobile device **18-1** (step **1700**). In one embodiment, the historical request identifies either a POI or an AOI and a time window. A POI is a geographic point whereas an AOI is a geographic area. In one embodiment, the historical request is for a POI and a time window, where the POI is a POI corresponding to the current location of the user **20-1**, a POI selected from a list of POIs defined by the user **20-1** of the mobile device **18-1**, a POI selected from a list of POIs defined by the MAP application **32-1** or the MAP server **12**, a POI selected by the user **20-1** from a map, a POI implicitly defined via a separate application (e.g., POI is implicitly defined as the location of the nearest Starbucks coffee house in response to the user **20-1** performing a Google search for “Starbucks”), or the like. If the POI is selected from a list of POIs, the list of POIs may include static POIs which may be defined by street addresses or latitude and longitude coordinates, dynamic POIs which may be defined as the current locations of one or more friends of the user **20-1**, or both.

[0118] In another embodiment, the historical request is for an AOI and a time window, where the AOI may be an AOI of a geographic area of a predefined shape and size centered at the current location of the user **20-1**, an AOI selected from a list of AOIs defined by the user **20-1**, an AOI selected from a list of AOIs defined by the MAP application **32-1** or the MAP server **12**, an AOI selected by the user **20-1** from a map, an AOI implicitly defined via a separate application (e.g., AOI is implicitly defined as an area of a predefined shape and size centered at the location of the nearest Starbucks coffee house in response to the user **20-1** performing a Google search for “Starbucks”), or the like. If the AOI is selected from a list of AOIs, the list of AOIs may include static AOIs, dynamic AOIs which may be defined as areas of a predefined shape and size centered at the current locations of one or more friends of the user **20-1**, or both. Note that the POI or AOI of the historical request may be selected by the user **20-1** via the MAP application **32-1**. In yet another embodiment, the MAP application **32-1** automatically uses the current location of the user **20-1** as the POI or as a center point for an AOI of a predefined shape and size.

[0119] The time window for the historical request may be relative to the current time. For example, the time window may be the last hour, the last day, the last week, the last month, or the like. Alternatively, the time window may be an arbitrary time window selected by the user **20-1** such as, for example, yesterday from 7:00 pm-9:00 pm, last Friday, last week, or the like. Note that while in this example the historical request includes a single POI or AOI and a single time window, the historical request may include multiple POIs or AOIs and/or multiple time windows.

[0120] In one embodiment, the historical request is made in response to user input from the user **20-1** of the mobile device **18-1**. For instance, in one embodiment, the user **20-1** selects either a POI or an AOI and a time window and then instructs the MAP application **32-1** to make the historical request by, for example, selecting a corresponding button on a graphical user interface. In another embodiment, the historical request is made automatically in response to some event such as, for example, opening the MAP application **32-1**.

[0121] Upon receiving the historical request from the MAP application **32-1**, the MAP client **30-1** forwards the historical request to the MAP server **12** (step **1702**). Note that the MAP client **30-1** may, in some cases, process the historical request from the MAP application **32-1** before forwarding the his-

torical request to the MAP server **12**. For example, if the historical request from the MAP application **32-1** is for multiple POIs/AOIs and/or for multiple time windows, the MAP client **30-1** may process the historical request from the MAP application **32-1** to produce multiple historical requests to be sent to the MAP server **12**. For instance, a separate historical request may be produced for each POI/AOI and time window combination. However, for this discussion, the historical request is for a single POI or AOI for a single time window.

[0122] Upon receiving the historical request from the MAP client **30-1**, the MAP server **12** processes the historical request (step **1704**). More specifically, the historical request is processed by the history manager **56** of the MAP server **12**. First, the history manager **56** obtains history objects that are relevant to the historical request from the datastore **64** of the MAP server **12**. The relevant history objects are those recorded for locations relevant to the POI or AOI and the time window for the historical request. The history manager **56** then processes the relevant history objects to provide historical aggregate profile data for the POI or AOI in a time context and/or a geographic context. More specifically, in one embodiment history objects are stored for all location buckets or quadtree data structure nodes having at least one user (e.g., the embodiment of FIG. **11**). As such, the history manager **56** generates the historical aggregate profile data for the historical request by obtaining the relevant history objects and aggregating the profile data from only those relevant history objects having rate of change of population values and, optionally, sizes that satisfy corresponding predetermined minimum values. In another embodiment, history objects are stored only for those location buckets or quadtree data structure nodes having rate of change of population values and, optionally, sizes that satisfy corresponding predetermined minimum values (e.g., the embodiment of FIG. **12**). As such, the history manager **56** generates the historical aggregate profile data for the historical request by aggregating the profile data from the relevant history objects.

[0123] Once the MAP server **12** has processed the historical request, the MAP server **12** returns the resulting historical aggregate profile data to the MAP client **30-1** (step **1706**). Again, as discussed below in detail, the historical aggregate profile data may be in a time context or a geographic context. In an alternative embodiment, the data returned to the MAP client **30-1** may be raw historical data. The raw historical data may be the relevant history objects or a subset of the relevant history objects satisfying the predetermined rate of population change and, optionally, size criteria. Alternatively, the raw historical data may be data from the relevant history objects or a subset of the relevant history objects that satisfy the predetermined rate of population change and, optionally, size criteria such as, for example, the corresponding anonymous user records in the relevant history objects, the user profiles of the corresponding anonymous user records, or the like.

[0124] Upon receiving the historical aggregate profile data, the MAP client **30-1** passes the historical aggregate profile data to the MAP application **32-1** (step **1708**). Note that in an alternative embodiment where the data returned by the MAP server **12** is raw historical data, the MAP client **30-1** may process the raw historical data to provide desired data. For example, the MAP client **30-1** may process the raw historical data in order to generate historical aggregate profile data in either a time or geographic context (e.g., average aggregate profiles for time bands within the time window of the histori-

cal request and/or average aggregate profiles for regions near the POI or within the AOI of the historical request). The MAP application 32-1 then presents the historical aggregate profile data to the user 20-1 (step 1710).

[0125] FIGS. 16A and 16B illustrate a flow chart for a process for generating historical aggregate profile data in a time context according to one embodiment of the present disclosure. In this embodiment, the historical aggregate profile data is generated based on history objects created and stored according to the process of FIG. 11 where history objects are stored regardless of their rate of change of population values. First, upon receiving a historical request, the history manager 56 establishes a bounding box for the historical request based on the POI or the AOI for the historical request (step 1800). Note that while a bounding box is used in this example, other geographic shapes may be used to define a bounding region for the historical request (e.g., a bounding circle). In this embodiment, the historical request is from a mobile device of a requesting user, which in this example is the user 20-1. If the historical request is for a POI, the bounding box is a geographic region corresponding to or surrounding the POI. For example, the bounding box may be a square geographic region of a predefined size centered on the POI. If the historical request is for an AOI, the bounding box is the AOI. In addition to establishing the bounding box, the history manager 56 establishes a time window for the historical request (step 1802). For example, if the historical request is for the last week and the current date and time are Sep. 17, 2009 at 10:00 pm, the history manager 56 may generate the time window as Sep. 10, 2009 at 10:00 pm through Sep. 17, 2009 at 10:00 pm.

[0126] Next, the history manager 56 obtains history objects relevant to the bounding box and the time window for the historical request from the datastore 64 of the MAP server 12 (step 1804). The relevant history objects are history objects recorded for time periods within or intersecting the time window and for locations, or geographic areas, within or intersecting the bounding box for the historical request. The history manager 56 also determines an output time band size (step 1806). In one exemplary embodiment, the output time band size is $\frac{1}{100}^{th}$ of the amount of time from the start of the time window to the end of the time window for the historical request. For example, if the amount of time in the time window for the historical request is one week, the output time band size may be set to $\frac{1}{100}^{th}$ of a week, which is 1.68 hours or 1 hour and 41 minutes.

[0127] The history manager 56 then sorts the relevant history objects into the appropriate output time bands of the time window for the historical request. More specifically, in this embodiment, the history manager 56 creates an empty list for each output time band of the time window (step 1808). Then, the history manager 56 gets the next history object from the history objects identified in step 1804 as being relevant to the historical request (step 1810). Next, the history manager 56 determines whether the rate of change of population for the history object is greater than or equal to a predetermined minimum rate of change and, optionally, whether the size of the history object (i.e., the number of users for the history object) is greater than or equal to a predetermined minimum size (step 1812). As discussed above, the predetermined minimum rate of change may be system-defined or user-defined depending on the particular implementation. In one embodiment, the predetermined minimum rate of change is static. Notably, in this embodiment, the predetermined minimum

rate of change and/or the minimum size may be defined by the requestor, which in this example is the user 20-1. In another embodiment, the predetermined minimum rate of change is dynamic. For example, the predetermined minimum rate of change may be defined as being a function of the predetermined minimum size such that the predetermined minimum rate of change is inversely related to the predetermined minimum size. The predetermined minimum size may be system-defined or user-defined depending on the particular implementation. In one embodiment, the predetermined minimum size is static. In another embodiment, the predetermined minimum size value is dynamic. For example, the predetermined minimum size may be defined as being a function of the predetermined minimum rate of change such that the predetermined minimum size is inversely related to the predetermined minimum rate of change. Again, note that in one embodiment, the rate of change of population for the node is expressed as a rate of user ingress and a rate of user egress. In this case, the rate of change of population is greater than or equal to the predetermined minimum rate of change if both the rate of user ingress and the rate of user egress is greater than or equal to the same predetermined minimum rate of change value, if at least one of the rate of user ingress and the rate of user egress is greater than or equal to the same predetermined minimum rate of change value, if both the rate of user ingress and the rate of user egress are greater than or equal to corresponding minimum values, or if at least one of the rate of user ingress and the rate of user egress is greater than or equal to corresponding minimum values, depending on the particular implementation.

[0128] If the rate of change of population for the history object is not greater than the predetermined minimum rate of change and, optionally, if the size of the history object is not greater than or equal to the predetermined minimum size, the process proceeds to step 1816. Otherwise, the history manager 56 adds the history object to the list(s) for the appropriate output time band(s) (step 1814). Note that if the history object is recorded for a time period that overlaps two or more of the output time bands, then the history object may be added to all of the output time bands to which the history object is relevant. The history manager 56 then determines whether there are more relevant history objects to sort into the output time bands (step 1816). If so, the process returns to step 1810 and is repeated until all of the relevant history objects have been sorted into the appropriate output time bands.

[0129] Once sorting is complete, the history manager 56 determines an equivalent depth of the bounding box (D_{BB}) within the quadtree data structures used to store the history objects (step 1818). More specifically, the area of the base quadtree region (e.g., the base quadtree region 98) is referred to as A_{BASE} . Then, at each depth of the quadtree, the area of the corresponding quadtree nodes is $(\frac{1}{4})^D * A_{BASE}$. In other words, the area of a child node is $\frac{1}{4}^{th}$ of the area of the parent node of that child node. The history manager 56 determines the equivalent depth of the bounding box (D_{BB}) by determining a quadtree depth at which the area of the corresponding quadtree nodes most closely matches an area of the bounding box (A_{BB}).

[0130] Note that the equivalent quadtree depth of the bounding box (D_{BB}) determined in step 1818 is used below in order to efficiently determine the ratios of the area of the bounding box (A_{BB}) to areas of the relevant history objects (A_{HO}). However, in an alternative embodiment, the ratios of the area of the bounding box (A_{BB}) to the areas of the relevant

history objects (A_{HO}) may be otherwise computed, in which case step 1818 would not be needed.

[0131] At this point, the process proceeds to FIG. 16B where the history manager 56 gets the list of history objects for the next output time band of the time window for the historical request (step 1820). The history manager 56 then gets the next history object in the list for the output time band (step 1822). Next, the history manager 56 sets a relevancy weight for the history object, where the relevancy weight is indicative of a relevancy of the history object to the bounding box (step 1824). For instance, a history object includes anonymized user profile data for a corresponding geographic area. If that geographic area is within or significantly overlaps the bounding box, then the history object will have a high relevancy weight. However, if the geographic area only overlaps the bounding box slightly, then the history object will have a low relevancy weight. In this embodiment, the relevancy weight for the history object is set to an approximate ratio of the area of the bounding box (A_{BB}) to an area of the history object (A_{HO}) computed based on a difference between the quadtree depth of the history object (D_{HO}) and the equivalent quadtree depth of the bounding box (D_{EQ}). The quadtree depth of the history object (D_{HO}) is stored in the history object. More specifically, in one embodiment, the relevancy weight of the history object is set according to the following:

$$\text{relevancy} = \frac{A_{BB}}{A_{HO}} \approx \left(\frac{1}{4}\right)^{D_{HO}-D_{BB}}, \text{ for } D_{HO} > D_{BB}, \text{ and}$$

$$\text{relevancy} = 1, \text{ for } D_{HO} \leq D_{BB}.$$

[0132] Next, the history manager 56 generates an aggregate profile for the history object using the user profile of the requesting user, which for this example is the user 20-1, or a select subset thereof (step 1826). Note that the requesting user 20-1 may be enabled to select a subset of his user profile to be compared to the user profiles of the anonymous user records in the history objects by, for example, selecting one or more desired profile categories. In order to generate the aggregate profile for the history object, the history manager 56 compares the user profile of the user 20-1, or the select subset thereof, to the user profiles of the anonymous user records stored in the history object. In one embodiment, the resulting aggregate profile for the history object includes a number of user matches and a total number of users. In the embodiment where user profiles include lists of keywords for a number of profile categories, the number of user matches is the number of anonymous user records in the history object having user profiles that include at least one keyword that matches at least one keyword in the user profile of the user 20-1 or at least one keyword in the select subset of the user profile of the user 20-1. The total number of users is the total number of anonymous user records in the history object.

[0133] In addition or alternatively, the aggregate profile for the history object may include a list of keywords from the user profile of the user 20-1, or the select subset of the user profile of the user 20-1, having at least one user match in the user profiles of the anonymous user records stored in the history object. Still further, the aggregate profile for the history object may include the number of user matches for each of the keywords from the user profile of the user 20-1, or the select subset of the user profile of the user 20-1, or the number of matches for each of the keywords from the user profile of the

user 20-1, or the select subset of the user profile of the user 20-1 having at least one user match. In addition or alternatively, the aggregate profile may include a ratio of the number of user matches over the total number of users for each keyword in the user profile of the user 20-1, or the select subset thereof, or a ratio of the number of user matches over the total number of users for each keyword in the user profile of the user 20-1, or the select subset thereof, having at least one user match.

[0134] The history manager 56 then determines whether there are more history objects in the list for the output time band (step 1828). If so, the process returns to step 1822 and is repeated until all of the history objects in the list for the output time band have been processed. Once all of the history objects in the list for the output time band have been processed, the history manager 56 combines the aggregate profiles of the history objects in the output time band to provide a combined aggregate profile for the output time band. More specifically, in this embodiment, the history manager 56 computes a weighted average of the aggregate profiles for the history objects in the output time band using the relevancy weights of the history objects (step 1830). In one embodiment, the aggregate profile of each of the history objects includes the number of user matches for the history object and the total number of users for the history object. In this embodiment, the weighted average of the aggregate profiles of the history objects in the output time band (i.e., the average aggregate profile for the output time band) includes the weighted average of the number of user matches for all of the history objects in the output time band, which may be computed as:

$$\text{user_matches}_{AVG} = \frac{\sum_{i=1}^n (\text{relevancy}_i \cdot \text{number_of_user_matches}_i)}{\sum_{i=1}^n \text{relevancy}_i},$$

where relevancy_i is the relevancy weight computed in step 1824 for the i-th history object, number_of_user_matches_i is the number of user matches from the aggregate profile of the i-th history object, and n is the number of history objects in the list for the output time band. In a similar manner, in this embodiment, the average aggregate profile for the output time band includes the weighted average of the total number of users for all of the history objects in the output time band, which may be computed as:

$$\text{total_users}_{AVG} = \frac{\sum_{i=1}^n (\text{relevancy}_i \cdot \text{total_users}_i)}{\sum_{i=1}^n \text{relevancy}_i},$$

where relevancy_i is the relevancy weight computed in step 1824 for the i-th history object, total_users_i is the total number of users from the aggregate profile of the i-th history object, and n is the number of history objects in the list for the output time band. In addition or alternatively, the average aggregate profile for the output time band may include the weighted average of the ratio of user matches to total users for all of the history objects in the output time band, which may be computed as:

$$\frac{\text{user_matches}}{\text{total_users}_{\text{AVG}}} = \frac{\sum_{i=1}^n \left(\text{relevancy}_i \cdot \frac{\text{number_of_user_matches}_i}{\text{total_users}_i} \right)}{\sum_{i=1}^n \text{relevancy}_i},$$

where relevancy_i is the relevancy weight computed in step 1824 for the i-th history object, number_of_user_matches_i is the number of user matches from the aggregate profile of the i-th history object, total_users_i is the total number of users from the aggregate profile of the i-th history object, and n is the number of history objects in the list for the output time band.

[0135] In addition or alternatively, if the aggregate profiles for the history objects in the output time band include the number of user matches for each keyword in the user profile of the user 20-1, or the select subset thereof, or only those having at least one user match, the average aggregate profile for the output time band may include a weighted average of the number of user matches for each of those keywords, which may be computed as:

$$\text{user_matches}_{\text{KEYWORD_j,AVG}} = \frac{\sum_{i=1}^n (\text{relevancy}_i \cdot \text{number_of_user_matches}_{\text{KEYWORD_ji}})}{\sum_{i=1}^n \text{relevancy}_i},$$

where relevancy_i is the relevancy weight computed in step 1824 for the i-th history object, number_of_user_matches_{KEYWORD_j,i} is the number of user matches for the j-th keyword for the i-th history object, and n is the number of history objects in the list for the output time band. In addition or alternatively, the average aggregate profile for the output time band may include the weighted average of the ratio of the user matches to total users for each keyword, which may be computed as:

$$\frac{\text{user_matches}}{\text{total_users}_{\text{KEYWORD_j,AVG}}} = \frac{\sum_{i=1}^n \left(\text{relevancy}_i \cdot \frac{\text{number_of_user_matches}_{\text{KEYWORD_ji}}}{\text{total_users}_i} \right)}{\sum_{i=1}^n \text{relevancy}_i},$$

where relevancy_i is the relevancy weight computed in step 1824 for the i-th history object, number_of_user_matches_{KEYWORD_j,i} is the number of user matches for the j-th keyword for the i-th history object, total_users_i is the total number of users from the aggregate profile of the i-th history object, and n is the number of history objects in the list for the output time band.

[0136] Next, the history manager 56 determines whether there are more output time bands to process (step 1832). If so, the process returns to step 1820 and is repeated until the lists for all output time bands have been processed. Once all of the output time bands have been processed, the history manager

56 outputs the combined aggregate profiles for the output time bands. More specifically, in this embodiment, the history manager 56 outputs the weighted averages of the aggregate profiles computed in step 1830 for the output time bands as the historical aggregate profile data to be returned to the mobile device 18-1 (step 1834).

[0137] FIGS. 17A and 17B illustrate a flow chart for a process for generating historical aggregate profile data in a time context according to another embodiment of the present disclosure. In this embodiment, the historical aggregate profile data is generated based on history objects created and stored according to the process of FIG. 12 where history objects are stored only if their rate of change of population values and, optionally, sizes satisfy predetermined criteria. In general, the process of FIGS. 17A and 17B is substantially the same as that of FIGS. 16A and 16B without step 1812. In other words, the process of FIGS. 17A and 17B does not include the step that filters history objects having rate of change of population values less than the predetermined minimum rate of change and, optionally, sizes less than the predetermined minimum size because such filtering has already been performed during creation and storage of the history objects according to the process of FIG. 12. Steps 1900-1932 of FIGS. 17A and 17B are substantially the same as steps 1800-1810 and 1814-1834 of FIGS. 16A and 16B. Therefore, the details of those steps are not repeated herein.

[0138] FIG. 18 is an exemplary Graphical User Interface (GUI) 108 that may be provided by the MAP application 32-1 of the mobile device 18-1 (FIG. 1) in order to present historical aggregate profile data in a time context according to one embodiment of this disclosure. In operation, the MAP application 32-1 issues a historical request for a POI 110 in the manner described above. In response, the MAP server 12 uses the process of FIGS. 16A and 16B or FIGS. 17A and 17B to generate historical aggregate profile data in response to the historical request in the time context. More specifically, the historical aggregate profile data includes an average aggregate profile for each of a number of output time bands within a time window established for the historical request. In this example, the time window is a four week period extending from the week of July 5 to the week of July 26.

[0139] Using the average aggregate profiles for the output time bands included in the historical aggregate profile data, the MAP application 32-1 generates a timeline 112 for the time window of the historical request. The timeline 112 is a graphical illustration of the average aggregate profiles for the output time bands. For example, if the average aggregate profile for each of the output time bands includes a weighted average of the number of user matches and a weighted average of the number of total users for the output time band, the timeline 112 may be indicative of the ratio of the weighted average of user matches to the weighted average of total users for each of the output time bands. In this example, the output time bands having a ratio of weighted average of user matches to weighted average of total users that is less than 0.25 are represented as having a low similarity, the output time bands having a ratio of weighted average of user matches to weighted average of total users that is in the range of 0.25-0.75 are represented as having varying degrees of intermediate similarity, and the output time bands having a ratio of weighted average of user matches to weighted average of total users that is greater than 0.75 are represented as having a high similarity. Note that output time bands for which there are no history objects may be grayed-out or otherwise indicated.

[0140] In addition, in this example, the GUI 108 also includes a second timeline 114 that zooms in on an area of the timeline 112 that includes the most activity or that includes the greatest number of output time bands having a high or medium similarity. Lastly, in this example, the GUI 108 includes an aggregate profile 116 for a crowd that is currently at the POI 110.

[0141] FIGS. 19A and 19B illustrate a flow chart of a process for generating historical aggregate profile data in a geographic context according to one embodiment of the present disclosure. In this embodiment, the historical aggregate profile data is generated based on history objects created and stored according to the process of FIG. 11 where history objects are stored regardless of their rate of change of population values. First, upon receiving a historical request, the history manager 56 establishes a bounding box for the historical request based on the POI or the AOI for the historical request (step 2000). Note that while a bounding box is used in this example, other geographic shapes may be used to define a bounding region for the historical request (e.g., a bounding circle). In this embodiment, the historical request is from a mobile device of a requesting user, which in this example is the user 20-1. If the historical request is for a POI, the bounding box is a geographic region corresponding to or surrounding the POI. For example, the bounding box may be a square geographic region of a predefined size centered on the POI. If the historical request is for an AOI, the bounding box is the AOI. In addition to establishing the bounding box, the history manager 56 establishes a time window for the historical request (step 2002). For example, if the historical request is for the last week and the current date and time are Sep. 17, 2009 at 10:00 pm, the history manager 56 may generate the time window as Sep. 10, 2009 at 10:00 pm through Sep. 17, 2009 at 10:00 pm.

[0142] Next, the history manager 56 obtains history objects relevant to the bounding box and the time window of the historical request from the datastore 64 of the MAP server 12 (step 2004). The relevant history objects are history objects recorded for time periods within or intersecting the time window and for locations, or geographic areas, within or intersecting the bounding box for the historical request. The history manager 56 then sorts the relevant history objects into base quadtree regions. More specifically, in this embodiment, the history manager 56 creates an empty list for each relevant base quadtree region (step 2006). A relevant base quadtree region is a base quadtree region within which all or at least a portion of the bounding box is located. Therefore, for example, if a bounding box is located at the intersection of four base quadtree regions such that the bounding box overlaps a portion of each of the four base quadtree regions, then all four of the bounding boxes would be identified as relevant base quadtree regions. In contrast, if the bounding box is contained within a single base quadtree region, then that base quadtree region is the only relevant base quadtree region.

[0143] The history manager 56 then gets the next history object from the history objects identified in step 2004 as being relevant to the historical request (step 2008). Next, the history manager 56 determines whether the rate of change of population for the history object is greater than or equal to a predetermined minimum rate of change and, optionally, whether the size of the history object (i.e., the number of users for the history object) is greater than or equal to a predetermined minimum size (step 2010). As discussed above, the predetermined minimum rate of change may be system-

defined or user-defined depending on the particular implementation. In one embodiment, the predetermined minimum rate of change is static. Notably, in this embodiment, the predetermined minimum rate of change and/or the minimum size may be defined by the requestor, which in this example is the user 20-1. In another embodiment, the predetermined minimum rate of change is dynamic. For example, the predetermined minimum rate of change may be defined as being a function of the predetermined minimum size such that the predetermined minimum rate of change is inversely related to the predetermined minimum size. The predetermined minimum size may be system-defined or user-defined depending on the particular implementation. In one embodiment, the predetermined minimum size is static. In another embodiment, the predetermined minimum size is dynamic. For example, the predetermined minimum size may be defined as being a function of the predetermined minimum rate of change such that the predetermined minimum size is inversely related to the predetermined minimum rate of change. Again, note that in one embodiment, the rate of change of population for the node is expressed as a rate of user ingress and a rate of user egress. In this case, the rate of change of population is greater than or equal to the predetermined minimum rate of change if both the rate of user ingress and the rate of user egress is greater than or equal to the same predetermined minimum rate of change, if at least one of the rate of user ingress and the rate of user egress is greater than or equal to the same predetermined minimum rate of change, if both the rate of user ingress and the rate of user egress are greater than or equal to corresponding minimum values, or if at least one of the rate of user ingress and the rate of user egress is greater than or equal to corresponding minimum values, depending on the particular implementation.

[0144] If the rate of change of population for the history object is not greater than the predetermined minimum rate of change and, optionally, if the size of the history object is not greater than or equal to the predetermined minimum size, the process proceeds to step 2014. Otherwise, the history manager 56 adds the history object to the list for the appropriate base quadtree region (step 2012). The history manager 56 then determines whether there are more relevant history objects to sort (step 2014). If so, the process returns to step 2008 and is repeated until all of the relevant history objects have been sorted into the appropriate base quadtree regions.

[0145] Once sorting is complete, the process proceeds to FIG. 19B. The following steps generally operate to divide each base quadtree region into a grid, where a size of each grid location is set to a smallest history record size of all the history objects sorted into the list for that base quadtree region. Using the history objects in the base quadtree region, aggregate profiles are generated for each of the grid locations covered by the history object. Then, a combined aggregate profile is generated for each grid location based on the aggregate profiles generated using the corresponding history objects.

[0146] More specifically, the history manager 56 gets the list for the next base quadtree region (step 2016). The history manager 56 then gets the next history object in the list for the base quadtree region (step 2018). Next, the history manager 56 creates an aggregate profile for the history object using the user profile of the requesting user, which in this example is the user 20-1, or a select subset of the user profile of the requesting user (step 2020). Note that the user 20-1 may be enabled to select a subset of his user profile to be used for aggregate

profile creation by, for example, selecting one or more profile categories. In order to generate the aggregate profile for the history object, the history manager **56** compares the user profile of the user **20-1**, or the select subset thereof, to the user profiles of the anonymous user records stored in the history object. In one embodiment, the resulting aggregate profile for the history object includes a number of user matches and a total number of users. In the embodiment where user profiles include lists of keywords for a number of profile categories, the number of user matches is the number of anonymous user records in the history object having user profiles that include at least one keyword that matches at least one keyword in the user profile of the user **20-1** or at least one keyword in the select subset of the user profile of the user **20-1**. The total number of users is the total number of anonymous user records in the history object.

[**0147**] In addition or alternatively, the aggregate profile for the history object may include a list of keywords from the user profile of the user **20-1**, or the select subset of the user profile of the user **20-1**, having at least one user match in the user profiles of the anonymous user records stored in the history object. Still further, the aggregate profile for the history object may include the number of user matches for each of the keywords from the user profile of the user **20-1**, or the select subset of the user profile of the user **20-1**, or the number of matches for each of the keywords from the user profile of the user **20-1**, or the select subset of the user profile of the user **20-1** having at least one user match. In addition or alternatively, the aggregate profile may include a ratio of the number of user matches over the total number of users for each keyword in the user profile of the user **20-1**, or the select subset thereof, or a ratio of the number of user matches over the total number of users for each keyword in the user profile of the user **20-1**, or the select subset thereof, having at least one user match.

[**0148**] Next, the history manager **56** determines whether a size of the history object is greater than the smallest history object size in the list of history objects for the base quadtree region (step **2022**). If not, the aggregate profile for the history object is added to an output list for the corresponding grid location for the base quadtree region (step **2024**) and the process proceeds to step **2032**. If the size of the history object is greater than the smallest history object size, the history manager **56** splits the geographic area, or location, of the history object into a number of grid locations, each of the smallest history object size of all the history objects in the list for the base quadtree region (step **2026**). The history manager **56** then divides the aggregate profile of the history object evenly over the grid locations for the history object (step **2028**) and adds resulting aggregate profiles for the grid locations to output lists for those grid locations (step **2030**). For example, if the geographic area of the history object is split into four grid locations and the aggregate profile for the history object includes eight user matches and sixteen total users, then the aggregate profile is divided evenly over the four grid locations such that each of the four grid locations is given an aggregate profile of two user matches and four total users.

[**0149**] The history manager **56** then determines whether there are more history objects to process for the base quadtree region (step **2032**). If so, the process returns to step **2018** and is repeated until all of the history objects for the base quadtree region are processed. At that point, for each grid location in the base quadtree region having at least one aggregate profile

in its output list, the history manager **56** combines the aggregate profiles in the output list for the grid location to provide a combined aggregate profile for the grid location. More specifically, in this embodiment, the history manager **56** computes average aggregate profiles for the grid locations for the base quadtree region (step **2034**). In one embodiment, for each grid location, the average aggregate profile for the grid location includes an average number of user matches and an average total number of users for all of the aggregate profiles in the output list for that grid location.

[**0150**] Next, the history manager **56** determines whether there are more relevant base quadtree regions to process (step **2036**). If so, the process returns to step **2016** and is repeated until all of the relevant base quadtree regions have been processed. At that point, the history manager **56** outputs the grid locations and the average aggregate profiles for the grid locations in each of the relevant base quadtree regions (step **2038**). The grid locations and their corresponding average aggregate profiles form the historical aggregate profile data that is returned to the mobile device **18-1** of the user **20-1** in response to the historical request.

[**0151**] FIGS. **20A** and **20B** illustrate a flow chart for a process for generating historical aggregate profile data in a geographic context according to another embodiment of the present disclosure. In this embodiment, the historical aggregate profile data is generated based on history objects created and stored according to the process of FIG. **12** where history objects are stored only if their rate of change of population values and, optionally, sizes satisfy predetermined criteria. In general, the process of FIGS. **20A** and **20B** is substantially the same as that of FIGS. **19A** and **19B** without step **2010**. In other words, the process of FIGS. **20A** and **20B** does not include the step that filters history objects having rate of change of population values less than the predetermined minimum rate of change value and, optionally, sizes less than the predetermined minimum size value because such filtering has already been performed during creation and storing of the history objects according to the process of FIG. **12**. Steps **2100-2136** of FIGS. **20A** and **20B** are substantially the same as steps **2000-2008** and **2012-2038** of FIGS. **19A** and **19B**. Therefore, the details of those steps are not repeated herein.

[**0152**] FIG. **21** illustrates an exemplary GUI **118** that may be provided by the MAP application **32-1** of the mobile device **18-1** (FIG. **1**) to present historical aggregate profile data in the geographic context to the user **20-1** in response to a historical request. As illustrated, the GUI **118** includes a map **120** including a grid **122**. The grid **122** provides graphical information indicative of aggregate profiles for grid locations returned by the MAP server **12** in response to a historical request. The GUI **118** also includes buttons **124** and **126** enabling the user **20-1** to zoom in or zoom out on the map **120**, buttons **128** and **130** enabling the user **20-1** to toggle between the traditional map view as shown or a satellite map view, buttons **132** and **134** enabling the user **20-1** to switch between a historical mode and a current mode (i.e., a view of current crowd data as discussed below in detail), and buttons **136** and **138** enabling the user **20-1** to hide or show POIs on the map **120**.

[**0153**] It should be noted that while the aggregate profiles in FIGS. **16A** through **21** are generated based on the user profile of the user **20-1** or a select subset of the user profile of the user **20-1**, the aggregate profiles may alternatively be generated based on a target user profile defined or otherwise specified by the user **20-1**. For example, the user **20-1** may

define a target profile for a type of person with which the user 20-1 would like to interact. Then, by making a historical request with the target profile, the user 20-1 can learn whether people matching the target profile are historically located at a POI or an AOI

[0154] FIG. 22 illustrates the operation of the system 10 of FIG. 1 wherein the subscriber device 22 is enabled to request and receive historical aggregate profile data from the MAP server 12 according to one embodiment of the present disclosure. Note that, in a similar manner, the third-party service 26 may send historical requests to the MAP server 12. As illustrated, in this embodiment, the subscriber device 22 sends a historical request to the MAP server 12 (step 2200). The subscriber device 22 sends the historical request to the MAP server 12 via the web browser 38. In one embodiment, the historical request identifies either a POI or an AOI and a time window. The historical request may be made in response to user input from the subscriber 24 of the subscriber device 22 or made automatically in response to an event such as, for example, navigation to a website associated with a POI (e.g., navigation to a website of a restaurant).

[0155] Upon receiving the historical request, the MAP server 12 processes the historical request to provide resulting historical aggregate profile data (step 2202). More specifically, the historical request is processed by the history manager 56 of the MAP server 12. First, the history manager 56 obtains history objects that are relevant to the historical request from the datastore 64 of the MAP server 12. The relevant history objects are those recorded for locations relevant to the POI or AOI and the time window for the historical request. The history manager 56 then processes the relevant history objects to provide historical aggregate profile data for the POI or AOI in a time context and/or a geographic context. More specifically, in one embodiment, history objects are stored for all location buckets or quadtree data structure nodes having at least one user (e.g., the embodiment of FIG. 11). As such, the history manager 56 generates the historical aggregate profile data for the historical request by obtaining the relevant history objects and aggregating the profile data from only those relevant history objects having rate of change of population values and, optionally, sizes that satisfy corresponding predetermined minimum values. In another embodiment, history objects are stored only for those location buckets or quadtree data structure nodes having rate of change of population values and, optionally, sizes that satisfy corresponding predetermined minimum values (e.g., the embodiment of FIG. 12). As such, the history manager 56 generates the historical aggregate profile data for the historical request by aggregating the profile data from the relevant history objects.

[0156] Once the MAP server 12 has processed the historical request, the MAP server 12 returns the resulting historical aggregate profile data to the subscriber device 22 (step 2204). The historical aggregate profile data may be in the time context or the geographic context. In this embodiment where the historical aggregate profile data is to be presented via the web browser 38 of the subscriber device 22, the MAP server 12 formats the historical aggregate profile data in a suitable format before sending the historical aggregate profile data to the web browser 38 of the subscriber device 22. Upon receiving the historical aggregate profile data, the web browser 38 of the subscriber device 22 presents the historical aggregate profile data to the user 20-1 (step 2206).

[0157] FIGS. 23A and 23B illustrate a process for generating historical aggregate profile data in a time context in response to a historical request from the subscriber 24 at the subscriber device 22 according to one embodiment of the present disclosure. The process of FIGS. 23A and 23B is substantially the same as that described above with respect to FIGS. 16A and 16B. More specifically, steps 2300 through 2324 are substantially the same as steps 1800 through 1824 of FIGS. 16A and 16B. Likewise, steps 2328 through 2334 are substantially the same as steps 1828 through 1834 of FIG. 16B. However, step 2326 of FIG. 23B is different from step 1826 of FIG. 16B with respect to the manner in which the aggregate profiles for the history objects in the lists for the output time bands are computed.

[0158] More specifically, in step 2326, since the historical request is from the subscriber 24, the aggregate profile for the history object is generated by comparing the user profiles of the anonymous user records in the history object to one another. In this embodiment, the aggregate profile for the history object includes an aggregate list of keywords from the user profiles of the anonymous user records, the number of occurrences of each of those keywords in the user profiles of the anonymous user records, and the total number of anonymous user records in the history object. As such, in step 2330, the weighted average of the aggregate profiles for the history objects in the output time band may provide an average aggregate profile including, for each keyword occurring in the aggregate profile of at least one of the history objects, a weighted average of the number of occurrences of the keyword. In addition, the average aggregate profile may include a weighted average of the total number of anonymous user records in the history objects. In addition or alternatively, the average aggregate profile may include, for each keyword, a weighted average of the number of occurrences of the keyword to the total number of anonymous user records.

[0159] FIGS. 24A and 24B illustrate a flow chart for a process for generating historical aggregate profile data in a time context in response to a request from the subscriber 24 at the subscriber device 22 according to another embodiment of the present disclosure. In this embodiment, the historical aggregate profile data is generated based on history objects created and stored according to the process of FIG. 12 where history objects are stored only if their rate of change of population values and, optionally, sizes satisfy predetermined criteria. In general, the process of FIGS. 24A and 24B is substantially the same as that of FIGS. 23A and 23B without step 2312. In other words, the process of FIGS. 24A and 24B does not include the step that filters history objects having rate of change of population values less than the predetermined minimum rate of change value and, optionally, sizes less than the predetermined minimum size value because such filtering has already been performed during creation and storing of the history objects according to the process of FIG. 12. Steps 2400-2432 of FIGS. 24A and 24B are substantially the same as steps 2300-2310 and 2314-2334 of FIGS. 23A and 23B. Therefore, the details of those steps are not repeated herein.

[0160] FIGS. 25A and 25B illustrate a process for generating historical aggregate profile data in a geographic context in response to a historical request from the subscriber 24 at the subscriber device 22 according to one embodiment of the present disclosure. The process of FIGS. 25A and 25B is substantially the same as that described above with respect to FIGS. 19A and 19B. More specifically, steps 2500 through 2518 and 2522 through 2538 are substantially the same as

steps **2000** through **2018** and **2022** through **2038** of FIGS. **19A** and **19B**. However, step **2520** of FIG. **25B** is different from step **2020** of FIG. **19B** with respect to the manner in which the aggregate profiles for the history objects are computed.

[**0161**] More specifically, in this embodiment, since the historical request is from the subscriber **24**, the aggregate profile for the history object is generated by comparing the user profiles of the anonymous user records in the history object to one another. In this embodiment, the aggregate profile for the history object includes an aggregate list of keywords from the user profiles of the anonymous user records, the number of occurrences of each of those keywords in the user profiles of the anonymous user records, and the total number of anonymous user records in the history object. As such, in step **2534**, the weighted average of the aggregate profiles for the each of the grid locations may provide an average aggregate profile including, for each keyword, a weighted average of the number of occurrences of the keyword. In addition, the average aggregate profile for each grid location may include a weighted average of the total number of anonymous user records. In addition or alternatively, the average aggregate profile for each grid location may include, for each keyword, a weighted average of the number of occurrences of the keyword to the total number of anonymous user records.

[**0162**] FIGS. **26A** and **26B** illustrate a flow chart for a process for generating historical aggregate profile data in a geographic context in response to a request from the subscriber **24** at the subscriber device **22** according to another embodiment of the present disclosure. In this embodiment, the historical aggregate profile data is generated based on history objects created and stored according to the process of FIG. **12** where history objects are stored only if their rate of change of population values and, optionally, sizes satisfy predetermined criteria. In general, the process of FIGS. **26A** and **26B** is substantially the same as that of FIGS. **25A** and **25B** without step **2510**. In other words, the process of FIGS. **26A** and **26B** does not include the step that filters history objects having rate of change of population values less than the predetermined minimum rate of change value and, optionally, sizes less than the predetermined minimum size value because such filtering has already been performed during creation and storing of the history objects according to the process of FIG. **12**. Steps **2600-2636** of FIGS. **26A** and **26B** are substantially the same as steps **2500-2508** and **2512-2538** of FIGS. **25A** and **25B**. Therefore, the details of those steps are not repeated herein.

[**0163**] FIGS. **27** through **35** describe aspects of embodiments of the present disclosure wherein the crowd analyzer **58** of the MAP server **12** provides a crowd tracking feature and rates of change of a characteristic, such as population, of crowds is utilized to control whether aggregate profile data for the crowds is accessible from the MAP server **12**. FIG. **27** illustrates exemplary data records that may be used to represent crowds, users, crowd snapshots, and anonymous users according to one embodiment of the present disclosure. As illustrated, for each crowd created by the crowd analyzer **58** of the MAP server **12** (i.e., each crowd created that has three or more users), a corresponding crowd record **140** is created and stored in the datastore **64** of the MAP server **12**. The crowd record **140** for a crowd includes a users field, a crowd size field, a rate of change field, a North-East (NE) corner

field, a South-West (SW) corner field, a center field, a crowd snapshots field, a split from field, and a combined into field.

[**0164**] The users field stores a set or list of user records **142** corresponding to a subset of the users **20-1** through **20-N** that are currently in the crowd. The crowd size field stores a size of the crowd, which is the number of users in the crowd. The rate of change field stores a rate of change of one or more characteristics of the crowd. For example, the rate of change field preferably stores a rate of user ingress into the crowd and/or a rate of user egress from the crowd. However, the rate of change field may additionally or alternatively store rate of change values for other characteristics of the crowd such as, for example, a rate of change of a number of user matches for one or more keywords in an aggregate list of keywords for the user profiles of all of the users in the crowd, a rate of change of a ratio of a number of user matches over a total number of users in the crowd for each one or more keywords in an aggregate list of keywords for the user profiles of all of the users in the crowd, or the like.

[**0165**] The NE corner field stores a location corresponding to a NE corner of a bounding box for the crowd. The NE corner may be defined by latitude and longitude coordinates and optionally an altitude. Similarly, the SW corner field stores a location of a SW corner of the bounding box for the crowd. Like the NE corner, the SW corner may be defined by latitude and longitude coordinates and optionally an altitude. Together, the NE corner and the SW corner define a bounding box for the crowd, where the edges of the bounding box pass through the current locations of the outermost users in the crowd. The center field stores a location corresponding to a center of the crowd. The center of the crowd may be defined by latitude and longitude coordinates and optionally an altitude. Together, the NE corner, the SW corner, and the center of the crowd form spatial information defining the location of the crowd. Note, however, that the spatial information defining the location of the crowd may include additional or alternative information depending on the particular implementation. The crowd snapshots field stores a list of crowd snapshot records **144** corresponding to crowd snapshots for the crowd. The split from field may be used to store a reference to a crowd record corresponding to another crowd from which the crowd split, and the combined into field may be used to store a reference to a crowd record corresponding to another crowd into which the crowd has been merged. The combined into field is also referred to herein as a merged into field.

[**0166**] Each of the user records **142** includes an ID field, a location field, a profile field, a crowd field, and a previous crowd field. The ID field stores a unique ID for one of the users **20-1** through **20-N** for which the user record **142** is stored. The location field stores the current location of the user, which may be defined by latitude and longitude coordinates and optionally an altitude. The profile field stores the user profile of the user, which may be defined as a list of keywords for one or more profile categories. The crowd field is used to store a reference to a crowd record of a crowd of which the user is currently a member. The previous crowd field may be used to store a reference to a crowd record of a crowd of which the user was previously a member.

[**0167**] Each of the crowd snapshot records **144** includes an anonymous users field, a NE corner field, a SW corner field, a center field, a sample time field, and a vertices field. Depending on the particular embodiment, the crowd snapshot records **144** may also include a crowd size field and a rate of change field, as described below. The anonymous users field

stores a set or list of anonymous user records **146**, which are anonymized versions of user records for the users that are in the crowd at a time the crowd snapshot was created. The NE corner field stores a location corresponding to a NE corner of a bounding box for the crowd at the time the crowd snapshot was created. The NE corner may be defined by latitude and longitude coordinates and optionally an altitude. Similarly, the SW corner field stores a location of a SW corner of the bounding box for the crowd at the time the crowd snapshot was created. Like the NE corner, the SW corner may be defined by latitude and longitude coordinates and optionally an altitude. The center field stores a location corresponding to a center of the crowd at the time the crowd snapshot was created. The center of the crowd may be defined by latitude and longitude coordinates and optionally an altitude. Together, the NE corner, the SW corner, and the center of the crowd form spatial information defining the location of the crowd at the time the crowd snapshot was created. Note, however, that the spatial information defining the location of the crowd at the time the crowd snapshot was created may include additional or alternative information depending on the particular implementation.

[0168] The sample time field stores a timestamp indicating a time at which the crowd snapshot was created. The timestamp preferably includes a date and a time of day at which the crowd snapshot was created. The vertices field stores locations of users in the crowd at the time the crowd snapshot was created that define an actual outer boundary of the crowd (e.g., as a polygon) at the time the crowd snapshot was created. Note that the actual outer boundary of a crowd may be used to show the location of the crowd when displayed to a user. The crowd size field stores the size of the crowd, which is the number of users in the crowd, from the crowd record **140** at the time the crowd snapshot was created. The rate of change field stores the rate of change of one or more characteristics of the crowd from the crowd record **140** at the time the crowd snapshot was created.

[0169] Each of the anonymous user records **146** includes an anonymous ID field and a profile field. The anonymous ID field stores an anonymous user ID, which is preferably a unique user ID that is not tied, or linked, back to any of the users **20-1** through **20-N** and particularly not tied back to the user or the user record for which the anonymous user record **146** has been created. In one embodiment, the anonymous user records **146** for a crowd snapshot record **144** are anonymized versions of the user records **142** of the users in the crowd at the time the crowd snapshot was created. The manner in which the user records **142** are anonymized to create the anonymous user records **146** may be the same as that described above with respect to maintaining a historical record of anonymized user profile data according to location. The profile field stores the anonymized user profile of the anonymous user, which may be defined as a list of keywords for one or more profile categories.

[0170] FIGS. 28A through 28D illustrate one embodiment of a spatial crowd formation process that may be used to enable the crowd tracking feature. In this embodiment, the spatial crowd formation process is triggered in response to receiving a location update for one of the users **20-1** through **20-N** and is preferably repeated for each location update received for the users **20-1** through **20-N**. As such, first, the crowd analyzer **58** receives a location update, or a new location, for a user (step **2700**). In response, the crowd analyzer **58** retrieves an old location of the user, if any (step **2702**). The old

location is the current location of the user prior to receiving the new location of the user. The crowd analyzer **58** then creates a new bounding box of a predetermined size centered at the new location of the user (step **2704**) and an old bounding box of a predetermined size centered at the old location of the user, if any (step **2706**). The predetermined size of the new and old bounding boxes may be any desired size. As one example, the predetermined size of the new and old bounding boxes is **40** meters by **40** meters. Note that if the user does not have an old location (i.e., the location received in step **2700** is the first location received for the user), then the old bounding box is essentially null. Also note that while bounding “boxes” are used in this example, the bounding regions may be of any desired shape.

[0171] Next, the crowd analyzer **58** determines whether the new and old bounding boxes overlap (step **2708**). If so, the crowd analyzer **58** creates a bounding box encompassing the new and old bounding boxes (step **2710**). For example, if the new and old bounding boxes are **40**×**40** meter regions and a **1**×**1** meter square at the northeast corner of the new bounding box overlaps a **1**×**1** meter square at the southwest corner of the old bounding box, the crowd analyzer **58** may create a **79**×**79** meter square bounding box encompassing both the new and old bounding boxes.

[0172] The crowd analyzer **58** then determines the individual users and crowds relevant to the bounding box created in step **2710** (step **2712**). Note that the crowds relevant to the bounding box are pre-existing crowds resulting from previous iterations of the spatial crowd formation process. In this embodiment, the crowds relevant to the bounding box are crowds having crowd bounding boxes that are within or overlap the bounding box established in step **2710**. In order to determine the relevant crowds, the crowd analyzer **58** queries the datastore **64** of the MAP server **12** to obtain crowd records for crowds that are within or overlap the bounding box established in step **2710**. The individual users relevant to the bounding box are users that are currently located within the bounding box and are not already members of a crowd. In order to identify the relevant individual users, the crowd analyzer **58** queries the datastore **64** of the MAP server **12** for user records of users that are currently located in the bounding box created in step **2710** and are not already members of a crowd. Next, the crowd analyzer **58** computes an optimal inclusion distance for individual users based on user density within the bounding box (step **2714**). More specifically, in one embodiment, the optimal inclusion distance for individuals, which is also referred to herein as an initial optimal inclusion distance, is set according to the following equation:

$$\text{initial_optimal_inclusion_dist} = a \cdot \sqrt{\frac{A_{\text{BoundingBox}}}{\text{number_of_users}}},$$

where a is a number between 0 and 1, $A_{\text{BoundingBox}}$ is an area of the bounding box, and number_of_users is the total number of users in the bounding box. The total number of users in the bounding box includes both individual users that are not already in a crowd and users that are already in a crowd. In one embodiment, a is $2/3$.

[0173] The crowd analyzer **58** then creates a crowd of one user for each individual user within the bounding box established in step **2710** that is not already included in a crowd and sets the optimal inclusion distance for those crowds to the

initial optimal inclusion distance (step 2716). The crowds created for the individual users are temporary crowds created for purposes of performing the crowd formation process. At this point, the process proceeds to FIG. 28B where the crowd analyzer 58 analyzes the crowds in the bounding box established in step 2710 to determine whether any of the crowd members (i.e., users in the crowds) violate the optimal inclusion distance of their crowds (step 2718). Any crowd member that violates the optimal inclusion distance of his or her crowd is then removed from that crowd and the previous crowd fields in the corresponding user records are set (step 2720). More specifically, in this embodiment, a member is removed from a crowd by removing the user record of the member from the set or list of user records in the crowd record of the crowd and setting the previous crowd stored in the user record of the user to the crowd from which the member has been removed. The crowd analyzer 58 then creates a crowd of one user for each of the users removed from their crowds in step 2720 and sets the optimal inclusion distance for the newly created crowds to the initial optimal inclusion distance (step 2722).

[0174] Next, the crowd analyzer 58 determines the two closest crowds in the bounding box (step 2724) and a distance between the two closest crowds (step 2726). The distance between the two closest crowds is the distance between the crowd centers of the two closest crowds, which are stored in the crowd records for the two closest crowds. The crowd centers may be computed using a central of mass algorithm. The crowd analyzer 58 then determines whether the distance between the two closest crowds is less than the optimal inclusion distance of a larger of the two closest crowds (step 2728). If the two closest crowds are of the same size (i.e., have the same number of users), then the optimal inclusion distance of either of the two closest crowds may be used. Alternatively, if the two closest crowds are of the same size, the optimal inclusion distances of both of the two closest crowds may be used such that the crowd analyzer 58 determines whether the distance between the two closest crowds is less than the optimal inclusion distances of both of the crowds. As another alternative, if the two closest crowds are of the same size, the crowd analyzer 58 may compare the distance between the two closest crowds to an average of the optimal inclusion distances of the two crowds.

[0175] If the distance between the two closest crowds is greater than the optimal inclusion distance, the process proceeds to step 2740. However, if the distance between the two closest crowds is less than the optimal inclusion distance, the two crowds are merged (step 2730). The manner in which the two crowds are merged differs depending on whether the two crowds are pre-existing crowds or temporary crowds created for the spatial crowd formation process. If both crowds are pre-existing crowds, one of the two crowds is selected as a non-surviving crowd and the other is selected as a surviving crowd. If one crowd is larger than the other, the smaller crowd is selected as the non-surviving crowd and the larger crowd is selected as a surviving crowd. If the two crowds are of the same size, one of the crowds is selected as the surviving crowd and the other crowd is selected as the non-surviving crowd using any desired technique. The non-surviving crowd is then merged into the surviving crowd by adding the set or list of user records for the non-surviving crowd to the set or list of user records for the surviving crowd and setting the merged into field of the non-surviving crowd to a reference to the crowd record of the surviving crowd. In addition, the crowd analyzer 58 sets the previous crowd fields of the user

records in the set or list of user records from the non-surviving crowd to a reference to the crowd record of the non-surviving crowd.

[0176] If one of the crowds is a temporary crowd and the other crowd is a pre-existing crowd, the temporary crowd is selected as the non-surviving crowd, and the pre-existing crowd is selected as the surviving crowd. The non-surviving crowd is then merged into the surviving crowd by adding the set or list of user records from the crowd record of the non-surviving crowd to the set or list of user records in the crowd record of the surviving crowd. However, since the non-surviving crowd is a temporary crowd, the previous crowd field(s) of the user record(s) of the user(s) in the non-surviving crowd are not set to a reference to the crowd record of the non-surviving crowd. Similarly, the crowd record of the temporary record may not have a merged into field, but, if it does, the merged into field is not set to a reference to the surviving crowd.

[0177] If both the crowds are temporary crowds, one of the two crowds is selected as a non-surviving crowd and the other is selected as a surviving crowd. If one crowd is larger than the other, the smaller crowd is selected as the non-surviving crowd and the larger crowd is selected as a surviving crowd. If the two crowds are of the same size, one of the crowds is selected as the surviving crowd and the other crowd is selected as the non-surviving crowd using any desired technique. The non-surviving crowd is then merged into the surviving crowd by adding the set or list of user records for the non-surviving crowd to the set or list of user records for the surviving crowd. However, since the non-surviving crowd is a temporary crowd, the previous crowd field(s) of the user record(s) of the user(s) in the non-surviving crowd are not set to a reference to the crowd record of the non-surviving crowd. Similarly, the crowd record of the temporary record may not have a merged into field, but, if it does, the merged into field is not set to a reference to the surviving crowd.

[0178] Next, the crowd analyzer 58 removes the non-surviving crowd (step 2732). In this embodiment, the manner in which the non-surviving crowd is removed depends on whether the non-surviving crowd is a pre-existing crowd or a temporary crowd. If the non-surviving crowd is a pre-existing crowd, the removal process is performed by removing or nulling the users field, the NE corner field, the SW corner field, and the center field of the crowd record of the non-surviving crowd. In this manner, the spatial information for the non-surviving crowd is removed from the corresponding crowd record such that the non-surviving or removed crowd will no longer be found in response to spatial-based queries on the datastore 64. However, the crowd snapshots for the non-surviving crowd are still available via the crowd record for the non-surviving crowd. In contrast, if the non-surviving crowd is a temporary crowd, the crowd analyzer 58 may remove the crowd by deleting the corresponding crowd record.

[0179] The crowd analyzer 58 also computes a new crowd center for the surviving crowd (step 2734). Again, a center of mass algorithm may be used to compute the crowd center of a crowd. In addition, a new optimal inclusion distance for the surviving crowd is computed (step 2736). In one embodiment, the new optimal inclusion distance for the resulting crowd is computed as:

$$\text{average} = \frac{1}{n+1} \cdot \left(\text{initial_optimal_inclusion_dist} + \sum_{i=1}^n d_i \right),$$

$$\text{optimal_inclusion_dist} = \text{average} + \sqrt{\left(\frac{1}{n} \cdot \sum_{i=1}^n (d_i - \text{average})^2 \right)},$$

where n is the number of users in the crowd and d_i is a distance between the i th user and the crowd center. In other words, the new optimal inclusion distance is computed as the average of the initial optimal inclusion distance and the distances between the users in the crowd and the crowd center plus one standard deviation.

[0180] At this point, the crowd analyzer **58** determines whether a maximum number of iterations have been performed (step **2738**). The maximum number of iterations is a predefined number that ensures that the crowd formation process does not indefinitely loop over steps **2718** through **2736** or loop over steps **2718** through **2736** more than a desired maximum number of times. If the maximum number of iterations has not been reached, the process returns to step **2718** and is repeated until either the distance between the two closest crowds is not less than the optimal inclusion distance of the larger crowd or the maximum number of iterations has been reached. At that point, the crowd analyzer **58** removes crowds with less than three users, or members (step **2740**). In this embodiment, the manner in which a crowd is removed depends on whether the crowd is a pre-existing crowd or a temporary crowd. If the crowd is a pre-existing crowd, a removal process is performed by removing or nulling the users field, the NE corner field, the SW corner field, and the center field of the crowd record of the crowd. In this manner, the spatial information for the crowd is removed from the corresponding crowd record such that the crowd will no longer be found in response to spatial-based queries on the datastore **64**. However, the crowd snapshots for the crowd are still available via the crowd record for the crowd. In contrast, if the crowd is a temporary crowd, the crowd analyzer **58** may remove the crowd by deleting the corresponding crowd record. In this manner, crowds having less than three members are removed in order to maintain privacy of individuals as well as groups of two users (e.g., a couple).

[0181] The crowd analyzer **58** then computes and stores new rate of change values for the remaining crowds (step **2742**). For each crowd, the rate of change of one or more characteristics of the crowd, such as population, are computed. Using the rate of change of the population of the crowd as an example, the rate of change of the population of the crowd preferably includes a rate of user ingress and/or a rate of user egress. The rate of user ingress may be, for example, a running average of the number of users that enter the crowd over time or a moving average of the number of users that enter the crowd over a defined period of time such as, for example, the last hour, the last day, or the like. As a more specific example, the rate of user ingress into the crowd may be expressed as a number of users per hour, a number of users per day, or the like. Likewise, the rate of user egress may be, for example, a running average of the number of users that leave the crowd or a moving average of the number of users that leave the crowd over a defined period of time such as, for example, the last hour, the last day, or the like. As a more specific example, the rate of user egress from the crowd may

be expressed as a number of users per hour, a number of users per day, or the like. In another embodiment, the rate of change of the population of the crowd may be a net rate of change of the population of the crowd taking into consideration both users that leave the crowd and users that enter the crowd. The net rate of change may be expressed as a rate of change of a total number of users in the crowd for a predetermined period amount of time such as, for example, the last hour, the last day, or the like.

[0182] Note that, in order to compute the rate of change value(s) for the crowd, data may need to be stored with respect to the crowd. For example, after performing the crowd formation process of FIGS. **28A** through **28D** in response to receiving a location update, for each affected crowd, a list of users in the crowd prior to performing the crowd formation process may be compared to a list of users in the crowd after performing the crowd formation process in order to determine a number of users that have left the crowd and a number of users that have entered the crowd. These two values may then be stored in association with a timestamp defining a time at which these two values were computed. Subsequently, the rate of user ingress for the crowd may be computed based on the number of users that have entered the crowd over a defined amount of time as determined based on the stored data. Likewise, the rate of user egress for the crowd may be computed based on the number of users that have left the crowd over a defined amount of time as determined based on the stored data. In a similar manner, rate of change values for different characteristics of the crowds may additionally or alternatively be stored. Once the rate of change values are computed and stored for the remaining crowds, the process ends.

[0183] Returning to step **2708** in FIG. **28A**, if the new and old bounding boxes do not overlap, the process proceeds to FIG. **28C** and the bounding box to be processed is set to the old bounding box (step **2744**). In general, the crowd analyzer **58** then processes the old bounding box in much that same manner as described above with respect to steps **2712** through **2742**. More specifically, the crowd analyzer **58** determines the individual users and crowds relevant to the bounding box (step **2746**). Again, note that the crowds relevant to the bounding box are pre-existing crowds resulting from previous iterations of the spatial crowd formation process. In this embodiment, the crowds relevant to the bounding box are crowds having crowd bounding boxes that are within or overlap the bounding box. The individual users relevant to the bounding box are users that are currently located within the bounding box and are not already members of a crowd. Next, the crowd analyzer **58** computes an optimal inclusion distance for individual users based on user density within the bounding box in the manner described above (step **2748**).

[0184] The crowd analyzer **58** then creates a crowd of one user for each individual user within the bounding box that is not already included in a crowd and sets the optimal inclusion distance for the crowds to the initial optimal inclusion distance (step **2750**). The crowds created for the individual users are temporary crowds created for purposes of performing the crowd formation process. At this point, the crowd analyzer **58** analyzes the crowds in the bounding box to determine whether any crowd members (i.e., users in the crowds) violate the optimal inclusion distance of their crowds (step **2752**). Any crowd member that violates the optimal inclusion distance of his or her crowd is then removed from that crowd and the previous crowd fields in the corresponding user records are set (step **2754**). More specifically, in this embodiment, a

member is removed from a crowd by removing the user record of the member from the set or list of user records in the crowd record of the crowd and setting the previous crowd stored in the user record of the user to the crowd from which the member has been removed. The crowd analyzer **58** then creates a crowd for each of the users removed from their crowds in step **2754** and sets the optimal inclusion distance for the newly created crowds to the initial optimal inclusion distance (step **2756**).

[**0185**] Next, the crowd analyzer **58** determines the two closest crowds in the bounding box (step **2758**) and a distance between the two closest crowds (step **2760**). The distance between the two closest crowds is the distance between the crowd centers of the two closest crowds. The crowd analyzer **58** then determines whether the distance between the two closest crowds is less than the optimal inclusion distance of a larger of the two closest crowds (step **2762**). If the two closest crowds are of the same size (i.e., have the same number of users), then the optimal inclusion distance of either of the two closest crowds may be used. Alternatively, if the two closest crowds are of the same size, the optimal inclusion distances of both of the two closest crowds may be used such that the crowd analyzer **58** determines whether the distance between the two closest crowds is less than the optimal inclusion distances of both of the two closest crowds. As another alternative, if the two closest crowds are of the same size, the crowd analyzer **58** may compare the distance between the two closest crowds to an average of the optimal inclusion distances of the two closest crowds.

[**0186**] If the distance between the two closest crowds is greater than the optimal inclusion distance, the process proceeds to step **2774**. However, if the distance between the two closest crowds is less than the optimal inclusion distance, the two crowds are merged (step **2764**). The manner in which the two crowds are merged differs depending on whether the two crowds are pre-existing crowds or temporary crowds created for the spatial crowd formation process. If both crowds are pre-existing crowds, one of the two crowds is selected as a non-surviving crowd and the other is selected as a surviving crowd. If one crowd is larger than the other, the smaller crowd is selected as the non-surviving crowd and the larger crowd is selected as a surviving crowd. If the two crowds are of the same size, one of the crowds is selected as the surviving crowd and the other crowd is selected as the non-surviving crowd using any desired technique. The non-surviving crowd is then merged into the surviving crowd by adding the set or list of user records for the non-surviving crowd to the set or list of user records for the surviving crowd and setting the merged into field of the non-surviving crowd to a reference to the crowd record of the surviving crowd. In addition, the crowd analyzer **58** sets the previous crowd fields of the set or list of user records from the non-surviving crowd to a reference to the crowd record of the non-surviving crowd.

[**0187**] If one of the crowds is a temporary crowd and the other crowd is a pre-existing crowd, the temporary crowd is selected as the non-surviving crowd, and the pre-existing crowd is selected as the surviving crowd. The non-surviving crowd is then merged into the surviving crowd by adding the user records from the set or list of user records from the crowd record of the non-surviving crowd to the set or list of user records in the crowd record of the surviving crowd. However, since the non-surviving crowd is a temporary crowd, the previous crowd field(s) of the user record(s) of the user(s) in the non-surviving crowd are not set to a reference to the crowd

record of the non-surviving crowd. Similarly, the crowd record of the temporary record may not have a merged into field, but, if it does, the merged into field is not set to a reference to the surviving crowd.

[**0188**] If both the crowds are temporary crowds, one of the two crowds is selected as a non-surviving crowd and the other is selected as a surviving crowd. If one crowd is larger than the other, the smaller crowd is selected as the non-surviving crowd and the larger crowd is selected as a surviving crowd. If the two crowds are of the same size, one of the crowds is selected as the surviving crowd and the other crowd is selected as the non-surviving crowd using any desired technique. The non-surviving crowd is then merged into the surviving crowd by adding the set or list of user records for the non-surviving crowd to the set or list of user records for the surviving crowd. However, since the non-surviving crowd is a temporary crowd, the previous crowd field(s) of the user record(s) of the user(s) in the non-surviving crowd are not set to a reference to the crowd record of the non-surviving crowd. Similarly, the crowd record of the temporary record may not have a merged into field, but, if it does, the merged into field is not set to a reference to the surviving crowd.

[**0189**] Next, the crowd analyzer **58** removes the non-surviving crowd (step **2766**). In this embodiment, the manner in which the non-surviving crowd is removed depends on whether the non-surviving crowd is a pre-existing crowd or a temporary crowd. If the non-surviving crowd is a pre-existing crowd, the removal process is performed by removing or nulling the users field, the NE corner field, the SW corner field, and the center field of the crowd record of the non-surviving crowd. In this manner, the spatial information for the non-surviving crowd is removed from the corresponding crowd record such that the non-surviving or removed crowd will no longer be found in response to spatial-based queries on the datastore **64**. However, the crowd snapshots for the non-surviving crowd are still available via the crowd record for the non-surviving crowd. In contrast, if the non-surviving crowd is a temporary crowd, the crowd analyzer **58** may remove the crowd by deleting the corresponding crowd record.

[**0190**] The crowd analyzer **58** also computes a new crowd center for the surviving crowd (step **2768**). Again, a center of mass algorithm may be used to compute the crowd center of a crowd. In addition, a new optimal inclusion distance for the surviving crowd is computed in the manner described above (step **2770**).

[**0191**] At this point, the crowd analyzer **58** determines whether a maximum number of iterations have been performed (step **2772**). If the maximum number of iterations has not been reached, the process returns to step **2752** and is repeated until either the distance between the two closest crowds is not less than the optimal inclusion distance of the larger crowd or the maximum number of iterations has been reached. At that point, the crowd analyzer **58** removes crowds with less than three users, or members (step **2774**). As discussed above, in this embodiment, the manner in which a crowd is removed depends on whether the crowd is a pre-existing crowd or a temporary crowd. If the crowd is a pre-existing crowd, a removal process is performed by removing or nulling the users field, the NE corner field, the SW corner field, and the center field of the crowd record of the crowd. In this manner, the spatial information for the crowd is removed from the corresponding crowd record such that the crowd will no longer be found in response to spatial-based queries on the

datastore **64**. However, the crowd snapshots for the crowd are still available via the crowd record for the crowd. In contrast, if the crowd is a temporary crowd, the crowd analyzer **58** may remove the crowd by deleting the corresponding crowd record. In this manner, crowds having less than three members are removed in order to maintain privacy of individuals as well as groups of two users (e.g., a couple).

[0192] The crowd analyzer **58** then computes and stores new rate of change values for the remaining crowds (step **2776**). For each crowd, the rate of change of one or more characteristics of the crowd, such as population, are computed as described above. The crowd analyzer **58** then determines whether the crowd formation process for the new and old bounding boxes is done (step **2778**). In other words, the crowd analyzer **58** determines whether both the new and old bounding boxes have been processed. If not, the bounding box is set to the new bounding box (step **2780**), and the process returns to step **2746** and is repeated for the new bounding box. Once both the new and old bounding boxes have been processed, the crowd formation process ends.

[0193] FIGS. **29A** through **29D** graphically illustrate the crowd formation process of FIGS. **28A** through **28D** for a scenario where the crowd formation process is triggered by a location update for a user having no old location. In this scenario, the crowd analyzer **58** creates a new bounding box **148** for the new location of the user, and the new bounding box **148** is set as the bounding box to be processed for crowd formation. Then, as illustrated in FIG. **29A**, the crowd analyzer **58** identifies all individual users currently located within the bounding box **148** and all crowds located within or overlapping the bounding box. In this example, crowd **150** is an existing crowd relevant to the bounding box **148**. Crowds are indicated by dashed circles, crowd centers are indicated by cross-hairs (+), and users are indicated as dots. Next, as illustrated in FIG. **29B**, the crowd analyzer **58** creates crowds **152** through **156** of one user for the individual users, and the optimal inclusion distances of the crowds **152** through **156** are set to the initial optimal inclusion distance. As discussed above, the initial optimal inclusion distance is computed by the crowd analyzer **58** based on a density of users within the bounding box **148**.

[0194] The crowd analyzer **58** then identifies the two closest crowds **152** and **154** in the bounding box **148** and determines a distance between the two closest crowds **152** and **154**. In this example, the distance between the two closest crowds **152** and **154** is less than the optimal inclusion distance. As such, the two closest crowds **152** and **154** are merged and a new crowd center and new optimal inclusion distance are computed, as illustrated in FIG. **29C**. The crowd analyzer **58** then repeats the process such that the two closest crowds **152** and **156** in the bounding box **148** are again merged, as illustrated in FIG. **29D**. At this point, the distance between the two closest crowds **150** and **152** is greater than the appropriate optimal inclusion distance. As such, the crowd formation process is complete.

[0195] FIGS. **30A** through **30F** graphically illustrate the crowd formation process of FIGS. **28A** through **28D** for a scenario where the new and old bounding boxes overlap. As illustrated in FIG. **30A**, a user moves from an old location to a new location, as indicated by an arrow. The crowd analyzer **58** receives a location update for the user giving the new location of the user. In response, the crowd analyzer **58** creates an old bounding box **158** for the old location of the user and a new bounding box **160** for the new location of the user.

Crowd **162** exists in the old bounding box **158**, and crowd **164** exists in the new bounding box **160**.

[0196] Since the old bounding box **158** and the new bounding box **160** overlap, the crowd analyzer **58** creates a bounding box **166** that encompasses both the old bounding box **158** and the new bounding box **160**, as illustrated in FIG. **30B**. In addition, the crowd analyzer **58** creates crowds **168** through **174** for individual users currently located within the bounding box **166**. The optimal inclusion distances of the crowds **168** through **174** are set to the initial optimal inclusion distance computed by the crowd analyzer **58** based on the density of users in the bounding box **166**.

[0197] Next, the crowd analyzer **58** analyzes the crowds **162**, **164**, and **168** through **174** to determine whether any members of the crowds **162**, **164**, and **168** through **174** violate the optimal inclusion distances of the crowds **162**, **164**, and **168** through **174**. In this example, as a result of the user leaving the crowd **162** and moving to his new location, both of the remaining members of the crowd **162** violate the optimal inclusion distance of the crowd **162**. As such, the crowd analyzer **58** removes the remaining users from the crowd **162** and creates crowds **176** and **178** of one user each for those users, as illustrated in FIG. **30C**.

[0198] The crowd analyzer **58** then identifies the two closest crowds in the bounding box **166**, which in this example are the crowds **172** and **174**. Next, the crowd analyzer **58** computes a distance between the two crowds **172** and **174**. In this example, the distance between the two crowds **172** and **174** is less than the initial optimal inclusion distance and, as such, the two crowds **172** and **174** are combined. In this example, crowds are combined by merging the smaller crowd into the larger crowd. Since the two crowds **172** and **174** are of the same size, the crowd analyzer **58** merges the crowd **174** into the crowd **172**, as illustrated in FIG. **30D**. A new crowd center and new optimal inclusion distance are then computed for the crowd **172**.

[0199] At this point, the crowd analyzer **58** repeats the process and determines that the crowds **164** and **170** are now the two closest crowds. In this example, the distance between the two crowds **164** and **170** is less than the optimal inclusion distance of the larger of the two crowds **164** and **170**, which is the crowd **164**. As such, the crowd **170** is merged into the crowd **164** and a new crowd center and optimal inclusion distance are computed for the crowd **164**, as illustrated in FIG. **30E**. At this point, there are no two crowds closer than the optimal inclusion distance of the larger of the two crowds. As such, the crowd analyzer **58** discards any crowds having less than three members, as illustrated in FIG. **30F**. In this example, the crowds **168**, **172**, **176**, and **178** have less than three members and are therefore removed. The crowd **164** has three or more members and, as such, is not removed. At this point, the crowd formation process is complete.

[0200] FIGS. **31A** through **31E** graphically illustrate the crowd formation process of FIGS. **28A** through **28D** in a scenario where the new and old bounding boxes do not overlap. As illustrated in FIG. **31A**, in this example, the user moves from an old location to a new location. The crowd analyzer **58** creates an old bounding box **180** for the old location of the user and a new bounding box **182** for the new location of the user. Crowds **184** and **186** exist in the old bounding box **180**, and crowd **188** exists in the new bounding box **182**. In this example, since the old and new bounding

boxes **180** and **182** do not overlap, the crowd analyzer **58** processes the old and new bounding boxes **180** and **182** separately.

[0201] More specifically, as illustrated in FIG. 31B, as a result of the movement of the user from the old location to the new location, the remaining users in the crowd **184** no longer satisfy the optimal inclusion distance for the crowd **184**. As such, the remaining users in the crowd **184** are removed from the crowd **184**, and crowds **190** and **192** of one user each are created for the removed users as shown in FIG. 31C. In this example, no two crowds in the old bounding box **180** are close enough to be combined. As such, processing of the old bounding box **180** is complete, and the crowd analyzer **58** proceeds to process the new bounding box **182**.

[0202] As illustrated in FIG. 31D, processing of the new bounding box **182** begins by the crowd analyzer **58** creating a crowd **194** of one user for the user. The crowd analyzer **58** then identifies the crowds **188** and **194** as the two closest crowds in the new bounding box **182** and determines a distance between the two crowds **188** and **194**. In this example, the distance between the two crowds **188** and **194** is less than the optimal inclusion distance of the larger crowd, which is the crowd **188**. As such, the crowd analyzer **58** combines the crowds **188** and **194** by merging the crowd **194** into the crowd **188**, as illustrated in FIG. 31E. A new crowd center and new optimal inclusion distance are then computed for the crowd **188**. At this point, the crowd formation process is complete.

[0203] FIG. 32 illustrates a process for creating crowd snapshots according to one embodiment of the present disclosure. In general, in this embodiment, crowd snapshots are created and stored for only those crowds for which crowd change events are detected and which have rate of change value(s) for one or more characteristics that satisfy predetermined criteria. More specifically, after the spatial crowd formation process of FIGS. 28A through 28D is performed in response to a location update for a user, the crowd analyzer **58** detects crowd change events, if any, for the relevant crowds (step **2800**). The relevant crowds are pre-existing crowds that are within the bounding region(s) processed during the spatial crowd formation process in response to the location update for the user. The crowd analyzer **58** may detect crowd change events by comparing the crowd records of the relevant crowds before and after performing the spatial crowd formation process in response to the location update for the user. The crowd change events may be a change in the users in the crowd, a change to a location of one of the users within the crowd, or a change in the spatial information for the crowd (e.g., the NE corner, the SW corner, or the crowd center). Note that if multiple crowd change events are detected for a single crowd, then those crowd change events are preferably consolidated into a single crowd change event.

[0204] Next, the crowd analyzer **58** determines whether there are any crowd change events (step **2802**). If not, the process ends. Otherwise, the crowd analyzer **58** gets the next crowd change event (step **2804**) and obtains the rate of change value(s) and, optionally, the crowd size from the crowd record of the corresponding crowd (step **2806**). The crowd analyzer **58** then determines whether the rate of change value(s) for the crowd is greater than or equal to a predetermined minimum rate of change value(s) and, optionally, whether the crowd size of the crowd is greater than or equal to a predetermined minimum crowd size (step **2808**). The predetermined minimum rate of change may be system-defined or user-defined depending on the particular implementation. In one embodi-

ment, the predetermined minimum rate of change is static. In another embodiment, the predetermined minimum rate of change is dynamic. For example, the predetermined minimum rate of change may be defined as being a function of a predetermined minimum size such that the predetermined minimum rate of change is inversely related to the predetermined minimum size. Note that in one embodiment, the rate of change value(s) for the crowd includes a rate of change of population for the crowd where the rate of change of population is expressed as a rate of user ingress and a rate of user egress. In this case, the rate of change of population is greater than or equal to the predetermined minimum rate of change of population if both the rate of user ingress and the rate of user egress is greater than or equal to the same predetermined minimum rate of change value, if at least one of the rate of user ingress and the rate of user egress is greater than or equal to the same predetermined minimum rate of change value, if both the rate of user ingress and the rate of user egress are greater than or equal to corresponding minimum values, or if at least one of the rate of user ingress and the rate of user egress is greater than or equal to corresponding minimum values, depending on the particular implementation.

[0205] In a similar manner, the predetermined minimum size may be system-defined or user-defined depending on the particular implementation. In one embodiment, the predetermined minimum size is static. In another embodiment, the predetermined minimum size is dynamic. For example, the predetermined minimum size may be defined as being a function of the predetermined minimum rate of change such that the predetermined minimum size is inversely related to the predetermined minimum rate of change.

[0206] If the rate of change value(s) for the crowd is not greater than or equal to a predetermined minimum rate of change value(s) and, optionally, if the crowd size of the crowd is not greater than or equal to a predetermined minimum crowd size, the process proceeds to step **2812**. Otherwise, the crowd analyzer **58** creates and stores a crowd snapshot for the corresponding crowd (step **2810**). More specifically, the crowd change event identifies a crowd record stored for a crowd for which the crowd change event was detected. A crowd snapshot is then created for that crowd by creating a new crowd snapshot record for the crowd and adding the new crowd snapshot to the list of crowd snapshots stored in the crowd record for the crowd. The crowd snapshot record includes a set or list of anonymized user records, which are an anonymized version of the user records for the users in the crowd at the current time. In addition, the crowd snapshot record includes the NE corner, the SW corner, and the center of the crowd at the current time as well as a timestamp defining the current time as the sample time at which the crowd snapshot record was created. Still further, locations of users in the crowd that define the outer boundary of the crowd at the current time may be stored in the crowd snapshot record as the vertices of the crowd. Lastly, the crowd size and/or rate of change value(s) for the crowd may be stored in the crowd snapshot record. After creating and storing the crowd snapshot, the crowd analyzer **58** determines whether there are any more crowd change events (step **2812**). If so, the process returns to step **2804** and is repeated for the next crowd change event. Once all of the crowd change events are processed, the process ends.

[0207] Before proceeding, an alternative embodiment should be described. In the embodiment of FIG. 32, crowd snapshots are stored only if the rate of change values of the

one or more characteristics of the corresponding crowds and, optionally, the sizes of the corresponding crowds are greater than or equal to corresponding predetermined minimum values. However, in an alternative embodiment, crowd snapshots are stored for all crowds regardless of the rates of change values of the one or more characteristics and, optionally, sizes of the crowds, but profile data is not stored in the crowd snapshots unless the rate of change values of the one or more characteristics and, optionally, sizes of the corresponding crowds are greater than or equal to the corresponding minimum values.

[0208] FIG. 33 illustrates a process for creating crowd snapshots according to another embodiment of the present disclosure. In general, in this embodiment, crowd snapshots are created and stored for each crowd for which a crowd change event is detected regardless of the rate of change value(s) for the one or more characteristics of the crowd. In general, the process of FIG. 33 is substantially the same as that of FIG. 32 without steps 2806 and 2808. In other words, the process of FIG. 33 does not control whether to create and store a crowd snapshot based on whether the rate of change value(s) and, optionally, the crowd size of the corresponding crowd are greater than or equal to corresponding predetermined minimum values. Steps 2900-2908 of FIG. 33 are substantially the same as steps 2800-2804 and 2810-2812 of FIG. 32. Therefore, the details of those steps are not repeated herein.

[0209] FIG. 34 illustrates the operation of the MAP server 12 of FIG. 1 to serve a request for crowd tracking data for a crowd according to one embodiment of the present disclosure. In this embodiment, the MAP server 12 stores crowd snapshots for crowds only if the corresponding rate of change values are greater than or equal to a predetermined minimum rate of change in the manner described above with respect to FIG. 32. First, the subscriber device 22 sends a crowd tracking data request for a crowd to the MAP server 12 (step 3000). Note that access to crowd tracking data is preferably a subscription service only available to subscribers, such as the subscriber 24 at the subscriber device 22, for a subscription fee. However, the present disclosure is not limited thereto. The crowd tracking data request identifies a particular crowd. For example, in one embodiment, the crowd data for a number of crowds near a POI or within an AOI is presented to the subscriber 24 at the subscriber device 22 in the manner described above. The subscriber 24 may then select one of those crowds and initiate a request for crowd tracking data for the selected crowd. In response, the subscriber device 22 sends the crowd tracking data request for the selected crowd to the MAP server 12.

[0210] In response to receiving the crowd tracking data request, the MAP server 12, and more specifically the crowd analyzer 58, obtains relevant crowd snapshots for the crowd (step 3002). In one embodiment, the crowd tracking data request is a general crowd tracking data request for the crowd. As such, the relevant crowd snapshots are all crowd snapshots for the crowd. In another embodiment, the crowd tracking data request may include one or more criteria to be used to identify the relevant crowd snapshots. The one or more criteria may include time-based criteria such that only those crowd snapshots for the crowd that satisfy the time-based criteria are identified as the relevant crowd snapshots. For example, the time-based criteria may define a range of dates such as Oct. 1, 2009 through Oct. 8, 2009 or define a range of times within a particular day such as 5:00 pm through 9:00 pm on Oct. 1,

2009. The one or more criteria may additionally or alternatively include user-based criteria such that only those crowd snapshots including anonymous users satisfying the user-based criteria are identified as the relevant crowd snapshots. For example, the user-based criteria may include one or more interests and a minimum number or percentage of users such that only those crowd snapshots including at least the minimum number or percentage of anonymous users having the one or more interests are identified as the relevant crowd snapshots. Note that by using user-based criteria, the subscriber 24 is enabled to track sub-crowds within a crowd.

[0211] The crowd analyzer 58 of the MAP server 12 then generates crowd tracking data for the crowd based on the relevant crowd snapshots obtained in step 3002 (step 3004). The crowd tracking data includes data indicative of the location of the crowd over time, which can be determined based on the spatial information and sample times from the relevant crowd snapshots. In addition, the crowd tracking data may include an aggregate profile for the crowd for each of the relevant crowd snapshots or at least some of the relevant crowd snapshots, an average aggregate profile for all of the relevant crowd snapshots, an average aggregate profile for a subset of the relevant crowd snapshots, or average aggregate profiles for a number of subsets of the relevant crowd snapshots. For example, the relevant crowd snapshots may be divided into a number of time bands such that at least some of the time bands include multiple relevant crowd snapshots. An average crowd snapshot may then be created for each of the time bands. The crowd analyzer 58 may utilize the aggregation engine 60 to obtain an aggregate profile for a crowd snapshot based on the interests of the anonymous users in the crowd snapshot. More specifically, in a manner similar to that described above, an aggregate profile for a crowd snapshot may be computed by comparing the interests of the anonymous users to one another or by comparing the interests of the anonymous users to a target profile. The crowd tracking data may also contain other information derived from the relevant crowd snapshots such as, for example, the number of users in the relevant crowd snapshots, crowd characteristics for the crowd for the relevant crowd snapshots, or the like.

[0212] The crowd analyzer 58 returns the crowd tracking data for the crowd to the subscriber device 22 (step 3006). Note that in the embodiment where the subscriber device 22 interacts with the MAP server 12 via the web browser 38, the MAP server 12 returns the crowd tracking data to the subscriber device 22 in a format suitable for use by the web browser 38. For example, the crowd tracking data may be returned via a web page including a map, wherein indicators of the location of the crowd over time as defined by the relevant crowd snapshots may be overlaid upon the map. The subscriber 24 may then be enabled to select one of those indicators to view additional information regarding the crowd at that time such as, for example, an aggregate profile of a corresponding crowd snapshot of the crowd. Once the crowd tracking data is received at the subscriber device 22, the crowd tracking data is presented to the subscriber 24 (step 3008).

[0213] It should be noted that crowd snapshots stored by the MAP server 12 may additionally or alternatively be used to serve other types of requests such as, for example, a historical aggregate profile request. More specifically, in response to a historical aggregate profile request for a POI or AOI, the MAP server 12 may obtain relevant crowd snapshots for a bounding region and time window established for the historical aggregate

gate profile request in the manner described above. The profile data (e.g., the user profiles of the anonymous user records) stored in the filtered relevant crowd snapshots may then be processed to provide historical aggregate profile data in a time context or geographic context in a manner similar to that described above.

[0214] FIG. 35 illustrates the operation of the MAP server 12 of FIG. 1 to serve a request for crowd tracking data for a crowd according to another embodiment of the present disclosure. In this embodiment, the MAP server 12 stores crowd snapshots for crowds regardless of rate of change values in the manner described above with respect to FIG. 33. First, the subscriber device 22 sends a crowd tracking data request for a crowd to the MAP server 12 (step 3100). Note that access to crowd tracking data is preferably a subscription service only available to subscribers, such as the subscriber 24 at the subscriber device 22, for a subscription fee. However, the present disclosure is not limited thereto. The crowd tracking data request identifies a particular crowd. For example, in one embodiment, the crowd data for a number of crowds near a POI or within an AOI is presented to the subscriber 24 at the subscriber device 22 in the manner described above. The subscriber 24 may then select one of those crowds and initiate a request for crowd tracking data for the selected crowd. In response, the subscriber device 22 sends the crowd tracking data request for the selected crowd to the MAP server 12.

[0215] In response to receiving the crowd tracking data request, the MAP server 12, and more specifically the crowd analyzer 58, obtains relevant crowd snapshots for the crowd (step 3102). In one embodiment, the crowd tracking data request is a general crowd tracking data request for the crowd. As such, the relevant crowd snapshots are all crowd snapshots for the crowd. In another embodiment, the crowd tracking data request may include one or more criteria to be used to identify the relevant crowd snapshots. The one or more criteria may include time-based criteria such that only those crowd snapshots for the crowd that satisfy the time-based criteria are identified as the relevant crowd snapshots. For example, the time-based criteria may define a range of dates such as Oct. 1, 2009 through Oct. 8, 2009 or define a range of times within a particular day such as 5:00 pm through 9:00 pm on Oct. 1, 2009. The one or more criteria may additionally or alternatively include user-based criteria such that only those crowd snapshots including anonymous users satisfying the user-based criteria are identified as the relevant crowd snapshots. For example, the user-based criteria may include one or more interests and a minimum number or percentage of users such that only those crowd snapshots including at least the minimum number or percentage of anonymous users having the one or more interests are identified as the relevant crowd snapshots. Note that by using user-based criteria, the subscriber 24 is enabled to track sub-crowds within a crowd.

[0216] Next, the crowd analyzer 58 of the MAP server 12 filters the relevant crowd snapshots based on the rate of change values for the relevant crowd snapshots and, optionally, the crowd sizes for the relevant crowd snapshots (step 3104). More specifically, in one embodiment, the crowd analyzer 58 filters the relevant crowd snapshots to remove from the relevant crowd snapshots those crowd snapshots having rate of change values that are less than a predetermined minimum rate of change and, optionally, those crowd snapshots having crowd sizes less than a predetermined minimum crowd size. Again, the predetermined minimum rate of change may be system-defined or user-defined depending on

the particular implementation. In one embodiment, the predetermined minimum rate of change is static. In another embodiment, the predetermined minimum rate of change is dynamic. For example, the predetermined minimum rate of change may be defined as being a function of a predetermined minimum size such that the predetermined minimum rate of change is inversely related to the predetermined minimum size. Note that in one embodiment, the rate of change value(s) for the crowd includes a rate of change of population for the crowd where the rate of change of population is expressed as a rate of user ingress and a rate of user egress. In this case, the rate of change of population is greater than or equal to the predetermined minimum rate of change of population if both the rate of user ingress and the rate of user egress is greater than or equal to the same predetermined minimum rate of change value, if at least one of the rate of user ingress and the rate of user egress is greater than or equal to the same predetermined minimum rate of change value, if both the rate of user ingress and the rate of user egress are greater than or equal to corresponding minimum values, or if at least one of the rate of user ingress and the rate of user egress is greater than or equal to corresponding minimum values, depending on the particular implementation.

[0217] In a similar manner, the predetermined minimum size may be system-defined or user-defined depending on the particular implementation. In one embodiment, the predetermined minimum size is static. In another embodiment, the predetermined minimum size is dynamic. For example, the predetermined minimum size may be defined as being a function of the predetermined minimum rate of change such that the predetermined minimum size is inversely related to the predetermined minimum rate of change.

[0218] The crowd analyzer 58 of the MAP server 12 then generates crowd tracking data for the crowd based on the relevant crowd snapshots resulting from the filtering in step 3104 (step 3106). The crowd tracking data includes data indicative of the location of the crowd over time, which can be determined based on the spatial information and sample times from the relevant crowd snapshots. In addition, the crowd tracking data may include an aggregate profile for the crowd for each of the relevant crowd snapshots or at least some of the relevant crowd snapshots, an average aggregate profile for all of the relevant crowd snapshots, an average aggregate profile for a subset of the relevant crowd snapshots, or average aggregate profiles for a number of subsets of the relevant crowd snapshots. For example, the relevant crowd snapshots may be divided into a number of time bands such that at least some of the time bands include multiple relevant crowd snapshots. An average crowd snapshot may then be created for each of the time bands. The crowd analyzer 58 may utilize the aggregation engine 60 to obtain an aggregate profile for a crowd snapshot based on the interests of the anonymous users in the crowd snapshot. More specifically, in a manner similar to that described above, an aggregate profile for a crowd snapshot may be computed by comparing the interests of the anonymous users to one another or by comparing the interests of the anonymous users to a target profile. The crowd tracking data may also contain other information derived from the relevant crowd snapshots such as, for example, the number of users in the relevant crowd snapshots, crowd characteristics for the crowd for the relevant crowd snapshots, or the like.

[0219] The crowd analyzer 58 returns the crowd tracking data for the crowd to the subscriber device 22 (step 3108). Note that in the embodiment where the subscriber device 22

interacts with the MAP server 12 via the web browser 38, the MAP server 12 returns the crowd tracking data to the subscriber device 22 in a format suitable for use by the web browser 38. For example, the crowd tracking data may be returned via a web page including a map, wherein indicators of the location of the crowd over time as defined by the relevant crowd snapshots may be overlaid upon the map. The subscriber 24 may then be enabled to select one of those indicators to view additional information regarding the crowd at that time such as, for example, an aggregate profile of a corresponding crowd snapshot of the crowd. Once the crowd tracking data is received at the subscriber device 22, the crowd tracking data is presented to the subscriber 24 (step 3110).

[0220] It should be noted that crowd snapshots stored by the MAP server 12 may additionally or alternatively be used to serve other types of requests such as, for example, a historical aggregate profile request. More specifically, in response to a historical aggregate profile request for a POI or AOI, the MAP server 12 may obtain relevant crowd snapshots for a bounding region and time window established for the historical aggregate profile request in the manner described above. The relevant crowd snapshots may then be filtered to remove those crowd snapshots having rate of change values that are less than a predetermined minimum rate of change and, optionally, those having crowd sizes that are less than a predetermined minimum crowd size. The profile data (e.g., the user profiles of the anonymous user records) stored in the filtered relevant crowd snapshots may then be processed to provide historical aggregate profile data in a time context or geographic context in a manner similar to that described above.

[0221] It should also be noted that the process of FIGS. 34 and 35 could also be used to serve requests from users and devices other than the subscriber device 22 and the subscriber 24. More specifically, the MAP server 12 may also use the process of FIG. 34 or 35 to serve requests from the users 20-1 through 20-N at the mobile devices 18-1 through 18-N, requests automatically made by the mobile devices 18-1 through 18-N, the third-party service 26, or the like.

[0222] Thus far, different embodiments have been described to control access to aggregate profile data for groups of users at the MAP server 12. FIG. 36 is a flow chart for a process for controlling access to aggregate profile data by controlling whether to provide user profiles of the users 20-1 through 20-N to the MAP server 12 based on rate of change values for current crowds at or near the locations of the users 20-1 through 20-N according to another embodiment of the present disclosure. First, the MAP application 32-1, or alternatively the MAP client 30-1, determines a current location of the mobile device 18-1, which is the current location of the user 20-1 (step 3200). The MAP application 32-1 then obtains rate of change value(s) of one or more characteristics of a crowd relevant to the current location of the user 20-1 and, optionally, a crowd size of that crowd from the MAP server 12 (step 3202). In one embodiment, the relevant crowd is the crowd in which the user 20-1 is, or will be, included based on the current location of the user 20-1. The MAP application 32-1 then determines whether the one or more rate of change values are greater than or equal to one or more predetermined minimum rate of change values and, optionally, whether the crowd size is greater than or equal to a predetermined minimum crowd size in the manner described above (step 3204). As discussed above, the minimum rate of change values may be system-defined or user-

defined and may be static or dynamic. Likewise, the minimum crowd size may be system-defined or user-defined and may be static or dynamic.

[0223] If the rate of change value(s) of the one or more characteristics of the relevant crowd and, optionally, the crowd size of the relevant crowd are greater than or equal to the corresponding predetermined minimum values, then the MAP application 32-1 enables the user profile of the user 20-1 at the MAP server 12 (step 3206). In one embodiment, the MAP application 32-1 may enable the user profile of the user 20-1 at the MAP server 12 by providing the user profile to the MAP server 12 if the user profile is not already enabled at the MAP server 12 or providing credentials to the MAP server 12 that enable the MAP server 12 to access the user profile of the user 20-1 from the profile server 14. However, other techniques for enabling the user profile of the user 20-1 at the MAP server 12 may be used.

[0224] If the rate of change value(s) of the one or more characteristics of the relevant crowd and, optionally, the crowd size of the relevant crowd are not greater than or equal to the corresponding predetermined minimum values, then the MAP application 32-1 disables the user profile of the user 20-1 at the MAP server 12 (step 3208). In one embodiment, the MAP application 32-1 may disable the user profile of the user 20-1 at the MAP server 12 by providing an empty, or null, user profile to the MAP server 12 as the user profile of the user 20-1. However, other techniques for disabling the user profile of the user 20-1 at the MAP server 12 may be used.

[0225] At this point, whether proceeding from step 3206 or 3208, the MAP application 32-1 reports the current location to the MAP server 12 as the current location of the user 20-1 (step 3210). In this manner, the MAP server 12 is enabled to form and track crowds in the manner described above while at the same time the MAP applications 32-1 through 32-N of the mobile devices 18-1 through 18-N enable and disable the user profiles of the users 20-1 through 20-N based on the rate of change values for corresponding crowds to thereby control access to aggregate profile data for the crowds at the MAP server 12. The MAP application 32-1 then waits a predetermined amount of time (step 3212) and then the process returns to step 3200.

[0226] FIG. 37 is a block diagram of the MAP server 12 according to one embodiment of the present disclosure. As illustrated, the MAP server 12 includes a controller 196 connected to memory 198, one or more secondary storage devices 200, and a communication interface 202 by a bus 204 or similar mechanism. The controller 196 is a microprocessor, digital Application Specific Integrated Circuit (ASIC), Field Programmable Gate Array (FPGA), or the like. In this embodiment, the controller 196 is a microprocessor, and the application layer 40, the business logic layer 42, and the object mapping layer 62 (FIG. 2) are implemented in software and stored in the memory 198 for execution by the controller 196. Further, the datastore 64 (FIG. 2) may be implemented in the one or more secondary storage devices 200. The secondary storage devices 200 are digital data storage devices such as, for example, one or more hard disk drives. The communication interface 202 is a wired or wireless communication interface that communicatively couples the MAP server 12 to the network 28 (FIG. 1). For example, the communication interface 202 may be an Ethernet interface, local wireless interface such as a wireless interface operating according to one of the suite of IEEE 802.11 standards, or the like.

[0227] FIG. 38 is a block diagram of the mobile device 18-1 according to one embodiment of the present disclosure. This discussion is equally applicable to the other mobile devices 18-2 through 18-N. As illustrated, the mobile device 18-1 includes a controller 206 connected to memory 208, a communication interface 210, one or more user interface components 212, and the location function 36-1 by a bus 214 or similar mechanism. The controller 206 is a microprocessor, digital ASIC, FPGA, or the like. In this embodiment, the controller 206 is a microprocessor, and the MAP client 30-1, the MAP application 32-1, and the third-party applications 34-1 are implemented in software and stored in the memory 208 for execution by the controller 206. In this embodiment, the location function 36-1 is a hardware component such as, for example, a GPS receiver. The communication interface 210 is a wireless communication interface that communicatively couples the mobile device 18-1 to the network 28 (FIG. 1). For example, the communication interface 210 may be a local wireless interface such as a wireless interface operating according to one of the suite of IEEE 802.11 standards, a mobile communications interface such as a cellular telecommunications interface, or the like. The one or more user interface components 212 include, for example, a touchscreen, a display, one or more user input components (e.g., a keypad), a speaker, or the like, or any combination thereof.

[0228] FIG. 39 is a block diagram of the subscriber device 22 according to one embodiment of the present disclosure. As illustrated, the subscriber device 22 includes a controller 216 connected to memory 218, one or more secondary storage devices 220, a communication interface 222, and one or more user interface components 224 by a bus 226 or similar mechanism. The controller 216 is a microprocessor, digital ASIC, FPGA, or the like. In this embodiment, the controller 216 is a microprocessor, and the web browser 38 (FIG. 1) is implemented in software and stored in the memory 218 for execution by the controller 216. The one or more secondary storage devices 220 are digital storage devices such as, for example, one or more hard disk drives. The communication interface 222 is a wired or wireless communication interface that communicatively couples the subscriber device 22 to the network 28 (FIG. 1). For example, the communication interface 222 may be an Ethernet interface, local wireless interface such as a wireless interface operating according to one of the suite of IEEE 802.11 standards, a mobile communications interface such as a cellular telecommunications interface, or the like. The one or more user interface components 224 include, for example, a touchscreen, a display, one or more user input components (e.g., a keypad), a speaker, or the like, or any combination thereof.

[0229] FIG. 40 is a block diagram of a computing device 228 operating to host the third-party service 26 according to one embodiment of the present disclosure. The computing device 228 may be, for example, a physical server. As illustrated, the computing device 228 includes a controller 230 connected to memory 232, one or more secondary storage devices 234, a communication interface 236, and one or more user interface components 238 by a bus 240 or similar mechanism. The controller 230 is a microprocessor, digital ASIC, FPGA, or the like. In this embodiment, the controller 230 is a microprocessor, and the third-party service 26 is implemented in software and stored in the memory 232 for execution by the controller 230. The one or more secondary storage devices 234 are digital storage devices such as, for example, one or more hard disk drives. The communication interface

236 is a wired or wireless communication interface that communicatively couples the computing device 228 to the network 28 (FIG. 1). For example, the communication interface 236 may be an Ethernet interface, local wireless interface such as a wireless interface operating according to one of the suite of IEEE 802.11 standards, a mobile communications interface such as a cellular telecommunications interface, or the like. The one or more user interface components 238 include, for example, a touchscreen, a display, one or more user input components (e.g., a keypad), a speaker, or the like, or any combination thereof.

[0230] Those skilled in the art will recognize improvements and modifications to the preferred embodiments of the present invention. All such improvements and modifications are considered within the scope of the concepts disclosed herein and the claims that follow.

What is claimed is:

1. A computer-implemented method comprising:

for each group of users of one or more groups of users, determining a rate of change of one or more characteristics of the group of users; and

controlling access to aggregate profile data for the one or more groups of users based on the rates of change of the one or more characteristics of the group of users.

2. The method of claim 1 wherein the rate of change of the one or more characteristics of the group of users comprises a rate of change of a population of the group of users.

3. The method of claim 2 wherein the rate of change of the population of the group of users comprises a rate of user ingress into the group of users.

4. The method of claim 2 wherein the rate of change of the population of the group of users comprises a rate of user egress from the group of users.

5. The method of claim 2 wherein the rate of change of the population of the group of users comprises a rate of user ingress into the group of users and a rate of user egress from the group of users.

6. The method of claim 2 wherein the rate of change of the population of the group of users is a net rate of change of the population of the group of users.

7. The method of claim 1 wherein the rate of change of the one or more characteristics of the group of users comprises a rate of change of a component of an aggregate profile of the group of users.

8. The method of claim 7 wherein the component of the aggregate profile of the group of users is a number of occurrences of a user interest included in the aggregate profile of the group of users.

9. The method of claim 7 wherein the component of the aggregate profile of the group of users is a ratio of a number of occurrences of a user interest included in the aggregate profile of the group of users to a total number of users in the group of users.

10. The method of claim 1 wherein the one or more groups of users are one or more crowds of users formed via a spatial crowd formation process, and, for each crowd of users, determining the rate of change of the one or more characteristics comprising determining the rate of change of the one or more characteristics of the crowd of users.

11. The method of claim 10 wherein controlling access to the aggregate profile data comprises, for each crowd of users of the one or more crowds of users, controlling storage of

crowd snapshots of the crowd of users based on the rate of change of the one or more characteristics of the crowd of users.

12. The method of claim 11 wherein controlling storage of the crowd snapshots of the crowd of users based on the rate of change of the one or more characteristics of the crowd of users comprises, at a predefined time interval, creating and storing a crowd snapshot of the crowd of users if the rate of change of the one or more characteristics are greater than one or more corresponding predetermined minimum rate of change values.

13. The method of claim 11 wherein controlling storage of the crowd snapshots of the crowd of users based on the rate of change of the one or more characteristics of the crowd of users comprises, at a predefined time interval, creating and storing a crowd snapshot of the crowd of users if the rate of change of the one or more characteristics and a crowd size of the crowd of users are greater than corresponding predetermined minimum values.

14. The method of claim 11 wherein controlling storage of the crowd snapshots of the crowd of users based on the rate of change of the one or more characteristics of the crowd of users comprises, at a predefined time interval, creating and storing a crowd snapshot of the crowd of users without profile data if the rate of change of the one or more characteristics is not greater than one or more corresponding predetermined minimum rate of change values.

15. The method of claim 11 wherein controlling storage of the crowd snapshots of the crowd of users based on the rate of change of the one or more characteristics of the crowd of users comprises, at a predefined time interval, creating and storing a crowd snapshot of the crowd of users without profile data if the rate of change of the one or more characteristics and a crowd size of the crowd of users are not greater than corresponding predetermined minimum values.

16. The method of claim 10 wherein controlling access to the aggregate profile data comprises:

receiving a request from a requestor;

identifying a plurality of relevant crowd snapshots created and stored for the one or more crowds of users over time that are relevant to the request, each crowd snapshot comprising the rate of change of the one or more characteristics of a corresponding crowd of users of the one or more crowds of users at a time at which the crowd snapshot was created and stored and profile data for users in the corresponding crowd of users at the time at which the crowd snapshot was created and stored;

filtering the plurality of relevant crowd snapshots such that one or more crowd snapshots of the plurality of relevant crowd snapshots having rates of change of the one or more characteristics that are not greater than or equal to one or more corresponding predetermined minimum values are removed from the plurality of relevant crowd snapshots to provide one or more filtered crowd snapshots;

generating data in response to the request based on the one or more filtered crowd snapshots; and

returning the data to the requestor.

17. The method of claim 10 wherein controlling access to the aggregate profile data comprises:

receiving a crowd tracking request from a requestor for a crowd of users from the one or more crowds of users;

identifying a plurality of relevant crowd snapshots created and stored for the crowd of users over time that are

relevant to the crowd tracking request, each crowd snapshot comprising the rate of change of the one or more characteristics of the crowd of users at a time at which the crowd snapshot was created and stored and profile data for users in the crowd of users at the time at which the crowd snapshot was created and stored;

filtering the plurality of relevant crowd snapshots such that one or more crowd snapshots of the plurality of relevant crowd snapshots having rates of change of the one or more characteristics that are not greater than or equal to one or more corresponding predetermined minimum values are removed from the plurality of relevant crowd snapshots to provide one or more filtered crowd snapshots;

generating crowd tracking data in response to the crowd tracking request based on the one or more filtered crowd snapshots; and

returning the crowd tracking data to the requestor.

18. The method of claim 10 wherein controlling access to the aggregate profile data comprises:

receiving a historical request from a requestor;

identifying a plurality of relevant crowd snapshots created and stored for the one or more crowds of users over time that are relevant to the historical request, each crowd snapshot comprising the rate of change of the one or more characteristics of a corresponding crowd of users of the one or more crowds of users at a time at which the crowd snapshot was created and stored and profile data for users in the corresponding crowd of users at the time at which the crowd snapshot was created and stored;

filtering the plurality of relevant crowd snapshots such that one or more crowd snapshots of the plurality of relevant crowd snapshots having rates of change of the one or more characteristics that are not greater than or equal to one or more corresponding predetermined minimum values are removed from the plurality of relevant crowd snapshots to provide one or more filtered crowd snapshots;

generating historical aggregate profile data in response to the historical request based on the one or more filtered crowd snapshots; and

returning the historical aggregate profile data to the requestor.

19. The method of claim 1 wherein each group of users of the one or more groups of users is a group of users in a geographic area during a corresponding time period.

20. The method of claim 19 wherein a geographic region is divided into a plurality of static geographic areas, and for each group of users of the one or more groups of users, the geographic area in which the group of users was located during the corresponding time period is one or more adjacent static geographic areas of the plurality of static geographic areas.

21. The method of claim 19 wherein a geographic region is divided into a plurality of static geographic areas, and for each group of users of the one or more groups of users, the geographic area in which the group of users was located during the corresponding time period is one of the plurality of static geographic areas.

22. The method of claim 19 further comprising:

maintaining a historical record of anonymized user profile data by location, the historical record comprising a plurality of history objects each stored for a corresponding geographic area for a corresponding period of time and comprising aggregate profile data for a group of users

located within the corresponding geographic area during the corresponding period of time;

wherein controlling access to the aggregate profile data for the one or more groups of users based on the rates of change of the one or more characteristics of the group of users comprises, for each group of users controlling storage of a history object for the group of users in the historical record of anonymized user profile data based on the rate of change of the one or more characteristics for the group of users.

23. The method of claim **22** wherein controlling the storage of the history object for the group of users in the historical record of anonymized user profile data based on the rate of change of the one or more characteristics for the group of users comprises storing a history object for the group of users in the historical record of anonymized user profile data if the rate of change of the one or more characteristics of the group of users is greater than or equal to one or more corresponding predetermined minimum rate of change values.

24. The method of claim **22** wherein controlling the storage of the history object for the group of users in the historical record of anonymized user profile data based on the rate of change of the one or more characteristics for the group of users comprises storing a history object for the group of users in the historical record of anonymized user profile data if the rate of change of the one or more characteristics of the group of users and a size of the group of users are greater than or equal to corresponding predetermined minimum values.

25. The method of claim **22** wherein controlling the storage of the history object for the group of users in the historical record of anonymized user profile data based on the rate of change of the one or more characteristics for the group of users comprises storing a history object for the group of users in the historical record of anonymized user profile data without profile data if the rate of change of the one or more characteristics of the group of users is not greater than or equal to one or more corresponding predetermined minimum rate of change values.

26. The method of claim **22** wherein controlling the storage of the history object for the group of users in the historical record of anonymized user profile data based on the rate of change of the one or more characteristics for the group of users comprises storing a history object for the group of users in the historical record of anonymized user profile data without profile data if the rate of change of the one or more characteristics of the group of users and a size of the group of users are not greater than or equal to corresponding predetermined minimum values.

27. The method of claim **19** wherein controlling access to the aggregate profile data comprises:

receiving a historical request from a requestor;

identifying a plurality of history objects created and stored for the one or more groups of users that are relevant to the historical request, each history object comprising the rate of change of the one or more characteristics of a corresponding group of users from the one or more groups of users and profile data for users in the corresponding group of users;

filtering the plurality of history objects such that one or more history objects of the plurality of history objects having rates of change of the one or more characteristics that are not greater than or equal to one or more corresponding predetermined minimum values are removed

from the plurality of history objects to provide one or more filtered history objects;

generating historical aggregate profile data in response to the historical request based on the profile data stored in the one or more filtered history objects; and
returning the historical aggregate profile data to the requestor.

28. The method of claim **1** wherein controlling access to the aggregate profile data for the one or more groups of users comprises, for each group of users of the one or more groups of users, enabling access to the aggregate profile data for the group of users if the rate of change of the one or more characteristics of the group of users is greater than or equal to one or more corresponding predetermined minimum rate of change values.

29. The method of claim **28** wherein the one or more corresponding predetermined minimum rate of change values are static values.

30. The method of claim **28** wherein the one or more corresponding predetermined minimum rate of change values are dynamic values.

31. The method of claim **1** wherein controlling access to the aggregate profile data for the one or more groups of users comprises, for each group of users of the one or more groups of users, enabling access to the aggregate profile data for the group of users if the rate of change of the one or more characteristics of the group of users is greater than or equal to one or more corresponding predetermined minimum rate of change values and a size of the group of users is greater than or equal to a predetermined minimum size.

32. The method of claim **31** wherein the one or more corresponding predetermined minimum rate of change values and the predetermined minimum size are dynamic values that are inversely related to one another.

33. A computer-readable medium storing software for instructing a controller of a computing device to:

for each group of users of one or more groups of users, determine a rate of change of one or more characteristics of the group of users; and

control access to aggregate profile data for the one or more groups of users based on the rates of change of the one or more characteristics of the group of users.

34. A server comprising:

a communication interface communicatively coupling the server to a network; and

a controller associated with the communication interface and adapted to:

for each group of users of one or more groups of users, determine a rate of change of one or more characteristics of the group of users; and

control access to aggregate profile data for the one or more groups of users based on the rates of change of the one or more characteristics of the group of users.

35. A method of operation of a mobile device comprising: determining a current location of the mobile device;

obtaining a rate of change of one or more characteristics of a relevant crowd of users that is relevant to the current location from a mobile aggregate profile server that operates to form crowds of users and provide access to aggregate profile data for the crowds of users;

determining whether to enable access to a user profile of a user of the mobile device at the mobile aggregate profile server based on the rate of change of the one or more characteristics of the relevant crowd of users; and

enabling access to the user profile of the user of the mobile device at the mobile aggregate profile server if a determination is made to enable access to the user profile of the user of the mobile device at the mobile aggregate profile server.

36. A computer-readable medium storing software for instructing a controller of a mobile device to:

- determine a current location of the mobile device;
- obtain a rate of change of one or more characteristics of a relevant crowd of users that is relevant to the current location from a mobile aggregate profile server that operates to form crowds of users and provide access to aggregate profile data for the crowds of users;
- determine whether to enable access to a user profile of a user of the mobile device at the mobile aggregate profile server based on the rate of change of the one or more characteristics of the relevant crowd of users; and
- enable access to the user profile of the user of the mobile device at the mobile aggregate profile server if a determination is made to enable access to the user profile of the user of the mobile device at the mobile aggregate profile server.

37. A server comprising:

- a communication interface communicatively coupling the server to a network; and
- a controller associated with the communication interface and adapted to:
 - determine a current location of a mobile device;
 - obtain a rate of change of one or more characteristics of a relevant crowd of users that is relevant to the current location from a mobile aggregate profile server that operates to form crowds of users and provide access to aggregate profile data for the crowds of users;
 - determine whether to enable access to a user profile of a user of the mobile device at the mobile aggregate profile server based on the rate of change of the one or more characteristics of the relevant crowd of users; and
 - enable access to the user profile of the user of the mobile device at the mobile aggregate profile server if a determination is made to enable access to the user profile of the user of the mobile device at the mobile aggregate profile server.

* * * * *