(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 February 2007 (08.02.2007)

PCT

(10) International Publication Number
WO 2007/016040 A2

(51) International Patent Classification:
G06F 17/00 (2006.01)

(21) International Application Number:
PCT/US2006/028709

(22) International Filing Date: 25 July 2006 (25.07.2006)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/190,179 26 July 2005 (26.07.2005) US

(71) Applicant *(for all designated States except US)*: **Invensys Systems, Inc.** [US/US]; 33 Commercial Street, Foxboro, Massachusetts 02035 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **JENSEN, Niels** [DK/US]; 38 Frontier Street, Trabuco Canyon, California 92679 (US). **MIDDLETON, Elliott S.** [—/US]; 26806 Belleza Circle, Mission Viejo, California 92692 (US). **VICTOR, Hendrik Johannes** [ZA/US]; 31 Ammolite Street, Rancho Santa Margarita, California 92688 (US). **KANE, Douglas** [US/US]; 29631 Silverado Canyon Road, Silverado, California 92676 (US).

(74) Agent: **JOY, Mark**; Leydig, Voit & Mayer, Ltd., Two Prudential Plaza, Suite 4900, Chicago, Illinois 60601 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) **Title:** SYSTEM AND METHOD FOR RETRIEVING INFORMATION FROM A SUPERVISORY CONTROL MANUFACTURING /PRODUCTION DATABASE

(57) **Abstract:** A database server for handling steams of time stamped data points for tagged variables is disclosed herein that supports a set of advanced data retrieval operations/queries invoked by clients of the database server. The advanced data retrieval operations are invoked by client queries to provide, on demand, secondary information by processing previously tabled data corresponding to received data streams rendered by a variety of data sources in a supervisory control/monitoring, process control and/or automated equipment environment. Calculations, on previously stored data, for rendering the secondary information are performed within the database server at the time the secondary information is requested by a client of the historian that maintains a database containing the previously stored data.

WO 2007/016040 A2

SYSTEM AND METHOD FOR RETRIEVING INFORMATION FROM A SUPERVISORY CONTROL MANUFACTURING/PRODUCTION DATABASE

TECHNICAL FIELD

The present invention generally relates to computing and networked data storage systems, and, more particularly, to techniques for managing (e.g., storing, retrieving, processing, etc.) streams of supervisory control, manufacturing, and production information. Such information is typically rendered and stored in the context of supervising automated processes and/or equipment. The data is thereafter accessed by a variety of database clients such as, for example, by trending applications.

BACKGROUND

Industry increasingly depends upon highly automated data acquisition and control systems to ensure that industrial processes are run efficiently and reliably while lowering their overall production costs. Data acquisition begins when a number of sensors measure aspects of an industrial process and report their measurements back to a data collection and control system. Such measurements come in a wide variety of forms. By way of example the measurements produced by a sensor/recorder include: a temperature, a pressure, a pH, a mass/volume flow of material, a counter of items passing through a particular machine/process, a tallied inventory of packages waiting in a shipping line, cycle completions, etc. Often sophisticated process management and control software examines the incoming data associated with an industrial process, produces status reports and operation summaries, and, in many cases, responds to events/operator instructions by sending commands to actuators/controllers that modify operation of at least a portion of the industrial process. The data produced by the sensors also allow an operator to perform a number of supervisory tasks including: tailor the process (e.g., specify new set points) in response to varying external conditions (including costs of raw materials), detect an inefficient/non-optimal operating condition and/or impending equipment failure, and take remedial action such as move equipment into and out of service as required.

A very simple and familiar example of a data acquisition and control system is a thermostat-controlled home heating/air conditioning system. A thermometer measures a current

temperature, the measurement is compared with a desired temperature range, and, if necessary, commands are sent to a furnace or cooling unit to achieve a desired temperature. Furthermore, a user can program/manually set the controller to have particular setpoint temperatures at certain time intervals of the day.

Typical industrial processes are substantially more complex than the above-described simple thermostat example. In fact, it is not unheard of to have thousands or even tens of thousands of sensors and control elements (e.g., valve actuators) monitoring/controlling all aspects of a multi-stage process within an industrial plant. The amount of data sent for each measurement and the frequency of the measurements varies from sensor to sensor in a system. For accuracy and to facilitate quick notice/response of plant events/upset conditions, some of these sensors update/transmit their measurements several times every second. When multiplied by thousands of sensors/control elements, the volume of data generated by a plant's supervisory process control and plant information system can be very large.

Specialized process control and manufacturing/production information data storage facilities (also referred to as plant historians) have been developed to handle the potentially massive amounts of process/production information generated by the aforementioned systems. An example of such system is the WONDERWARE IndustrialSQL Server historian. A data acquisition service associated with the historian collects time series data from a variety of data sources (e.g., data access servers). The collected data is thereafter deposited with the historian to achieve data access efficiency and querying benefits/capabilities of the historian's relational database. Through its relational database, the historian integrates plant data with event, summary, production and configuration information.

Traditionally, plant databases, referred to as historians have collected and stored in an organized manner to facilitate efficient retrieval by a database server (i.e., "tabled") streams of time stamped data representing process/plant/production status over the course of time. The status data is of value for purposes of maintaining a record of plant performance and presenting/recreating the state of a process or plant equipment at a particular point in time. However, individual pieces of data taken at single points in time are often insufficient to discern whether an industrial process is operating properly/optimally. Further processing of the

previously tabled streams of time stamped data often renders more useful "secondary" information for engineers, rendering reports, and even operator decision-making. Over the course of time, even in relatively simple systems, Terabytes of the steaming time stamped information are generated by the system and tabled by the historian.

The tabled information is thereafter retrieved from the tables of historians and displayed by a variety of historian database client applications including trending and analytical applications at a supervisory level of an industrial process control system/enterprise. Such applications include displays for presenting/recreating the changing state of an industrial process or plant equipment at any particular point (or series of points) in time. A specific example of such client application is the WONDERWARE ActiveFactory trending and analysis application. This trending and analysis application provides a flexible set of display and analytical tools for accessing, visualizing and analyzing plant performance/status information.

Over the years vast improvements have occurred with regard to networks, data storage and processor device capacity and processing speeds. Notwithstanding such improvements, supervisory process control and manufacturing information system designs encounter a need to either increase system capacity/speed or forgo saving certain types of information derived from previously received/tabled streams of time stamped data because creating/maintaining the many types of derived information on a full-time basis draws too heavily from available storage/processor resources. Thus, while valuable, certain types of derived information are potentially not available in certain environments due to data storage capacity and/or processor limitations. Such choices can arise, for example, in large plant/production systems wherein processing the streams of time stamped data to render secondary information is potentially of greatest value yet very costly from the standpoint of creating and/or storing the secondary information.

SUMMARY OF THE INVENTION

The present invention comprises a system and method for rendering certain types of secondary information by processing data streams rendered by a variety of data sources in a supervisory control/monitoring, process control and/or automated equipment environment. Calculations, on previously tabled data, for rendering the secondary information are performed within a database server (historian) at the time the secondary information is requested by a client of the historian that maintains a database containing the previously tabled data. By performing, by the historian, the step of creating the secondary information on demand, substantial plant historian resource savings (e.g., storage space, processor cycles) are potentially realized in comparison to a system wherein the secondary information is created on all received data for the particular types of secondary information identified below. Furthermore, by processing the received/tabled data on-demand, the calculations can be flexibly tuned for a particular purpose (through a set of output tuning parameters submitted with a request invoking a particular advanced data retrieval operation).

The historian supports an extensible set of advanced data retrieval operations. For example, an engineering units-based integral data retrieval operation transparently converts a rate to a quantity and returns the quantity to a user. Another advanced data retrieval operation is derivative/slope data retrieval operation that returns rate change values. Yet another advanced data retrieval operations includes a counter data retrieval operation that automatically handles counter rollover. Another example of an advanced data retrieval operation incorporated within the historian includes a time-in-state data retrieval operation.

The extensible nature of the historian's advanced data retrieval set ensures that as additional needs are identified, new advanced data retrieval operations are developed and incorporated within the historian's infrastructure. Client's need only specify the new advanced operations with appropriate options specified.

BRIEF DESCRIPTION OF THE DRAWINGS

While the appended claims set forth the features of the present invention with particularity, the invention, together with its objects and advantages, may be best understood from the following detailed description taken in conjunction with the accompanying drawings of which:

FIG. 1 is a schematic diagram of an exemplary networked environment wherein a process control database server embodying the present invention is advantageously incorporated;

FIG. 2 is a schematic drawing of functional/structural aspects of a historian server/service embodying the present invention;

FIG. 3 is a set of advanced data retrieval operations supported by a database server embodying the present invention;

FIGs. 4a and 4b graphically depict stair-step and interpolated data retrieval processing;

FIG. 5 depicts an exemplary time-sequenced set of data point values used to illustrate a derivative/slope operation;

FIG. 6 depicts an exemplary time-sequenced set of data point values used to illustrate a counter operation;

FIG. 7 depicts an illustrative sequence of data points for the purpose of demonstrating an interpolated retrieval operation;

FIG. 8 depicts an illustrative sequence of data points for the purpose of demonstrating a best fit retrieval operation;

FIG. 9 depicts an illustrative sequence of data points for the purpose of demonstrating a time weighted averaging retrieval operation;

FIG. 10 depicts an illustrative sequence of data points for the purpose of demonstrating a minimum with time retrieval operation; and

FIG. 11 depicts an illustrative sequence of data points for the purpose of demonstrating a maximum with time retrieval operation; and

FIG. 12 is a flow diagram depicting the general steps performed to carry out advanced data retrieval operations.

DETAILED DESCRIPTION OF THE DRAWINGS

As noted previously in the background, plant information historian servers/services maintain a database comprising a wide variety of plant status information. The plant status information, when provided to operations managers in its unprocessed form, offers limited comparative information – such has how a process or the operation of plant equipment has changed over time. In many cases, performing additional analysis on received/tabled data streams to render secondary information greatly enhances the information value of the received/tabled data. In embodiments of the invention, such analysis is delayed until a client requests such secondary information from the historian service for a particular timeframe. As such, limited historian memory/processor resources are only allocated to the extent a client of the historian service has requested the secondary information. In particular, the historian service supports a set of advanced data retrieval operations wherein received/tabled data is processed to render particular types of secondary information "on demand" and in response to "client requests."

The term "tabled" is used herein to describe data, received by a database server/historian, stored in an organized manner to facilitate efficient retrieval by the database server.

The terms "client requests" and "on demand" are intended to be broadly defined. The process/plant historian service embodying the present invention does not distinguish between requests arising from human users and requests originating from automated processes. Thus, a "client request", unless specifically noted, includes requests initiated by human machine interface users and requests initiated by automated client processes. The automated client processes potentially include processes running on the same node as the historian service. The automated client processes request the secondary information and thereafter provide the received secondary information, in a service role, to others. Furthermore, the definition of "on demand" is intended to include both providing secondary information in response to specific requests as well as in accordance with a previously established subscription. By performing the calculations to render the secondary information on demand, rather than calculating (and tabling) them without regard to whether they will ever be requested by a client, the historian

system embodying the present invention is better suited to support a very broad/extensible set of secondary information types meeting diverse needs of a broad variety of historian service clients.

In an embodiment of the present invention, the historian service supports a variety of advanced retrieval operations for calculating and providing, on demand, a variety of secondary information types from data previously tabled in the historian database. Among others, the historian service specifically includes the following advanced data retrieval operations: "time-in-state", "counter", "engineering units-based integral", and "derivative". "Time-in-state" calculations render statistical information relating to an amount of time spent in specified states. Such states are represented, for example, by identified tag/value combinations. By way of example the time-in-state statistics include, for a specified time span and tagged state value: total amount of time in the state, percentage of time in the state, the average time in state, the shortest time in the state, and the longest time in the state.

With regard to the "counter" advanced data retrieval operation, it is noted that some instance counters "rollover" (i.e., return to zero) after reaching a particular count value. For example, a 4-digit decimal integer counter counts from zero to 9999 before rolling over to a zero value. The counter advanced data retrieval operation operates upon stored counter data to convert unprocessed counter readings into a meaningful summary of the amount of increase measured by the counter (whether real or integer) over time, factoring in any rollover and inferring rollover even if a rollover value (e.g., a rollover counter) itself is not directly sampled.

With regard to the "engineering units-based integral" advanced data retrieval operation, instantaneous measurement data is sampled and processed over a user-specified time period. Rather than use a fixed time unit (e.g., seconds only), the EU-based integral retrieval operation uses the time unit specified by the tabled data samples (e.g., liters/minute, liters/second, etc.) to render a quantity for the specified time period. The "derivative" advanced data retrieval operation involves calculating estimates of the instantaneous rate of change for a specified time span to render a time series sequence of data values reflecting the dynamic (i.e., changing) aspect of a particular received/tabled data stream. Each of the above advanced retrieval modes

is described in detail herein below in association with an exemplary system including a historian server/service incorporating the above-identified advanced data retrieval operations.

The following description is based on illustrative embodiments of the invention and should not be taken as limiting the invention with regard to alternative embodiments that are not explicitly described herein. Those skilled in the art will readily appreciate that the illustrative example in **FIG. 1** represents a simplified configuration used for illustrative purposes. In particular, the systems within which the present invention is incorporated are substantially larger and the breadth of network connections to client applications greater (including clients that access the historian via an Internet portal server). While the illustrative network arrangement depicts a local area network connection between a historian and a set relatively small number of data sources. In many instances, the number of data sources is several times larger – resulting in massive quantities of time-series process data associated with potentially hundreds and even thousands of data points in a process control system. Notwithstanding the recent improvements in secondary storage capacity, reducing the quantity of data by reducing the types of data stored, thereby reducing the size of files associated with tabled process data points, potentially improves the performance of the historian and its clients.

FIGURE 1 schematically depicts an illustrative environment wherein a supervisory process control and manufacturing/production information data storage facility (also referred to as a plant historian) 100 embodying the present invention is potentially incorporated. The historian 100 includes database services for maintaining and providing a variety of plant/process/production information including historical plant status, configuration, event, and summary information.

The network environment includes a plant floor network 101 to which a set of process control and manufacturing information data sources 102 are connected either directly or indirectly (via any of a variety of networked devices including concentrators, gateways, integrators, interfaces, etc.).

While FIG. 1 illustratively depicts the data sources 102 as a set of PLCs(1-N), the data sources 102 comprise any of a wide variety of data sources (and combinations thereof) including, for example, programmable logic controllers (PLCs), input/output modules, and

distributed control systems (DCSs). The data sources 102, in turn, are coupled to, communicate with, and control a variety of devices such as plant floor equipment, sensors, and actuators. Data received from the data sources 102 potentially represents, for example, discrete data such as states, counters, events, etc. and analog process data such as temperatures, tank levels/pressures, volume flow, etc. A set of I/O servers 104, for example DATA ACCESS SERVERS developed and provided by WONDERWARE, acquire data from the data sources 102 via the plant floor network 101 on behalf of a variety of potential clients/subscribers – including the historian 100.

The exemplary network environment includes a production network 110. In the illustrative embodiment the production network 110 comprises a set of client application nodes 112 that execute, by way of example, trending applications that receive and graphically display time-series values for a set of data points. One example of a trending application is WONDERWARE'S ACTIVE FACTORY application software. The data driving the trending applications on the nodes 112 is acquired, by way of example, from the plant historian 100 that also resides on the production network 110. The historian 100 includes database services for maintaining and providing a variety of plant/process/production information including historical plant status, configuration, event, and summary information.

A data acquisition service 116, for example WONDERWARE'S REMOTE IDAS, interposed between the I/O servers 104 and the plant historian 100 operates to maintain a continuous, up-to-date, flow of streaming plant data between the data sources 102 and the historian 100 for plant/production supervisors (both human and automated). The data acquisition service 116 acquires and integrates data (potentially in a variety of forms associated with various protocols) from a variety of sources into a plant information database, including time stamped data entries, incorporated within the historian 100.

The physical connection between the data acquisition service 116 and the I/O servers 104 can take any of a number of forms. For example, the data acquisition service 116 and the I/O servers 104 can comprise distinct nodes on a same network (e.g., the plant floor network 110). However, in alternative embodiments the I/O servers 104 communicate with the data acquisition service 116 via a network link that is separate and distinct from the plant floor

network 101. In an illustrative example, the physical network links between the I/O servers 104 and the data acquisition service 116 comprise local area network links (e.g., Ethernet, etc.) that are generally fast, reliable and stable.

The connection between the data acquisition service 116 and the historian 100 can also take any of a variety of forms. In an embodiment of the present invention, the physical connection comprises an intermittent/slow connection 118 that is potentially: too slow to handle a burst of data, unavailable, or faulty. The data acquisition service 116 and/or the historian therefore include components and logic for handling the stream of data from components connected to the plant floor network 101. In view of the potential throughput/connectivity limitations of connection 118, to the extent secondary information is to be generated/provided to clients of the historian 100 (e.g., nodes 112), such information should be rendered after the data has traversed the connection 118. In an embodiment, the secondary information is rendered by advanced data retrieval operations incorporated into the historian 100.

Turning to **FIG. 2** an exemplary schematic diagram depicts functional components associated with the historian 100. The historian 100 generally implements a storage interface 200 comprising a set of functions/operations for receiving and tabling data from the data acquisition service 116 via connection 118. The received data is stored in one or more tables 202 maintained by the historian 100.

By way of example, the tables 202 include pieces of data received by the historian 100 via a data acquisition interface to a process control/production information network such as the data acquisition service 116 on network 101. In the illustrative embodiment each piece data is stored in the form of a value, quality, and time stamp. These three parts to each piece of data stored in the tables 202 of the historian 100 is described briefly herein below.

Timestamps

The historian 100, tables data received from a variety of "real-time" data sources, including the I/O Servers 104 (via the data acquisition service 116). The historian 100 is also capable of accepting "old" data from sources such as text files. By way of example, "real-time" data can be defined to exclude data with timestamps outside of ±30 seconds of a current time of

a clock maintained by a computer node hosting the historian 100. However, data characterizing information is also addressable by a quality descriptor associated with the received data. Proper implementation of timestamps requires synchronization of the clocks utilized by the historian 100 and data sources.

Quality

The Historian 100 supports two descriptors of data quality: "QualityDetail" and "Quality." The Qualitydescriptor is based primarily on the quality of the data presented by the data source, while the QualityDetail descriptor is a simple indicator of "good", "bad" or "doubtful", derived at retrieval-time. Alternatively, the historian 100 supports an OPCQuality descriptor that is intended to be used as a sole data quality indicator that is fully compliant with OPC quality standard(s). In the alternatively embodiment, the QualityDetail descriptor is utilized as an internal data quality indicator.

Value

A value part of a stored piece of data corresponds to a value of a received piece of data. In exceptional cases, the value obtained from a data source is translated into a NULL value at the highest retrieval layer to indicate a special event, such as a data source disconnection. This behavior is closely related to quality, and clients typically leverage knowledge of the rules governing the translation to indicate a lack of data, for example by showing a gap on a trend display.

The following is a brief description of the manner in which the historian 100 receives data from a real-time data source and stores the data as a timestamp, quality and value combination in one or more of its tables 202. The historian 100 receives a data point for a particular tag (named data value) via the storage interface 200. The historian compares the timestamp on the received data to: (1) a current time specified by a clock on the node that hosts the historian 100, and (2) a timestamp of a previous data point received for the tag. If the timestamp of the received data point is earlier than, or equal to the current time on the historian node then:

- If the timestamp on the received data point is later than the timestamp of the previous point received for the tag, the received point is tabled with the timestamp provided by the real-time data source.
- If the time stamp on the received data point is earlier than the timestamp of the previous point received for the tag (i.e. the point is out of sequence), the received point is tabled with the timestamp of the previously tabled data point "plus 5 milliseconds". A special QualityDetail value is stored with the received point to indicate that it is out of sequence (the original quality received from the data source is stored in the "quality" descriptor field for the stored data point).

On the other hand, if the timestamp of the point is later than the current time on the historian 100's node (i.e. the point is in the future), the point is tabled with a time stamp equal to the current time of the historian 100's node. Furthermore, a special value is assigned to the QualityDetail descriptor for the received/tabled point value to indicate that its specified time was in the future (the original quality received from the data source is stored in the "quality" descriptor field for the stored data point).

The historian 100 can be configured to provide the timestamp for received data identified by a particular tag. After proper designation, the historian 100 recognizes that the tag identified by a received data point belongs to a set of tags for which the historian 100 supplies a timestamp. Thereafter, the time stamp of the point is replaced by the current time of the historian 100's node. A special QualityDetail value is stored for the stored point to indicate that it was timestamped by the historian 100. The original quality received from the data source is stored in the "quality" descriptor field for the stored data point.

It is further noted that the historian 100 supports application of a rate deadband filter to reject new data points for a particular tag where a value associated with the received point has not changed sufficiently from a previous stored value for the tag.

Having described a data storage interface for the historian 100, attention is directed to retrieving the stored data from the tables 202 of the historian 100. Access, by clients of the historian 100, to the stored contents of the tables 202 is facilitated by a retrieval interface 206. The retrieval interface 206 exposes a set of functions/operations/methods (including a set of

advanced data retrieval operations 204), callable by clients on the network 110 (e.g., client applications on nodes 112), for querying the contents of the tables 202. The advanced data retrieval operations 204 generate secondary information, on demand, by post processing data stored in the tables 202. In response to receiving a query message identifying one of the advanced data retrieval options carried out by the operations 204, the retrieval interface 206 invokes the identified one of the set of advanced data retrieval operations 204 supported by the historian 100. An exemplary set of operations included in the advanced data retrieval operations 204 are enumerated in **FIG. 3** described herein below.

Turning to **FIG. 3**, in addition to known/standard delta and cyclic data retrieval modes an exemplary set of advanced data retrieval modes are supported by the historian 100. The advanced data retrieval modes, adding post processing to rows of data (corresponding to stored data value points) retrieved from the tables 202 of the historian 100, are facilitated by the set of advanced data retrieval operations 204 enumerated in **FIG. 3**. The set of operations are capable of operating on rows of data (grouped as cyclic buckets) associated with a single, specific tag, or alternatively a set of specified tags. Furthermore, in addition to specifying a time period over which the operation is to occur, a client potentially specifies particular options (e.g., interpolation method) for customizing completion of an operation on a tag-specific basis. Furthermore, in the illustrative embodiment each of the retrieval operations is implemented as a distinct object class from which instances are created and started either at start-up, or alternatively, upon the historian receiving a particular type of advanced data retrieval request.

In an exemplary embodiment, the advanced retrieval operations support options for tailoring data retrieval and processing tasks performed by the operation in response to a requesting client. Options specified in a request invoking a particular advanced retrieval operation include, for example, an interpolation method, a timestamp rule, and a data quality rule. Each of these three options is described herein below.

With regard to the interpolation method option, wherever an estimated value is to be returned to a requesting client for a particular specified time, the returned value is potentially determined in any of a variety of ways. In an illustrative example, the advanced retrieval operations support stair-step and linear interpolation. In the stair-step method, the operation

returns the last known point, or a NULL if no valid point can be found, along with a cycle time with which the returned stair-step value is associated. Turning to the example illustrated in FIG. 4a, where a retrieval operation receives a request for a "stair-step" value for a cycle having a boundary at time T_c, and the most recent point stored for the tag is P_1, the operation extends the last stored value assigned at P_1 and returns the value V_1 at time T_c.

Alternatively, linear interpolation is performed on two points to render an estimated value for a specified time. Turning to the example illustrated in FIG. 4b, where a retrieval operation receives a request for a linearly interpolated value for a cycle boundary at time T_c, and the most recently stored point for the tag is P_1, and the first point stored beyond T_c is P_2, the operation linearly interpolates between points P_1, and P_2. It is possible that one of the points will have a NULL value. If either of the points is NULL value, then P_1 is returned at time T_c. If both points are non-NULL, then the value V_c is returned as the value where the line through both points intersects with the cycle boundary, and the value V_c at time T_c is returned to the client. Expressed in a formula V_c is calculated as:

$$V_c = V_1 + ((V_2 - V_1) * ((T_c - T_1) / (T_2 - T_1)))$$

for $(T_2 - T_1) \neq 0$.

In an exemplary embodiment, whether the stair-step method or linear interpolation is used by an advanced retrieval operation specifying a given tag is determined, if not overridden, by a setting on the tag. If the setting is 'system default', then the system default is used for the tag. A client can override a specified system default for a particular query and designate stair-step or linear interpolation for all tags regardless of how each individual tag has been configured.

The data quality rule option on an advanced retrieval operation request controls whether points with certain characteristics are explicitly excluded from consideration by the algorithms of the advanced retrieval operations. By way of example, a client request optionally specifies a data quality rule, which is handed over to a specified advanced data retrieval operation. A client optionally specifies a quality rule (e.g., reject data that does not meet a particular quality standard in a predetermined scale). If no quality rule option is specified in a client request, then a default rule (e.g., no exclusions of points) is applied. In an exemplary embodiment, the client

specifies a quality rule requiring the responding operation to discard/filter retrieved points having doubtful quality – applying an OLE for process control (OPC) standard. The responding operation, on a per tag basis, tracks the percentage of points considered as having good quality by an algorithm out of all potential points subject to a request, and the tracked percentage is returned to the client.

The time stamp rule option applied to an advanced data retrieval request controls whether cyclic results are time stamped with a time marking the beginning of a cycle or the end of the cycle. In an illustrative example, a client optionally specifies a time stamp rule, and the time stamp is handed over to the operation. Otherwise, if no parameter is specified, then a default is applied to the advanced retrieval operation.

Turning to the set of operations listed in **FIG. 3**, an engineering units-based integral operation 300 calculates values to be provided by the historian 100 to a requesting client over a period of time (e.g., at cycle boundaries) by integrating a set of instantaneous values provided by previously stored data points stored for a tag over the period. Once invoked upon a particular tag or tags, the integral operation 300, by way of example, renders output cyclically (e.g., every second, minute, etc.) to the requesting client for a period specified in the query. In the exemplary embodiment, the integral operation can only be specified for an analog tag.

In an embodiment wherein the integral operation 300 renders output for each cycle (the length of which is specified by either a resolution/duration value or a number of cycles in a period – such as a day), the integral operation 300 initially calculates the "area under the curve" based upon a set of values stored for an analog tag over a cycle/period – using the specified resolution parameter to guide the process of retrieving data point values. After determining the area under the curve, the result is scaled using a specifically designated integral devisor for the particular tag. In an illustrative embodiment, the integral devisor is stored in a referenced entry in an engineering unit table. The designated integral divisor expresses a conversion factor from the actual rate to one of units over a designated standard period (e.g., second). Thus, during execution of the integral operation 300 instantaneous rate measurements are converted into a quantity over a specified time span. However, rather than basing the conversion on a fixed time (e.g., seconds) divisor, the integral operation 300 uses a time basis used by the stored data

points (and specified in the engineering unit table) and performs an appropriate conversion by referencing a conversion value stored in the engineering unit table in the historian 100. The engineering unit value conversion step renders the time-units of the original data points, from which the integral is determined, transparent to a requesting client of the historian 100. For example, the engineering units-based integral operation makes it possible to compare results from two separately rendered sets of volume flow measurements wherein the first set of measurements are expressed in "liters/sec" and a second set of measurements is expressed in "liters/min". The operation 300 applies a conversion factor specified by a tag associated with each of the two measurements and renders both results in the form of "liters." This contrasts with known integral operation implementations that require the client to know how (and remember) to convert from hard-coded reference units (if seconds, divide the integral results of the "liters/min" measure by 60) to implement the comparison.

While the integral operation 300 described above is performed cyclically. In alternative embodiments the integral operation 300 is called by the client with a specified start and stop time. The integral operation 300 returns a value to the requesting client corresponding to a measured quantity over the time period without a time-basis.

A derivative/slope operation 310 calculates a series of instantaneous rate of change estimates for a set of point values for an identified tag within a specified time span. By way of example, the derivative/slope operation 310 generates a rate of change estimate, for each stored tag value of interest, based upon observation of one or more point values adjacent to the tag value of interest. In a specific implementation of the derivative/slope operation 310, the rate of change estimate for a particular point value is determined by calculating a difference between the particular point value and the immediately preceding point value, and dividing by an elapsed time period between the two stored point values. However, the derivative/slope operation 310 can be estimated through alternative methods including using other point value combinations (e.g., current point and immediately subsequent point) and estimation techniques (e.g., curve fitting). In an exemplary embodiment, a "quality" option instructs the derivative/slope operation 310 to disregard points identified as having doubtful quality.

The derivative/slope operation 310 returns the calculated derivative/slope values in chronological order. Turning to **FIG. 5**, a graphical example of a data set acted upon by the derivative/slope operation 310 is provided. In this case a set of values associated with a single tag are processed over a time period starting at T_S and ending at T_E. In the illustrative example, nine points of tabled data are retrieved for the identified tag and period. The points are represented by dots marked P_1 through P_9. Of the nine points, eight represent normal analog values. However, P_5 represents a NULL value arising from an I/O server disconnect that created an absence of data between points P_5 and P_6. As previously explained, the derivative/slope operation 310 calculates, for each data point falling within the specified period, the slope of the line going through the data point of interest and the data point immediately prior to it. In an illustrative embodiment, the data points are calculated and returned for a one second delta change in time.

Furthermore, points outside the specified time period are utilized to calculate the slopes at the specified time period boundaries. Point P_2 in **FIG. 5** is located at the query start time, and because a qualifying prior point P_1 exists, a slope for point P_2 (at time T_S) corresponds to the line drawn through the two points. The derivative/slope operation 310 calculates a slope for point P_2 (at time T_S) according to the following formula: $S(T_S) = (P_2 - P_1) / (T_2 - T_1)$. Similar slope calculations are performed to render slopes for points P_3 , P_4, P_7, and P_8. A final slope is calculated for a calculated point P_{TE} at time T_E using the values and timestamps associated with points P_8 and P_9.

With regard to the handling of NULL values, no calculated slope value exists for point P_6 due to the NULL value associated with point P_5 – the prior point that would otherwise be used to calculate a slope value. Instead of returning a slope value at point P_6, as depicted by the flat line through the point, a slope value of zero is returned. A slope value of NULL is returned for time T_5 (corresponding to point P_5 having a NULL value).

A counter retrieval operation 320 uses a tag's rollover point to calculate and return a delta change over a period of time (e.g., between consecutive cycles as defined by a resolution/timespan for each cycle or a cycle count equal to the number of measurements taken every 24 hours). The counter retrieval operation 320 can be used to calculate a number of items

passing through a production line using a counter register that potentially rolls-over during a relatively predicable/foreseeable finite period. The counter retrieval operation 320 operates in a cyclic mode wherein a counter-compensated delta value is returned for each tag (in a client's query) for each cycle. The counter retrieval operation 320 provided a series of cyclically rendered delta values for a tag based upon a specified cycle duration/resolution which is indirectly specified by a number of cycles within a period (e.g, one day). The counter retrieval operation 320, in essence extends the range of a counter of limited size that is subject to relatively frequent rollover and would otherwise provide inaccurate delta value data over time. Alternatively, in the case of a counter that is reset before reaching a maximum value (prior to rollover), the highest retrieved data point value before resetting the counter is treated as the "maximum" count value.

Turning to **FIG. 6**, an exemplary set of twelve previously tabled data points for a specified tag are graphically depicted to illustrate the functionality of the counter retrieval operation 320. In the example depicted in **FIG. 6**, the counter retrieval operation 320 is executed based upon a query start time of T_{C0} and an end time of T_{C3}. In the illustrative example, the query resolution has been set such that the operation 320 returns data for three complete cycles starting at T_{C0}, T_{C1} and T_{C2} and returns an incomplete cycle starting at time T_{C3}. In the illustrative example the twelve points in the cycles of interest are represented by the points marked P_1 through P_{12}. Of these points eleven represent normal analog values, and one, P_9, represents a NULL due to an I/O server disconnect, which causes a gap in the data between stored data points P_9 and P_{10}. All points are found and considered by the counter retrieval logic, but only stored points P_1, P_2, P_6, P_7, P_9, P_{11} and P_{12}, and three interpolated points V_1, V_2, and V_3 are used to determine values to return to the requesting client. In the illustrative example, the counter retrieval operation 320 returns points P_{C0}, P_{C1}, P_{C2} and P_{C3} shown at the top to indicate that there is no simple relation between these calculated points and the actual stored data points from which they are calculated.

An 'initial value' to be returned at the query start time is not a simple stair-step or interpolated value. The initial value is calculated just like all other cycle values as the delta

change between the cycle time in question and a value calculated during a previous cycle – taking into account a rollover, if any, that occurred between the two points in time.

With regard to utilizing the counter operation 320 to render values for a requesting client, take for example the calculation of the value P_{C1}. The rollover point for the queried tag has been set to a value V_R, the interpolation type has been set to linear interpolation, and the timestamp rule has been set for the results to be timestamped at the end of the cycle. First interpolation is performed by the counter retrieval operation 320 to find values V_1 and V_2 at a first and second cycle boundary. Assuming that both sets of values pass a quality rule test, a calculation is performed on the interpolated values V_1 and V_2 to determine a counter value.

By way of example, if ***n*** rollovers have occurred during the course of a single cycle, then the counter-compensated delta (difference) difference between a tag value at time T_{C0} and time T_{C1} is defined as follows:

$$P_{C1} = \boldsymbol{n} * V_R + V_2 - V_1$$

Thus, if ***n*** is zero, we just calculate the difference between the values V_2 and V_1.

NULLs, by way of example, are handled in a number of ways. In the case of cycle C_2 we have no value at the cycle time. The counter retrieval operation 320 returns a NULL value represented by point P_9. In the case of cycle C_3 a NULL value is returned because there is no counter value at the previous cycle boundary to plug into the above formula. If a gap is fully contained inside a cycle, and if valid data points exist within the cycle on both sides of the gap, then a counter value is returned even though it may occasionally be erroneous -- as zero or one rollovers are assumed when in-fact the counter may have rolled over multiple times.

Yet another form of advanced retrieval is carried out by a time-in-state retrieval operation 330. The time-in-state retrieval operation 330 returns to a requesting client a variety of collective/aggregate information about the length of time that a specified tag attribute/portion (e.g., value, quality, quality detail, etc.) has been designated as occupying each of a set of possible states. The time-in-state retrieval operation 330 calculates statistics on the amount of time spent in the states represented by distinct tag values and returns the results to a requesting client.

By way of example, a client issues a time-in-state query to the historian 100 specifying a tag (or set of tags) and a portion of the tag information (e.g., value, quality, etc.) for which time-in-state information is desired. The query specifies a time period for which the requested time-in-state information is desired (e.g., one cycle, start/stop time, etc.). In an illustrative example, a resolution (duration of each cycle) is specified which determines a set of cycles into which the query time period is divided when rendering results. A set of requested information is returned for each cycle within a time period covered by a query. A time-in-state query also optionally specifies a timestamp rule – determining the relevant timestamp assigned to the query results for a cycle (e.g., the end time of the cycle). A query also specifies a quality rule/filter. An embodiment of the invention supports a set of time-in-state aggregation types (described in detail below). Thus, a client's time-in-state query to the historian 100 also specifies one or more aggregation methods to be applied to retrieved data to render responsive time-in-state information.

The advanced retrieval operations are invoked in a variety of ways. In an illustrative example, the operations are invoked as OLE extensions to a standard/base SQL database interface. In an alternative example wherein one or more of the advanced retrieval operations are implemented by object instances (e.g., COM/DCOM objects), the historian 100 invokes the time-in-state retrieval operation 330 through a call to an object instance for calculating and retrieving time-in-state information specified by the received client query.

The time-in-state retrieval operation 330 returns a set of time-in-state information for a specified tag's value (or separately specifiable values under a tag). Examples of supported tag value data types include: integer, discrete, and string. Responses are based upon the occupation of particular "value" states of a specified tag during a particular cycle as evidenced by the timestamps and values of data points retrieved for the tag during the cycle.

With continued attention to the content of a query to the historian 100 to initiate the time-in-state retrieval operation 330, the illustrative embodiment supports client requests specifying a variety of time-in-state data aggregation types. The aggregation types include: minimum, maximum, average, total, and percent. In the case of a minimum time-in-state request, the time-in-state operation 330 returns a time-wise shortest occurrence of each distinct

value for a specified tag within a time period (e.g., a cycle of interest). Similarly, a maximum time-in-state request returns a time-wise longest occurrence of each distinct value for a specified tag within a time period. An average time-in-state request returns an average time-wise duration of occurrences of each distinct value for a specified tag within a time period. In the case of a total time-in-state request, the time-in-state retrieval operation 330 returns the total time-wise occurrence of each distinct value for a specified tag within a time period. A percent time-in-state request results in the time-in-state retrieval operation 330 returning the percentage of the time period (e.g., cycle) spent in each distinct value for a specified tag. It is noted that while the above described operation operates on a single tag, embodiments of the historian 100 support queries specifying multiple tags and/or multiple returned time-in-state aggregate data types.

The historian 100 also supports an interpolated data retrieval operation 340. The purpose of the interpolated retrieval operation 340 is to use linear interpolation to calculate values to be returned at cycle boundaries. This operation will be described further by way of the illustrated example set forth in **FIG. 7**. Interpolated retrieval operation 340 operates according to defined cycle boundaries (i.e., operates in a cyclic mode). The interpolated retrieval operation 340 returns one value for each cycle. The time period/span of the interpolated response is determined by the specified number of cycles (in a 24 hour period). In addition to general query options (e.g., tags, time frame, resolution), the interpolated retrieval operation 340 supports an option for overriding the interpolation type and selecting a time stamp rule. The interpolation retrieval operation 340 operates solely on analog tags. For the extent of the query the interpolation type of all other types of tags are forced to stair-step, and results are returned to the client application accordingly.

FIG. 7 shows an example of how points for an analog tag are selected in an interpolated query. In the example an interpolated query specifies a start time of T_{C0} and an end time of T_{C2}. The resolution/cycle duration has been set in such a way that the historian 100 returns data at three cycle boundaries (T_{C0}, T_{C1} and T_{C2}). For the queried tag we find a total of twelve points throughout the cycles represented by the dots marked P_1 through P_{12}. Of these points eleven represent normal analog values, and one, P_7, represents a NULL due to an I/O server

disconnect, which causes a gap in the data between P_7 and P_8. Points P_2 and P_{C2} are returned to the requesting client application. The points P_7 ,P_{11} and P_{12} are used to calculate the returned points at the cycle boundaries. It is noted that since P_2 is at the query start time, no interpolation function is used for that particular cycle boundary. At the next cycle boundary point P_{C1}, is returned – which is the NULL represented by point P_7 shifted forward to time T_{C1}. Finally, at the last cycle boundary point P_{C2} is returned which has been interpolated using points P_{11} and P_{12}.

The historian 100 supports a Best-fit data retrieval operation 350. The Best-fit retrieval operation 350 uses cyclic buckets, but it is not a true cyclic operation. Apart from an initial value, the Best-fit operation 350 only returns actual delta points. The Best-fit operation 350 invokes low level retrieval to retrieve and provide previously tabled data. The Best-fit retrieval operation 350 applies the best-fit algorithm to the retrieved values in view of the specified resolution. For best-fit and other queries, the user can specify the resolution indirectly by specifying a cycle count. The returned values typically number more than one per cycle. A query option available for the Best-fit data retrieval operation 350 allows overriding the interpolation type for the calculation of initial values. The Best-fit retrieval operation 350 applies a best-fit algorithm to all points found in any given cycle. Thus up to five delta points are returned within each cycle for each tag: the first value, the last value, the min value, the max value and the first occurrence of any existing exceptions. If one point fulfills multiple designations, then the data point is returned only once. In a cycle where a tag has no points, nothing will be returned. The best-fit operation 350 can only be applied to analog tags. For all other tags specified in a client's query, normal delta results are returned by the historian 100 to the client. All points returned to the client will be in chronological order, and if multiple data points are to be returned for a particular time stamp, then those points will be returned in the order in which the client has specified the respective tags in the query.

FIG. 8 shows an illustrative example of selecting points based upon a Best-fit query. In the example the best-fit operation 350, in response to a particular query, commences with a start time of T_{C0} and an end time of T_{C2}. The resolution of the request submitted to the historian is set such that data is returned for two complete cycles starting at T_{C0} and T_{C1} and an

incomplete cycle starting at T_{C2}. For the queried tag we again find a total of twelve points throughout the cycles represented by the dots marked P_1 through P_{12}. Of these points eleven represent normal analog values, and one, P_7, represents a NULL due to an I/O server disconnect, which causes a gap in the data between P_7 and P_8. Two points, P_1 and P_{12}, are not considered at all. P_1 is not considered because P_2 is located exactly at the start time and there is no need to interpolate. P_{12} is not considered because it is outside of the queried time frame. All other points are considered. However, for the reasons provided below, only data points 2, 4, 6, 7, 8, 9 and 11 are returned.

With continued reference to **FIG. 8**, four points are returned from the first cycle. P_2 is returned as the initial value of the query as well as the first value in the cycle. P_4 is returned as the minimum value in the cycle. P_6 returned as the maximum value and the last value in the cycle. P_7 is returned as the first occurring - and in this case the only - exception in the cycle. In the second cycle three points are returned. P_8 is returned as the first value in the cycle. P_9 is returned as the maximum value in the cycle. P_{11} is returned as both the minimum value and the last value in the cycle. As no exception occurs in the cycle, no point will be returned for this aspect of the best fit operation 350 for the second cycle. No points are returned for the incomplete third cycle starting at the query end time because the tag (associated with the displayed points) does not have a point exactly at that time.

The historian 100 supports a time-weighted average data retrieval operation 360. The time weighted average (TWA) retrieval operation 360 calculates values, returned at cycle boundaries, using a time weighted average algorithm. The TWA retrieval operation 360 is a true cyclic operation. It returns one value (the average) per cycle for each tag specified in the client's request. In addition to standard query options, the request invoking the TWA retrieval operation 360 allows a client to override the interpolation type and specify timestamp and quality rules. The TWA retrieval operation 360 is applied to analog tags. If a query contains discrete tags, then normal cyclic results are returned for those tags.

FIG. 9 illustratively depicts a sequence of data points for a specified tag to aid understanding the calculated results rendered by the TWA retrieval operation 360. In the example a TWA query has a start time of T_{C0} and an end time of T_{C2}. The resolution has been

set such that the TWA retrieval operation 360 returns data for two complete cycles starting at T_{C0} and T_{C1} and an incomplete cycle starting at T_{C2}.

A total of nine points (marked P_1 through P_9) are provided for the queried tag throughout the shown cycles. Of these points eight represent normal analog values, and one, P_5, represents a NULL due to an I/O server disconnect, which causes a gap in the data between data points P_5 and P_6.

Assuming the query calls for time stamping at the end of the cycle, the 'initial value' to be returned at the query start time, in this example T_{C0}, is not a simple stair-step or interpolated value as is usual. Instead the initial value is the result of applying the TWA algorithm to a cycle immediately preceding the query range. In the same scenario the value returned at time T_{C1} is the result of applying the TWA algorithm to points in the cycle starting at the query start time, and the value to be returned at the query end time T_{C2} is the result of applying the TWA algorithm to the points in the cycle starting at T_{C1}.

Taking the last cycle depicted in **FIG. 9** as an example, in order to calculate a time weighted average, the area under the curve (depicted by the lines connecting contained non-null points P_6, P_7, P_8 and P_{C2} which represents the interpolated value at time T_{C2} using points P_8 and P_9) is initially determined. The data gap at the beginning of the cycle, caused by the I/O server disconnect, does not contribute to the calculated area. Furthermore, if a quality rule of "only good" points is specified, then points with doubtful quality will not contribute to the area calculation. Focusing upon points P_6 and P_7, the TWA operation 360 calculates the area contribution between these two points by multiplying the arithmetic average of value P_6 and value P_7 by the time difference between the two points – that is $((P_6 + P_7) / 2) \times (T_7 - T_6)$. After calculating the area for the whole cycle, the TWA calculation is finished by dividing the area under the curve by the cycle time (less any periods within the cycle, which did not contribute anything to the area calculation). This calculated average is returned at the cycle end time.

Referring to the first cycle depicted in **FIG. 9**, it is noted that in the time frame between points P_4 and P_5, the line through point P_4 representing the area under the curve is parallel to the X-axis (i.e., the value is constant). This is due to the fact that P_5 represents a NULL point value, which cannot be used to calculate an arithmetic average. Instead the historian 100 uses

the value P_4 for the whole time period between points P_4 and P_5. The area calculation is signed. Therefore, if the arithmetic average between two points is negative, then the contribution to the area is negative.

The historian 100 supports a min-with-time data retrieval operation 370. The min-with-time data retrieval operation 370 operates upon cyclic buckets. However, the min-with-time operation 370 is not a true cyclic mode. With the exception of an initial value, the points retrieved are delta points (i.e., where a tag value has changed). The values/rows returned by low level retrieval components of the historian 100 potentially number more than one per cycle. A new column is supported that identifies the number of delta points for a tag that are returned for the cycle. The min-with-time data retrieval operation 370 supports a requestor overriding the default/specified interpolation type for calculating initial values.

The min-with-time retrieval operation 370 applies a simple minimum algorithm to all points retrieved by low level retrieval in any given cycle, and returns a data point having a minimum value along with its actual timestamp. In a cycle where a tag has no points nothing will be returned. The minimum-with-time algorithm can only be applied to analog tags. For all other tags normal delta results are returned to the requesting client of the historian 100. All points returned to the client are in chronological order. If multiple points (for different tags) are returned for a particular time stamp, then those points will be returned in the order in which the client has specified the respective tags in the query.

FIG. 10 shows an example of how the minimum algorithm selects points for an analog tag. The min-with-time operation 370 is executed with a start time of T_{C0} and an end time of T_{C2}. The time period has been set such that data is returned for two complete cycles starting at T_{C0} and T_{C1} and an incomplete cycle starting at T_{C2}. A total of twelve points are identified for the queried tag throughout the cycles represented by the dots marked P_1 through P_{12}. Of these points eleven represent normal analog values, and one, P_7, represents a NULL due to an I/O server disconnect, which causes a gap in the data between P_7 and P_8. Two points, P_1 and P_{12}, are not considered at all. P_1 is not considered because we do not need to interpolate at the query start time, since P_2 is located exactly at the start time. P_{12} is not considered because it is outside of the queried time frame. All other points are considered, but only points P_2 P_4 P_7 and P_{11} are

returned. P_2 is returned as the initial value of the query. P_4 is returned as the minimum value in the first cycle. P_7 is returned as the first and only exception occurring in the first cycle. P_{11} is returned as the minimum value in the second cycle. No points are returned for the incomplete third cycle starting at the query end time, because the tag does not have a point exactly at that time.

The last of the illustrative extensible set of advanced retrieval operations supported by the historian 100 is a maximum-with-time data retrieval operation 380. The maximum-with-time retrieval operation 380 uses cyclic buckets, but it is not a true cyclic operation. Apart from the initial value all subsequently returned data points are delta points. The rows returned by low level retrieval may number more than one per cycle. A call to the maximum-with-time data retrieval operation 380 optionally overrides a specified interpolation type for the calculation of initial values.

The maximum-with-time retrieval operation 380 applies a very simple maximum algorithm to time stamped data points for a tag found in any given cycle and returns a point having a maximum value with its actual timestamp. In a cycle where a tag has no data points, nothing will be returned. The MAX-with-time algorithm is applied to analog tags. For all other tags normal delta results are returned to the client. All points returned by the historian 100 to a requesting client are in chronological order. If multiple points (from different tags) are returned for a particular time stamp, then the points are returned in the order in which the client has specified the respective tags in the query that invoked the maximum-with-time operation 380.

FIG. 11 shows an example of how the maximum-with-time algorithm selects data points for an analog tag. In the example the maximum-with-time operation 380 executes with a start time of T_{C0} and an end time of T_{C2}. The time period has been set such that data is returned for two complete cycles starting at T_{C0} and T_{C1} and an incomplete cycle starting at T_{C2}. A total of 12 data points are identified for the specified tag during the designated time frame. The points are labeled P_1 through P_{12}. Of these points eleven represent normal analog values, and one, P_7, represents a NULL due to an I/O server disconnect, which causes a gap in the data between data points P_7 and P_8. Two points, P_1 and P_{12}, are not considered. P_1 is not considered because there is no need to interpolate at the query start time because P_2 is located exactly at the

start time, and P_{12} is not considered because it is outside of the queried time frame. All other points are considered, but only points P_2 P_6 P_7 and P_9 are returned. P_2 is returned as the initial value of the query. P_6 is returned as the maximum value within the first cycle. P_7 is returned as the first and only exception occurring in the first cycle, and P_9 is returned as the maximum value in the second cycle. No points are returned for the incomplete third cycle starting at the query end time because the tag does not have a point exactly at that time.

It is noted that the historian 100 embodies an extensible platform facilitating defining/incorporating any further developed advanced retrieval operations. The ability to handle a virtually limitless number of the advance retrieval operations is largely attributable to the production of the values in response to client queries as opposed to when the streaming input data is received and stored by the historian 100.

Having described an exemplary functional/structural arrangement for a historian incorporating advanced data retrieval operations, attention is directed to a flow diagram summarizing the general operation of the historian 100, schematically depicted in **FIG. 2** and including the advanced data retrieval operations 204, to process requests from clients invoking specified ones of the advanced data retrieval operations 204 to render secondary/advanced data. The invoked operations render the secondary/advanced data by applying defined data processing algorithms on data retrieved from the tables 202.

Turning to **FIG. 12**, during step 400, the historian 100 receives a request from the client application on node 112a via the data retrieval interface 206. The request generally identifies one or more data items (e.g., tags) and one of the advanced data retrieval operations. Thereafter, at step 410 an appropriate interface call (e.g., SQL Query) is formulated from the received request. During step 410 the options specified in the request and a set of option defaults are used to tailor a set of parameters passed in the interface call that will be used to invoke an advanced data retrieval operation corresponding to the received request. At step 420 the interface call is issued thereby invoking the advanced data retrieval operation corresponding to the received request. Next, at step 430, the invoked advanced data retrieval operation retrieves previously tabled data maintained in the data tables 202 and processes the retrieved

tabled data to render responsive advanced data corresponding to the advance data retrieval request received during step 400. At step 440 the historian 100 generates a response to the received advanced data retrieval request and passes the response to the client application on node 112a.

In view of the many possible embodiments to which the principles of this invention may be applied, it should be recognized that the embodiments described herein with respect to the drawing figures, as well as the described alternatives, are meant to be illustrative only and should not be taken as limiting the scope of the invention. The functional components disclosed herein can be incorporated into a variety of programmed computer systems in the form of software, firmware, and/or hardware. Furthermore, the illustrative steps may be modified, supplemented and/or reordered without deviating from the invention. Therefore, the invention as described herein contemplates all such embodiments as may come within the scope of the following claims and equivalents thereof.

What is claimed is:

1. A control system database server incorporating a database service supporting advanced data retrieval operations for creating, on-request by database service clients, sets of processed data from previously tabled control system data stored from streams of real-time data points, the control system database server comprising:

a set of advanced data retrieval operations, interposed between a general database interface and low level data retrieval components of the server, that, when invoked by an advanced data retrieval request from a client, retrieves data from previously tabled control system data, and processes the retrieved tabled data to render responsive data corresponding to the advance data retrieval request;

a data retrieval request handler, incorporated within the general database interface, for receiving the advanced data retrieval request from the client identifying one of the advanced data retrieval operations, and in response invoking the identified advanced data retrieval operation.

2. The control system database server of claim 1 wherein the set of advanced data retrieval operations includes an engineering units-based integral data retrieval operation comprising an integrated time unit conversion operation that converts a rate-based measurement to a quantity rendered as a result.

3. The control system database server of claim 1 wherein the set of advanced data retrieval operations includes a derivative/slope data retrieval operation that renders a set of rate of change values corresponding to a series of data points corresponding to the request.

4. The control system database server of claim 1 wherein the set of advanced data retrieval operations includes a counter data retrieval operation that analyzes a set of data points within a specified time span and renders a result accounting for counter rollovers occurring during the time span.

5. The control system database server of claim 1 wherein the set of advanced data retrieval operations includes a time-in-state data retrieval operation that returns a set of calculated time-in-state statistics for a data stream during a time span.

6. The control system database server of claim 5 wherein the time-in-state retrieval operation calculates and returns time-wise shortest occurrences of each distinct value for a specified data tag.

7. The control system database server of claim 5 wherein the time-in-state retrieval operation calculates and returns time-wise longest occurrences of each distinct value for a specified data tag.

8. The control system database server of claim 5 wherein the time-in-state retrieval operation calculates and returns time-wise occurrence averages of each distinct value for a specified data tag.

9. The control system database server of claim 5 wherein the time-in-state retrieval operation calculates and returns time-wise totals for occurrences of each distinct value for a specified data tag.

10. The control system database server of claim 5 wherein the time-in-state retrieval operation calculates and returns percentages of a cycle time spent in each distinct value for a specified data tag.

11. The control system database server of claim 1 wherein the set of advanced data retrieval operations is extensible.

12. A method, performed by a control system database server incorporating a database service supporting a set of advanced data retrieval operations, for creating, on-request by database service clients via a data retrieval request handler, sets of processed data from previously tabled control system data corresponding to previously received real-time data streams, the method comprising:

receiving, via the data retrieval request handler, a data retrieval request specifying one of the advanced data retrieval operations;

invoking, on the database server, an advanced data retrieval operation, interposed between the data retrieval request interface and low level data retrieval components of the server, corresponding to the advanced data retrieval operation specified in the received data retrieval request, and in response performing, in accordance with the invoked advanced data retrieval operation the further steps of:

retrieving data from the previously tabled control system data, and

processing the retrieved data to render responsive advanced data corresponding to the advance data retrieval request.

13. The method of claim 12 wherein the set of advanced data retrieval operations includes an engineering units-based integral data retrieval operation comprising an integrated time units conversion operation that converts a rate-based measurement to a quantity rendered as a result.

14. The method of claim 12 wherein the set of advanced data retrieval operations includes a derivative/slope data retrieval operation that renders a set of rate of change values corresponding to a series of data points corresponding to the request.

15. The method of claim 12 wherein the set of advanced data retrieval operations includes a counter data retrieval operation that analyzes a set of data points within a specified time span and renders a result accounting for counter rollovers occurring during the time span.

16. The method of claim 12 wherein the set of advanced data retrieval operations includes a time-in-state data retrieval operation that returns a set of calculated time-in-state statistics for a data stream during a time span.

17. A computer-readable medium including computer-executable instructions, performed by a control system database server incorporating a database service supporting a set of advanced data retrieval operations, for facilitating creating, on-request by database service clients via a data retrieval request handler, sets of processed data from previously tabled control system data corresponding to previously received real-time data streams, the computer-executable instructions facilitating performing the steps of:

receiving, via the data retrieval request handler, a data retrieval request specifying one of the advanced data retrieval operations;

invoking, on the database server, an advanced data retrieval operation, interposed between the data retrieval request interface and low level data retrieval components of the server, corresponding to the advanced data retrieval operation specified in the received data retrieval request, and in response performing, in accordance with the invoked advanced data retrieval operation the further steps of:

retrieving data from the previously tabled control system data, and

processing the retrieved data to render responsive advanced data corresponding to the advance data retrieval request.

18. The computer-readable medium of claim 17 wherein the set of advanced data retrieval operations includes an engineering units-based integral data retrieval operation comprising an integrated time units conversion operation that converts a rate-based measurement to a quantity rendered as a result.

19. The computer-readable medium of claim 17 wherein the set of advanced data retrieval operations includes a derivative/slope data retrieval operation that renders a set of rate of change values corresponding to a series of data points corresponding to the request.

20. The computer-readable medium of claim 17 wherein the set of advanced data retrieval operations includes a counter data retrieval operation that analyzes a set of data points within a specified time span and renders a result accounting for counter rollovers occurring during the time span.

21. The computer-readable medium of claim 17 wherein the set of advanced data retrieval operations includes a time-in-state data retrieval operation that returns a set of calculated time-in-state statistics for a data stream during a time span.

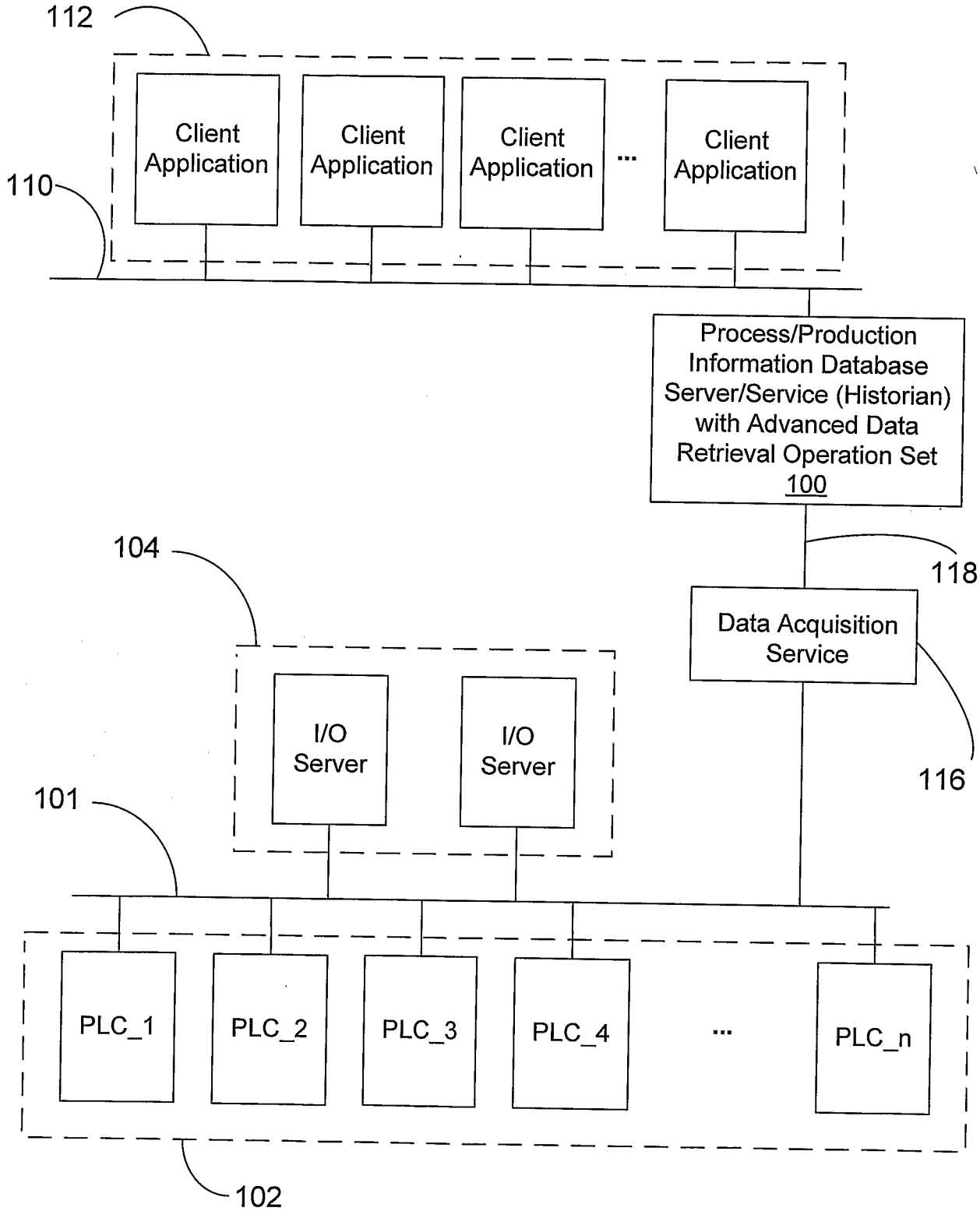
112
Client Application
Client Application
Client Application
...
Client Application
110
Process/Production Information Database Server/Service (Historian) with Advanced Data Retrieval Operation Set
100
104
118
Data Acquisition Service
I/O Server
I/O Server
116
101
PLC_1
PLC_2
PLC_3
PLC_4
...
PLC_n
102

FIG. 1

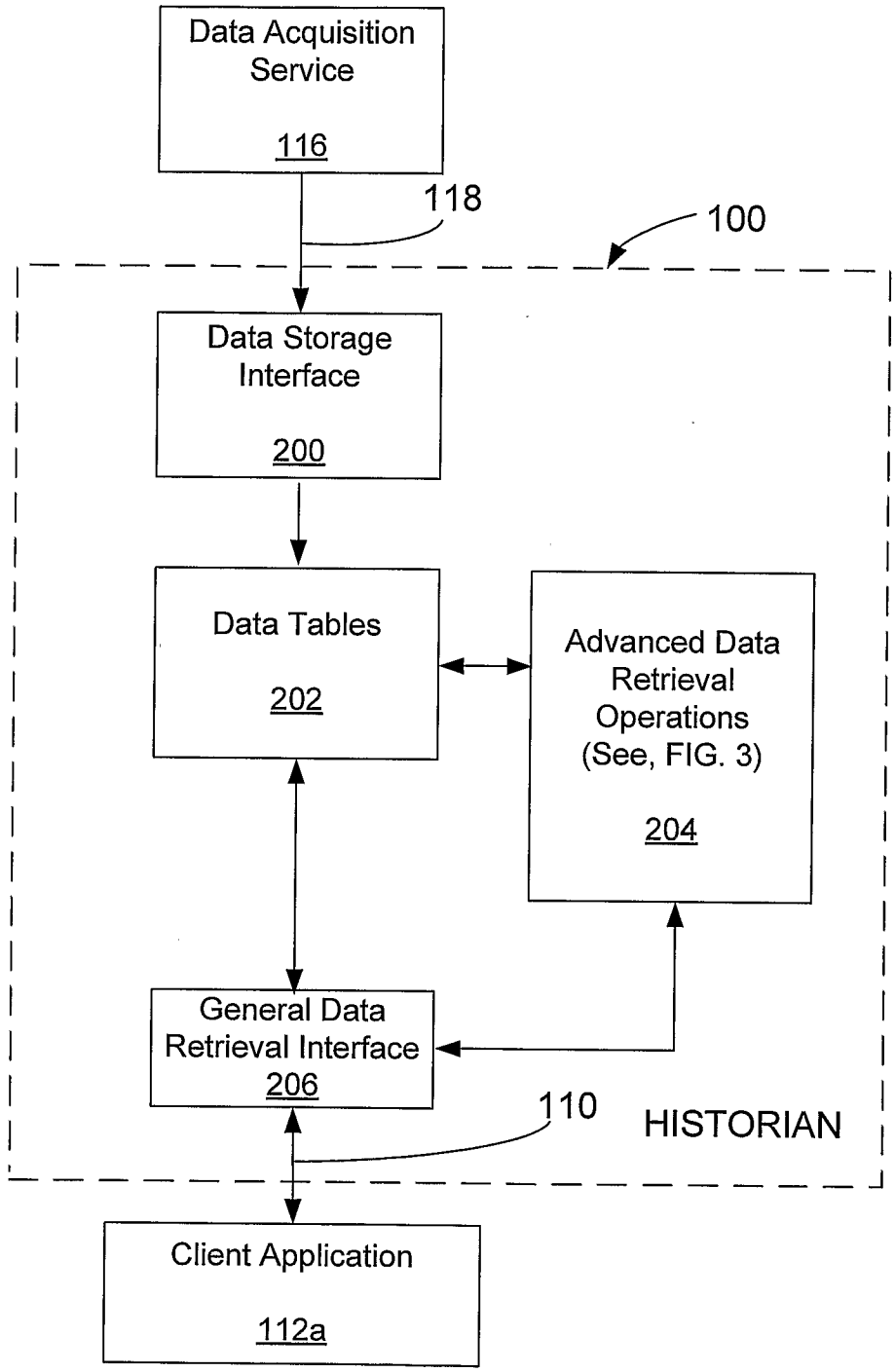
Data Acquisition Service
116
118
100
Data Storage Interface
200
Data Tables
202
Advanced Data Retrieval Operations (See, FIG. 3)
204
General Data Retrieval Interface
206
110
HISTORIAN
Client Application
112a

FIG. 2

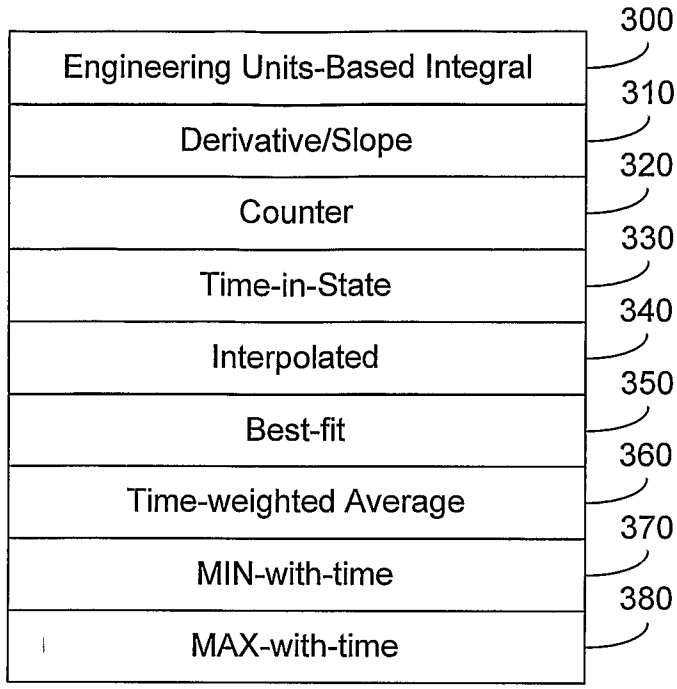

FIG. 3

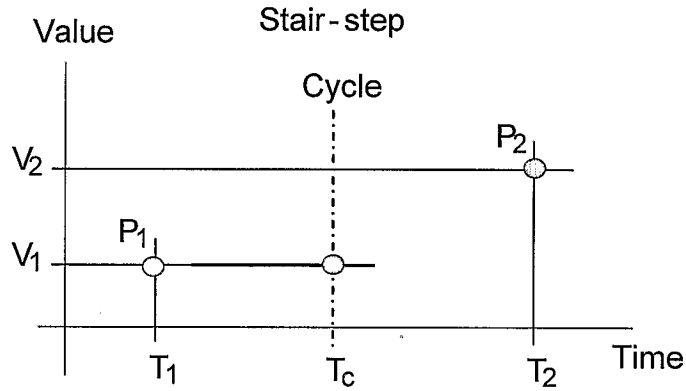

FIG. 4a

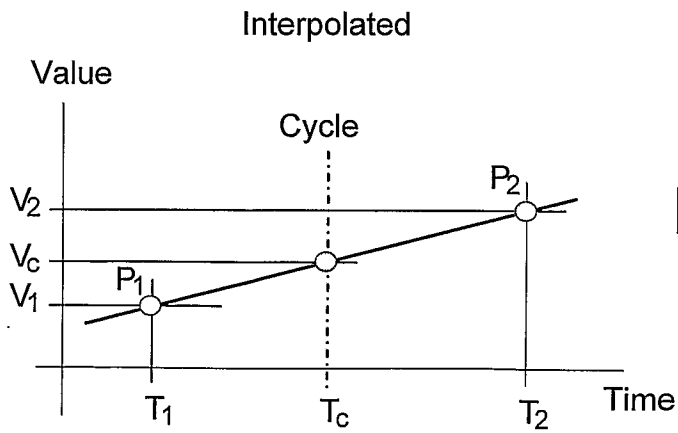

FIG. 4b

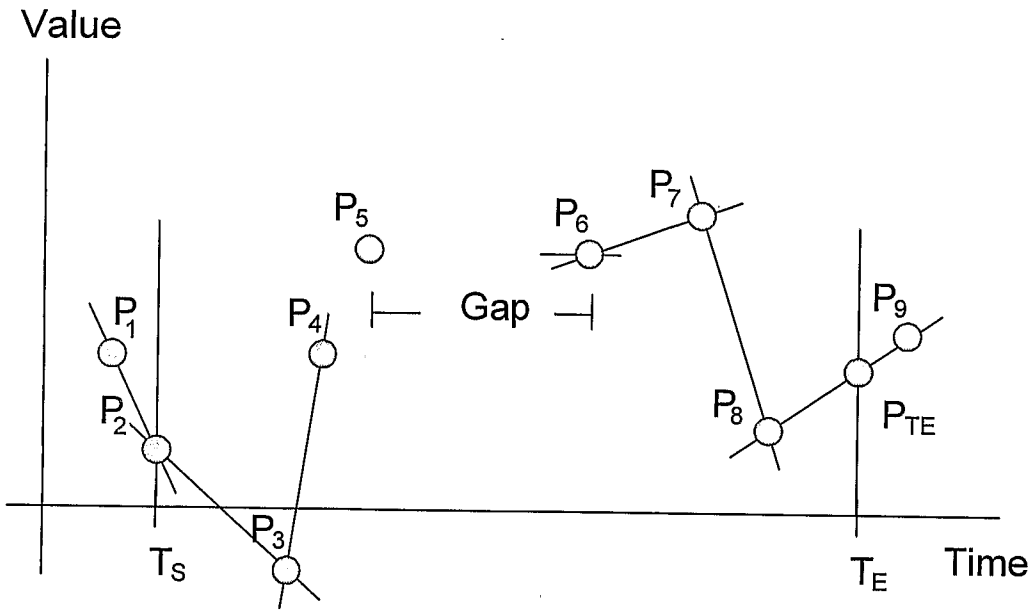
Value
P5
P6
P7
P1
P4
Gap
P9
P2
P8
PTE
P3
TS
TE
Time

FIG. 5

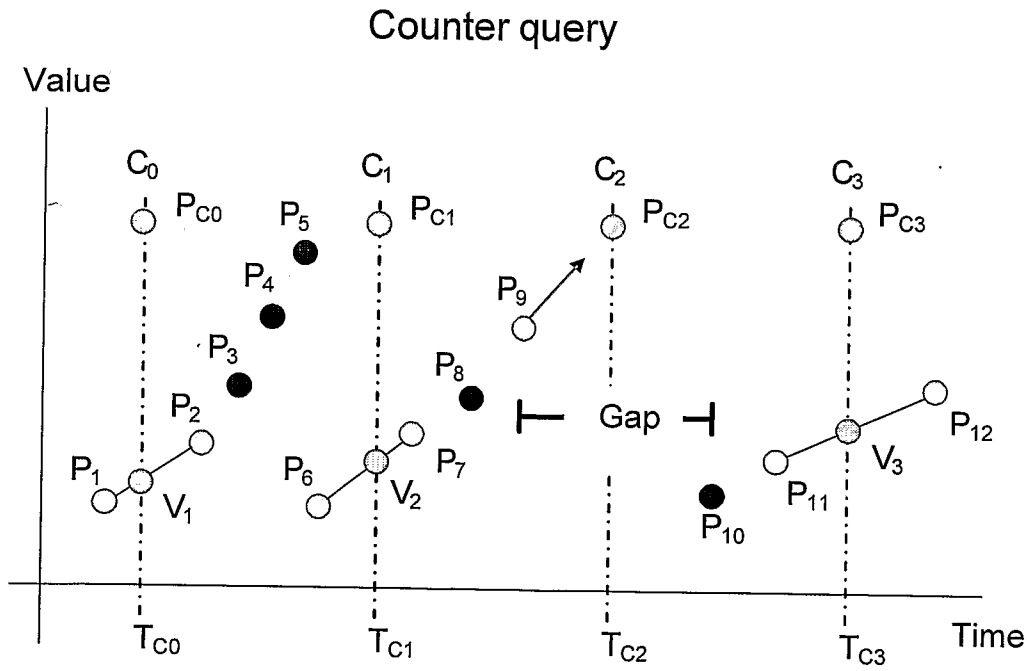
Counter query
Value
C0
PC0
P5
C1
PC1
C2
PC2
C3
PC3
P4
P9
P3
P8
P2
Gap
P12
P1
P6
P7
V3
V1
V2
P11
P10
TC0
TC1
TC2
TC3
Time

FIG. 6

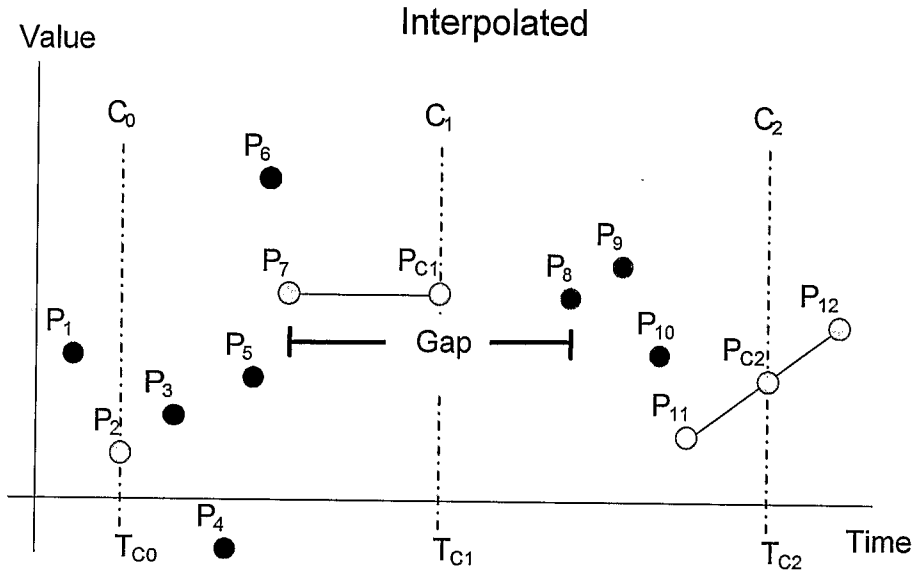
Interpolated
Value
C0
C1
C2
P6
P7
PC1
P9
P8
P12
P1
P5
Gap
P10
PC2
P3
P2
P11
P4
TC0
TC1
TC2
Time

FIG. 7

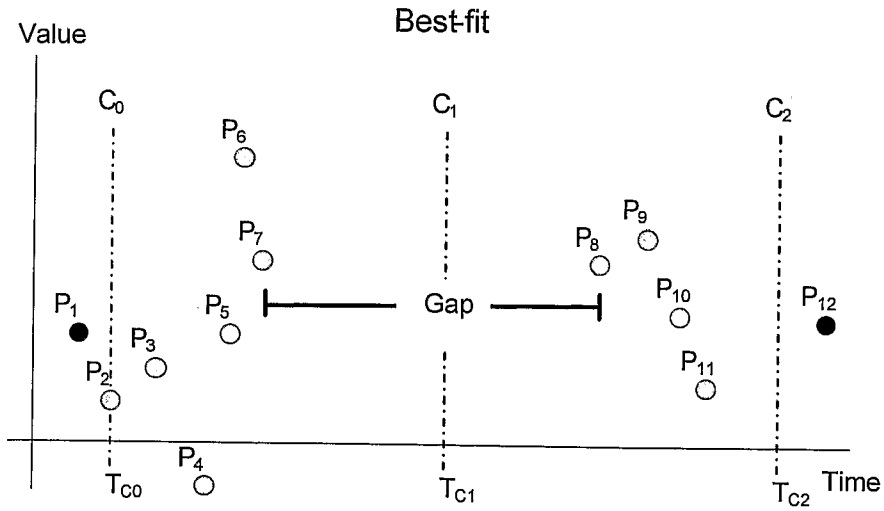
Best-fit
Value
C0
C1
C2
P6
P7
P9
P8
P1
P5
Gap
P10
P12
P3
P2
P11
P4
TC0
TC1
TC2
Time

FIG. 8

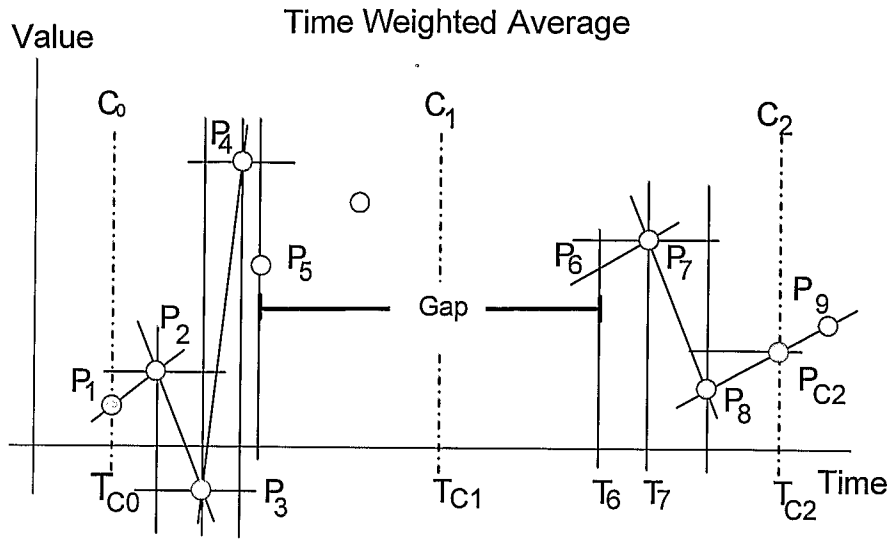

FIG. 9

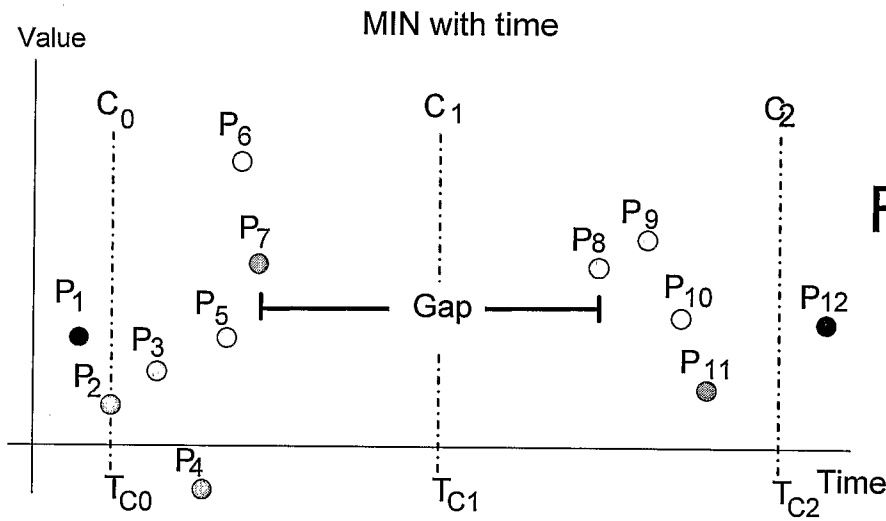

FIG. 10

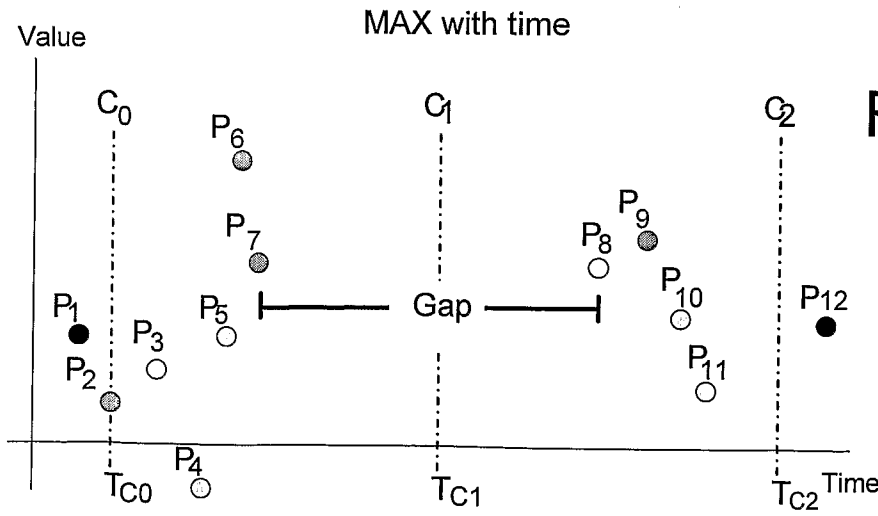

FIG. 11

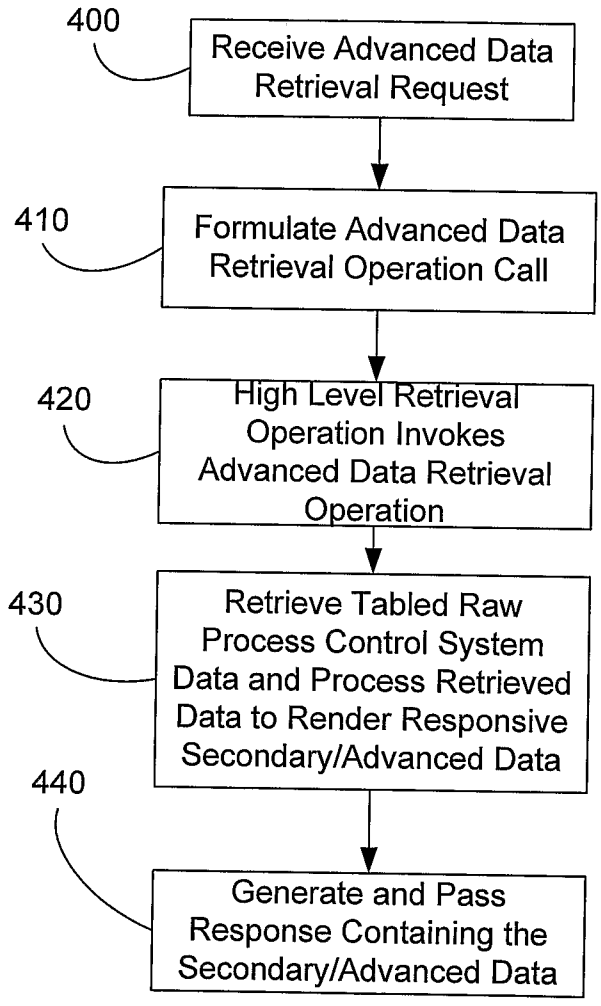
400
Receive Advanced Data Retrieval Request
410
Formulate Advanced Data Retrieval Operation Call
420
High Level Retrieval Operation Invokes Advanced Data Retrieval Operation
430
Retrieve Tabled Raw Process Control System Data and Process Retrieved Data to Render Responsive Secondary/Advanced Data
440
Generate and Pass Response Containing the Secondary/Advanced Data

FIG. 12