US 20090094271A1

(19) United States

(12) Patent Application Publication

Brewer et al.

(10) Pub. No.: US 2009/0094271 A1

(43) Pub. Date: Apr. 9, 2009

(54) VARIABLE DRIVEN METHOD AND SYSTEM FOR THE MANAGEMENT AND DISPLAY OF INFORMATION

(75) Inventors: **Michael A. Brewer**, Centennial, CO (US); **Frederick C. Wagner**, Centennial, CO (US)

Correspondence Address:
MARSH, FISCHMANN & BREYFOGLE LLP
8055 East Tufts Avenue, Suite 450
Denver, CO 80237 (US)

(73) Assignee: **ALLURDATA LLC**, Greenwood Village, CO (US)

(21) Appl. No.: **12/147,347**

(22) Filed: **Jun. 26, 2008**

Related U.S. Application Data

(60) Provisional application No. 60/946,293, filed on Jun. 26, 2007.

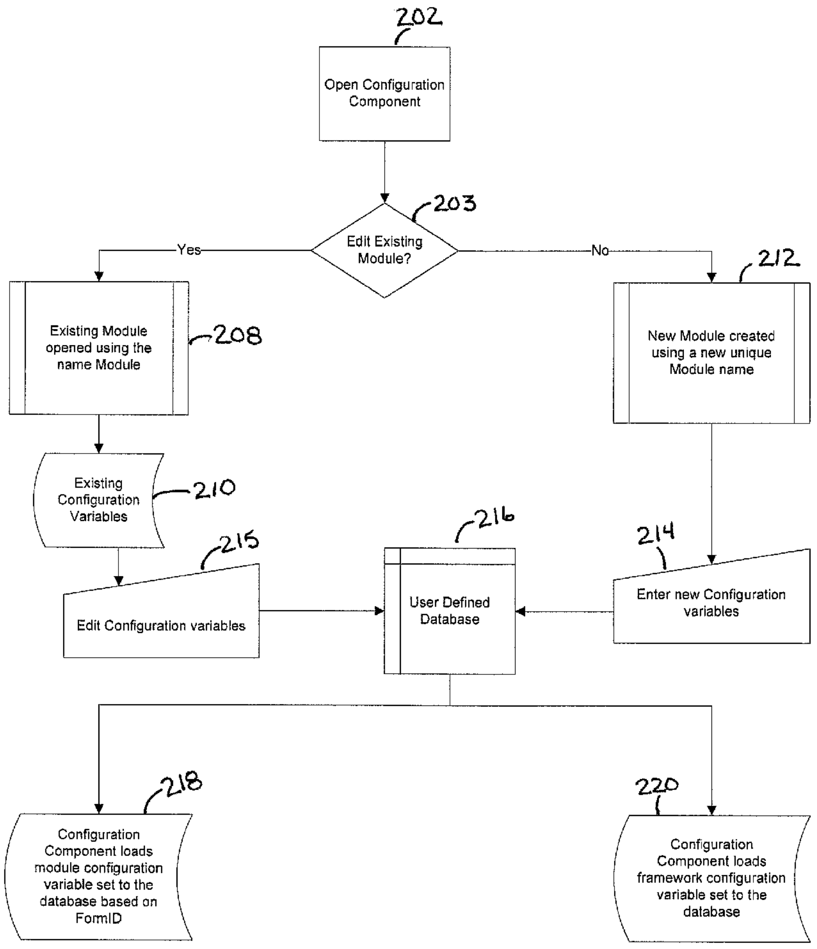
Publication Classification

(51) **Int. Cl.**
G06F 17/30 (2006.01)

(52) **U.S. Cl.** **707/102**; 707/E17.044

(57) **ABSTRACT**

A method and system for the management and display of information in a graphical user interface (GUI). Key data fields are configured, including a primary key data field and secondary key data fields, which are associated with the primary key data field. The data fields are configured using configuration variables and the data fields are populated with data field values. Information is displayed to a user in the GUI that includes secondary key data field values in a skewable display tree. Selection of a database record, identified by the value of the primary key data field, through the display tree generates a display of additional data field values that are associated with the selected database record, including text boxes for editing data field values by the user. Searching and reporting functions on the data field values under the primary key data field can also be provided.


202
Open Configuration Component
203
Edit Existing Module?
Yes
No
Existing Module opened using the name Module
208
212
New Module created using a new unique Module name
Existing Configuration Variables
210
215
Edit Configuration variables
216
User Defined Database
214
Enter new Configuration variables
218
Configuration Component loads module configuration variable set to the database based on FormID
220
Configuration Component loads framework configuration variable set to the database

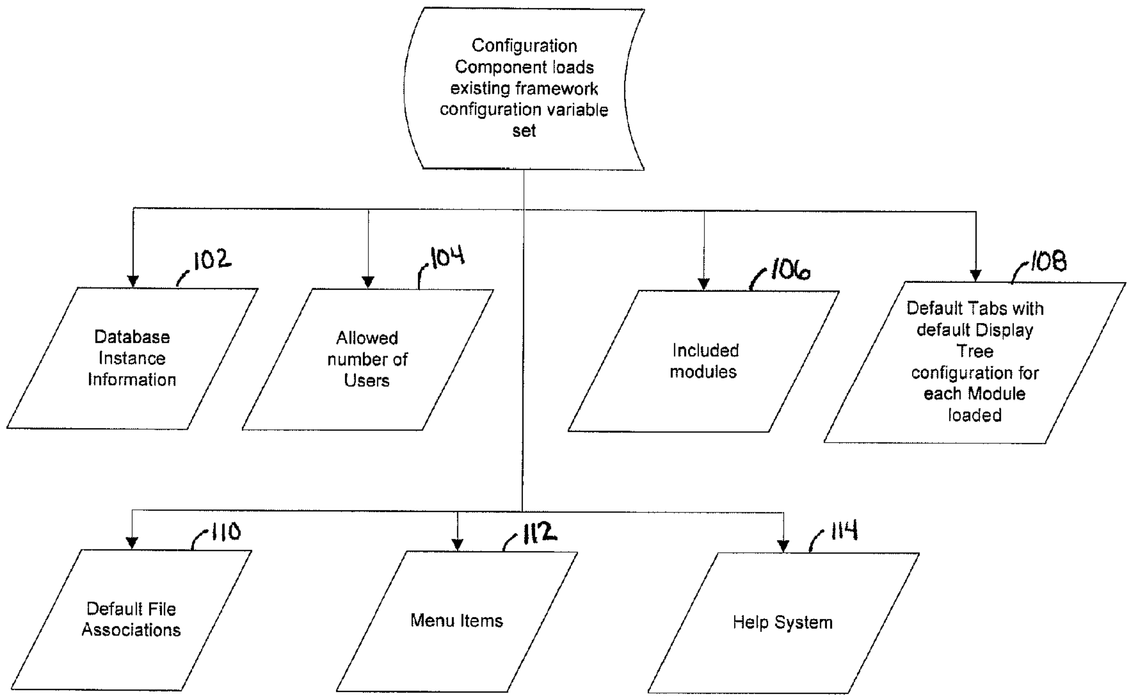

Configuration Component loads existing framework configuration variable set
102
Database Instance Information
104
Allowed number of Users
106
Included modules
108
Default Tabs with default Display Tree configuration for each Module loaded
110
Default File Associations
112
Menu Items
114
Help System

Fig. 1

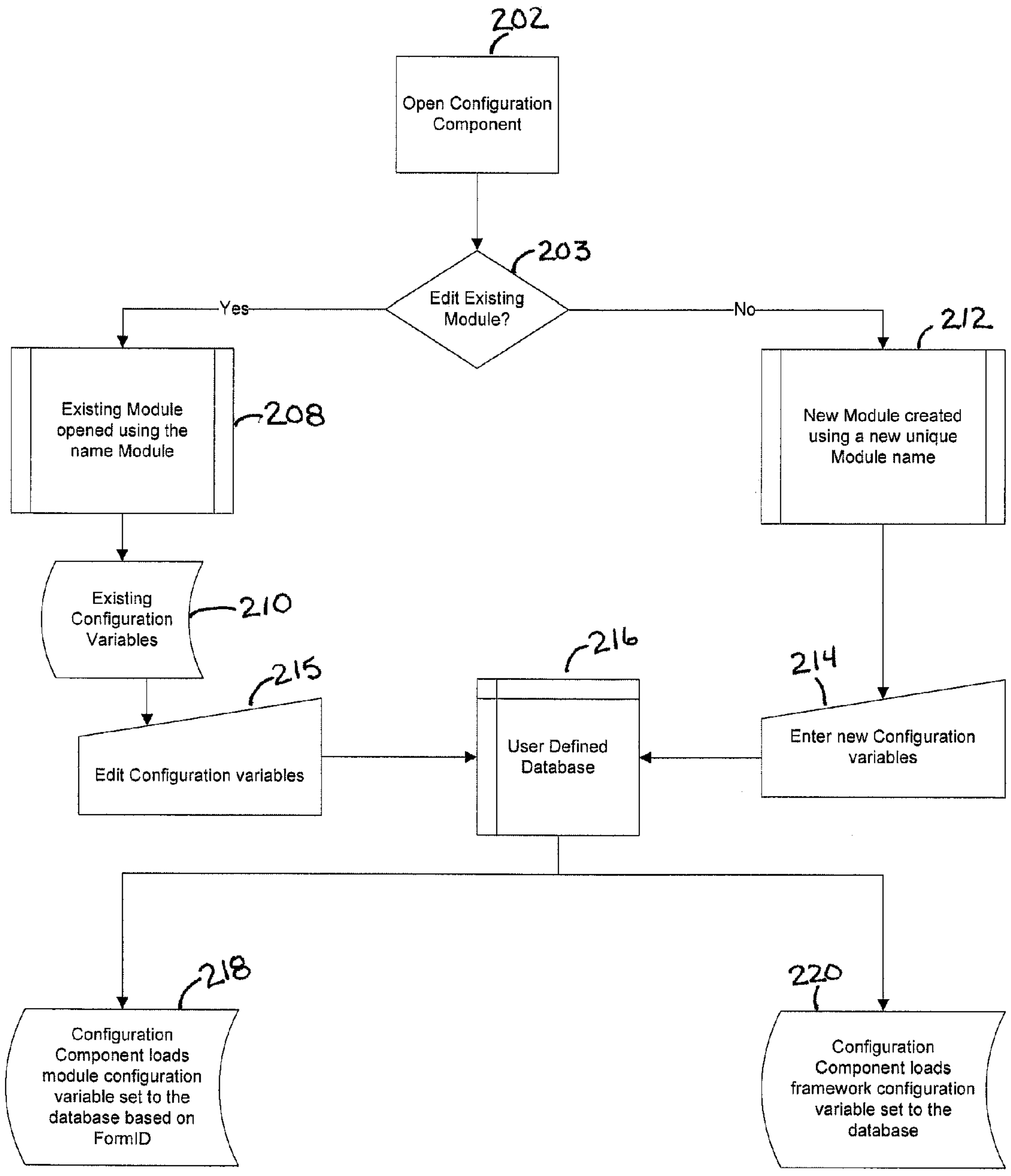

202
Open Configuration Component
203
Edit Existing Module?
Yes
No
208
Existing Module opened using the name Module
212
New Module created using a new unique Module name
210
Existing Configuration Variables
215
Edit Configuration variables
216
User Defined Database
214
Enter new Configuration variables
218
Configuration Component loads module configuration variable set to the database based on FormID
220
Configuration Component loads framework configuration variable set to the database

Fig. 2

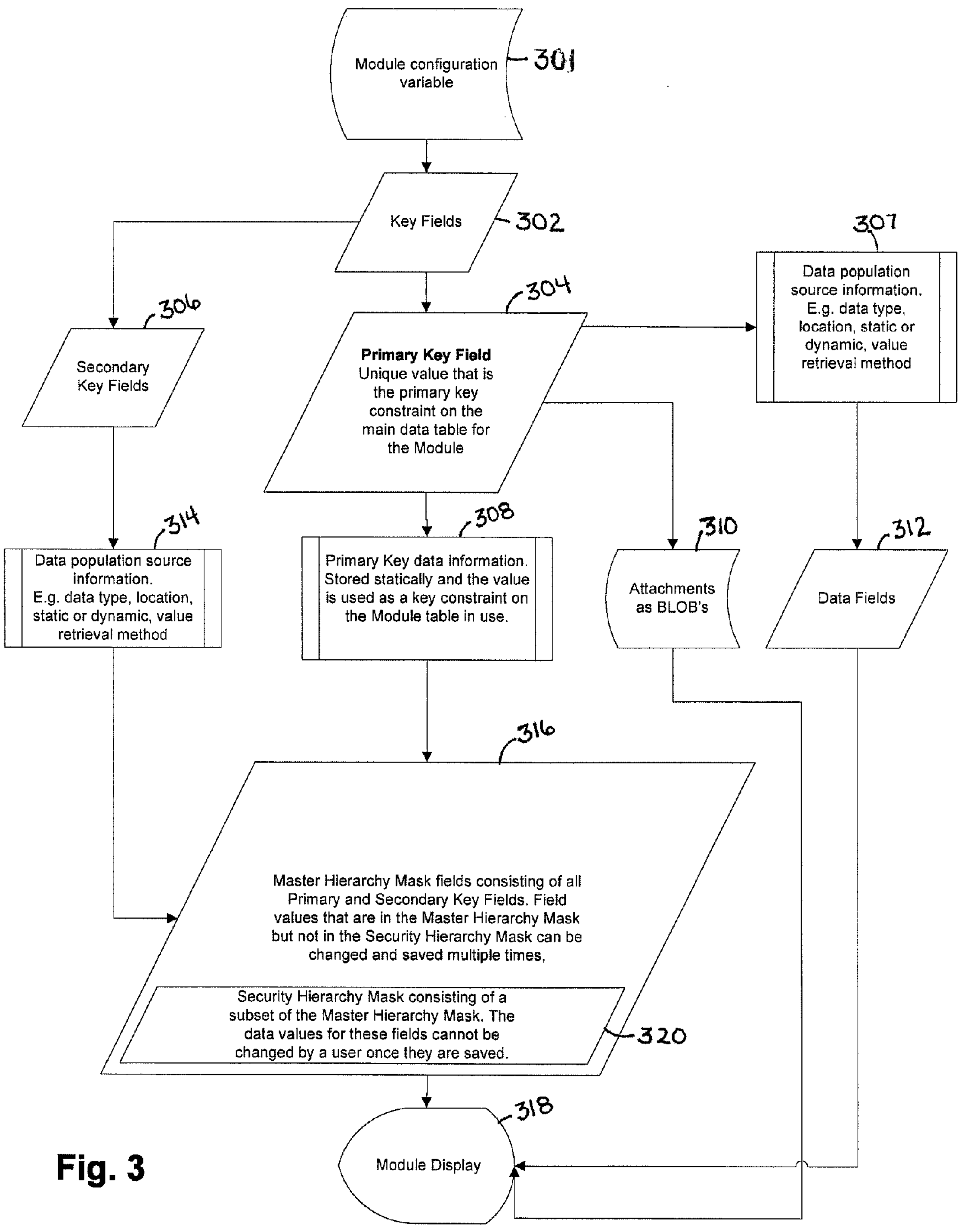

Module configuration variable
301
Key Fields
302
307
Data population source information. E.g. data type, location, static or dynamic, value retrieval method
304
306
Secondary Key Fields
Primary Key Field
Unique value that is the primary key constraint on the main data table for the Module
314
Data population source information. E.g. data type, location, static or dynamic, value retrieval method
308
Primary Key data information. Stored statically and the value is used as a key constraint on the Module table in use.
310
Attachments as BLOB's
312
Data Fields
316
Master Hierarchy Mask fields consisting of all Primary and Secondary Key Fields. Field values that are in the Master Hierarchy Mask but not in the Security Hierarchy Mask can be changed and saved multiple times,
Security Hierarchy Mask consisting of a subset of the Master Hierarchy Mask. The data values for these fields cannot be changed by a user once they are saved.
320
318
Module Display

Fig. 3

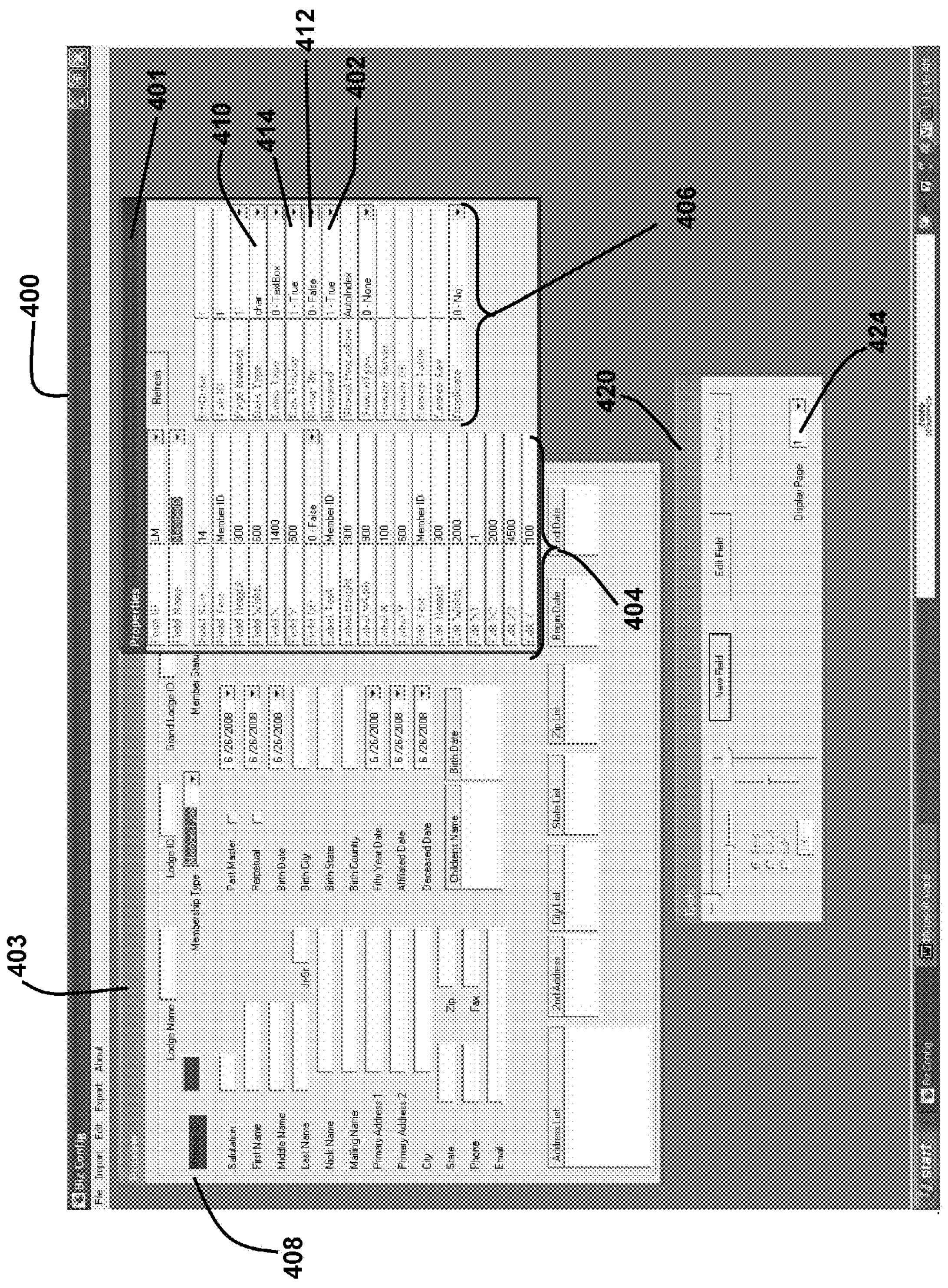
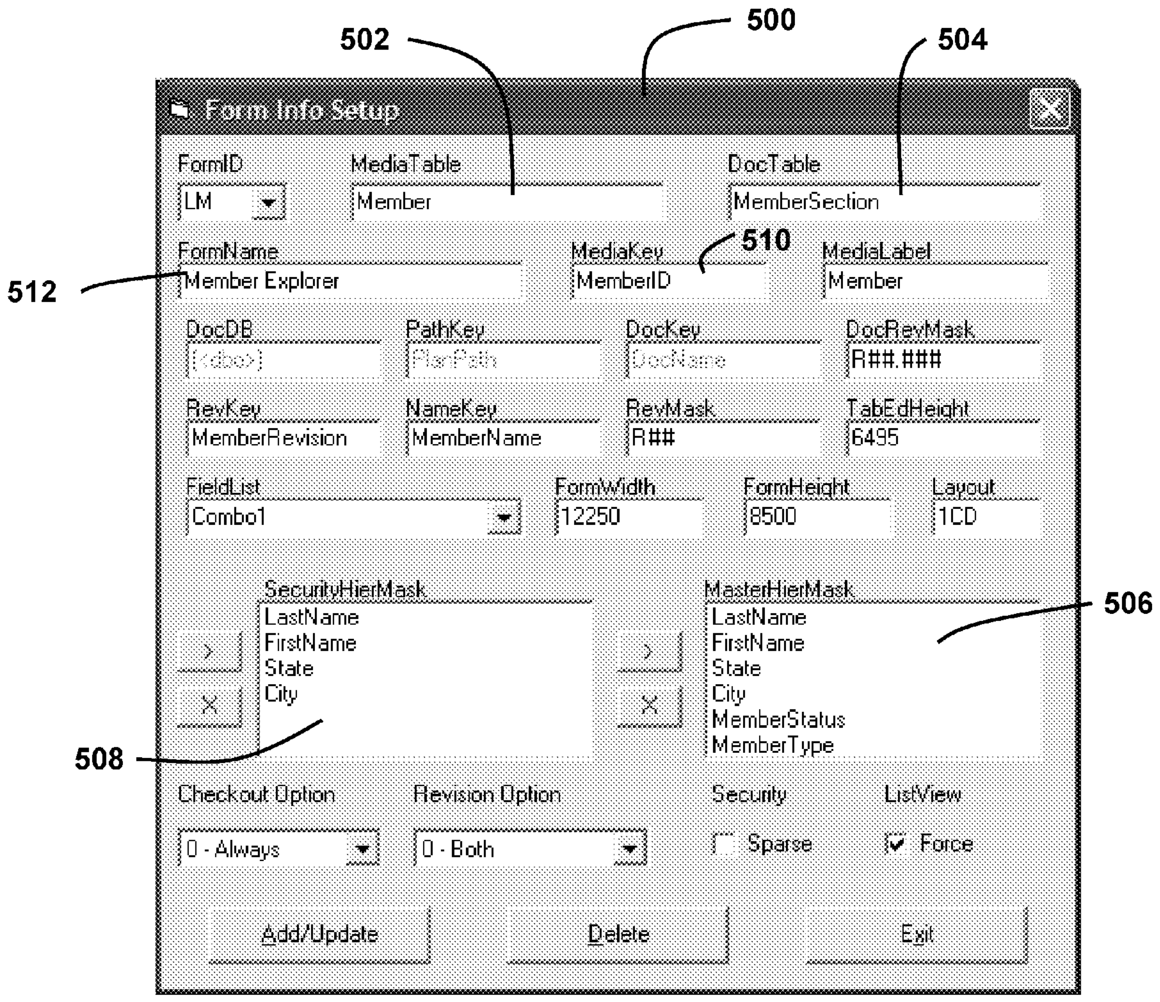

400
401
402
403
404
406
408
410
412
414
420
424
Properties
Refresh
New Field
Edit Field
Display Page
Lodge Name
Lodge ID
Grand Lodge ID
Membership Type
Salutation
First Name
Middle Name
Last Name
Nick Name
Mailing Name
Primary Address 1
Primary Address 2
City
State
Zip
Phone
Fax
Email
Past Master
Perpetual
Birth Date
Birth City
Birth State
Birth County
Fifty Year Date
Affiliated Date
Deceased Date
Childrens Name
Address List
2nd Address
City List
States List
Zip List
Begin Date
End Date

Fig. 4



FormID: LM
MediaTable: Member
DocTable: MemberSection
FormName: Member Explorer
MediaKey: MemberID
MediaLabel: Member
DocDB: [<dbo>]
PathKey: PlanPath
DocKey: DocName
DocRevMask: R##.###
RevKey: MemberRevision
NameKey: MemberName
RevMask: R##
TabEdHeight: 6495
FieldList: Combo1
FormWidth: 12250
FormHeight: 8500
Layout: 1CD

SecurityHierMask: LastName, FirstName, State, City

MasterHierMask: LastName, FirstName, State, City, MemberStatus, MemberType

Checkout Option: 0 - Always
Revision Option: 0 - Both
Security: Sparse [ ]
ListView: Force [✓]

Fig. 5

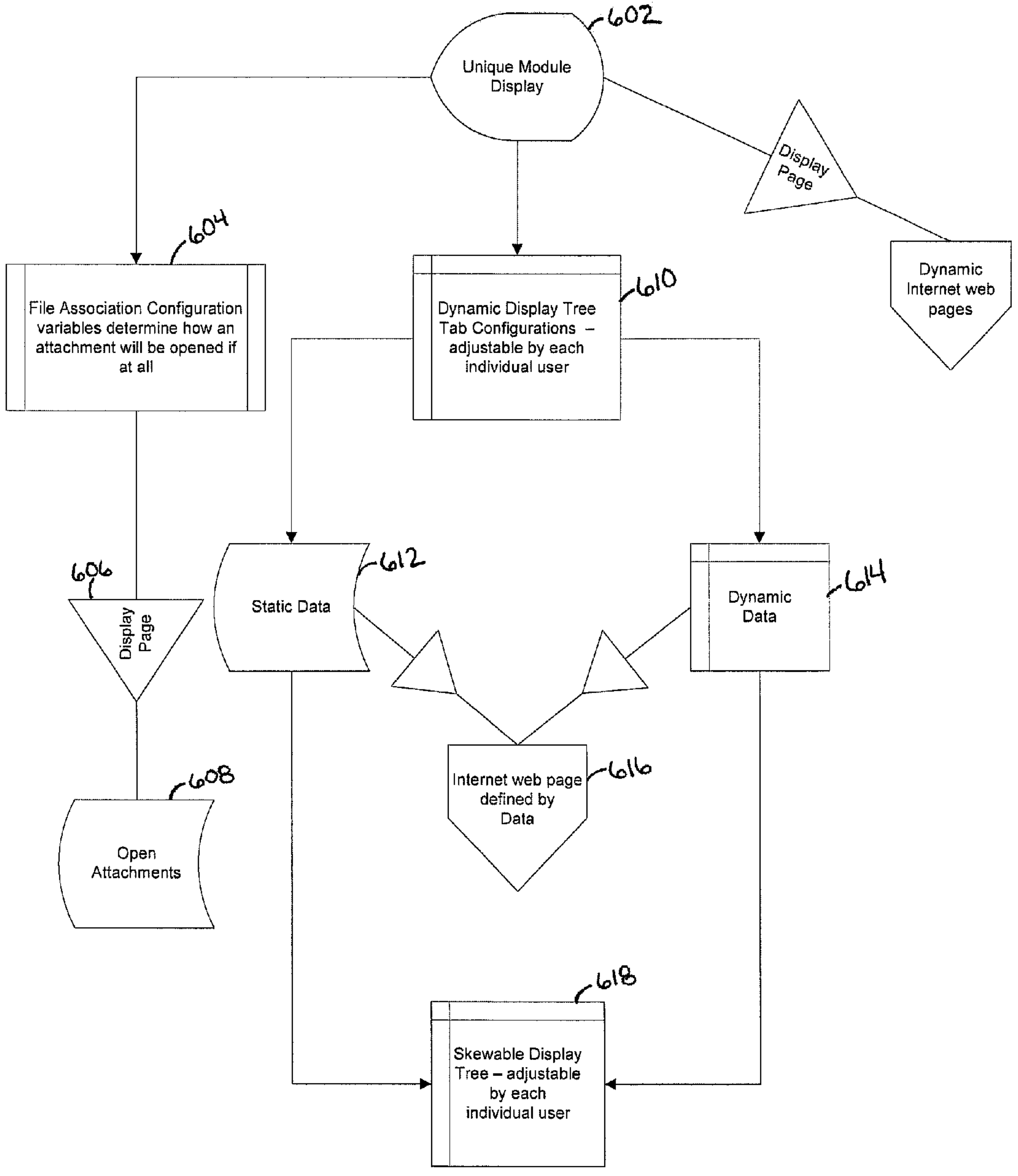
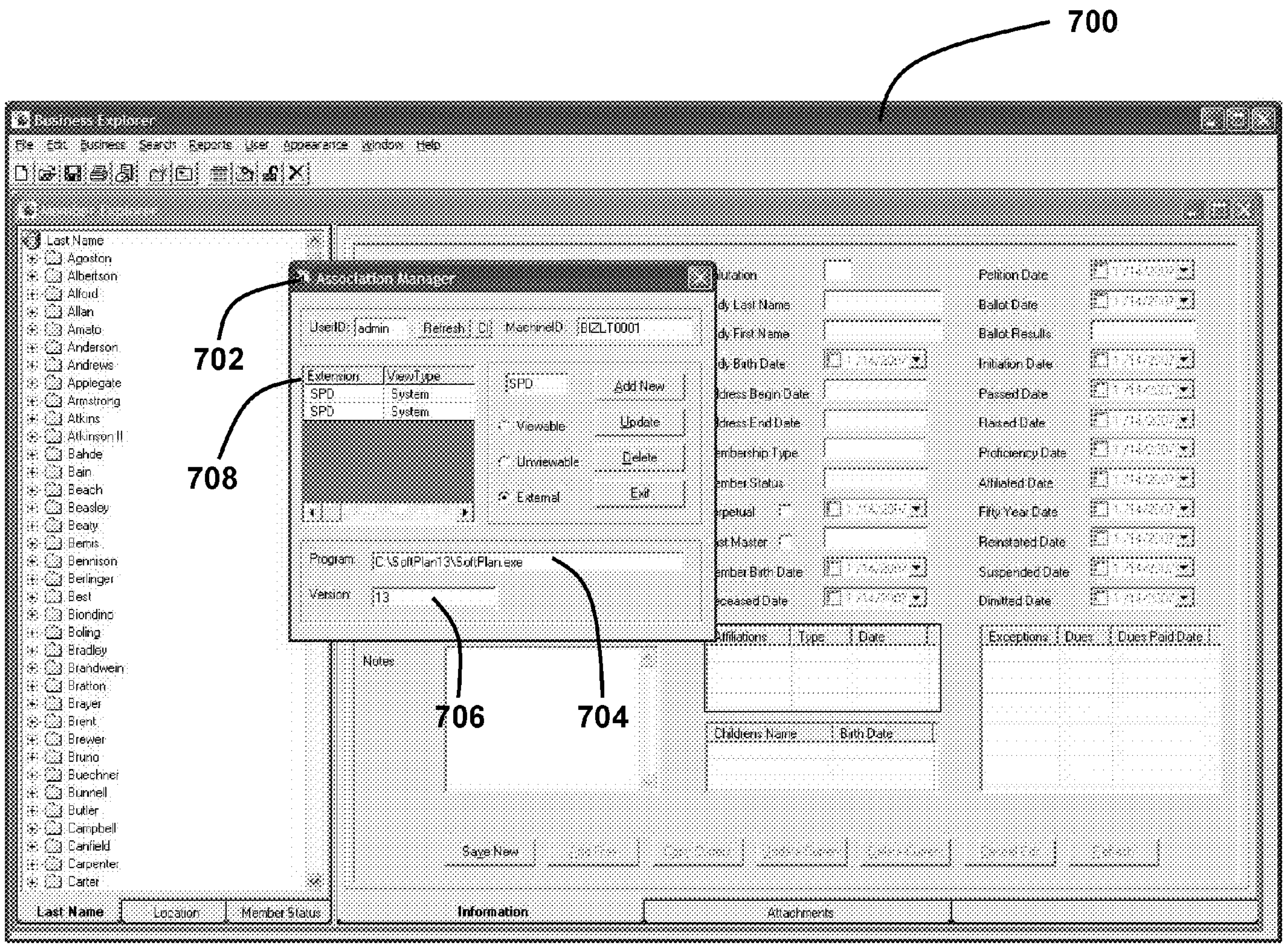

602
Unique Module Display
Display Page
Dynamic Internet web pages
604
File Association Configuration variables determine how an attachment will be opened if at all
610
Dynamic Display Tree Tab Configurations – adjustable by each individual user
606
Display Page
608
Open Attachments
612
Static Data
614
Dynamic Data
616
Internet web page defined by Data
618
Skewable Display Tree – adjustable by each individual user

Fig. 6



UserID: admin MachineID: BIZLT0001

Extension	ViewType
SPD	System
SPD	System

SPD

External

Program: C:\SoftPlan13\SoftPlan.exe

Version: 13

Fig. 7

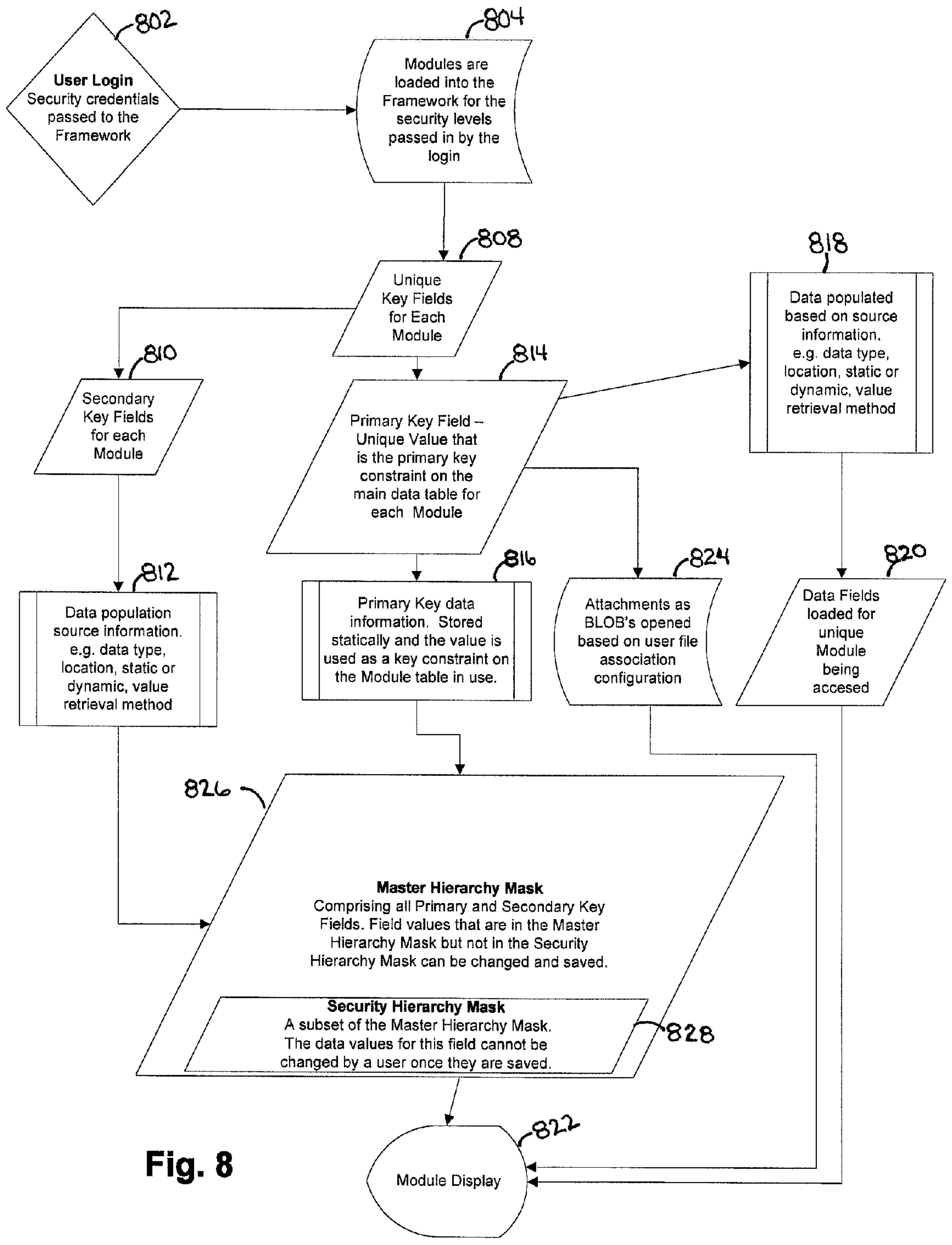
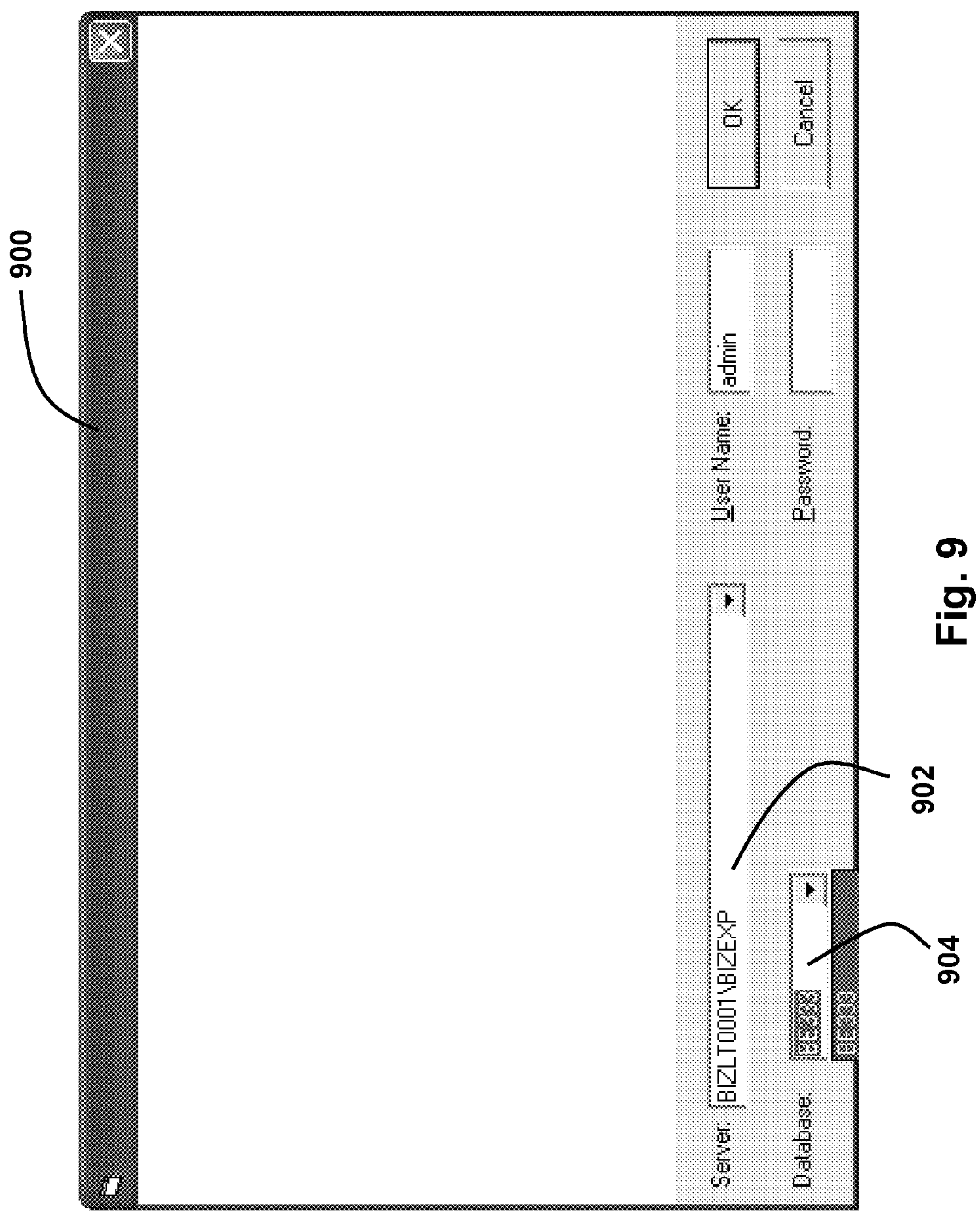

802
User Login
Security credentials passed to the Framework
804
Modules are loaded into the Framework for the security levels passed in by the login
808
Unique Key Fields for Each Module
810
Secondary Key Fields for each Module
814
Primary Key Field – Unique Value that is the primary key constraint on the main data table for each Module
818
Data populated based on source information. e.g. data type, location, static or dynamic, value retrieval method
812
Data population source information. e.g. data type, location, static or dynamic, value retrieval method
816
Primary Key data information. Stored statically and the value is used as a key constraint on the Module table in use.
824
Attachments as BLOB's opened based on user file association configuration
820
Data Fields loaded for unique Module being accesed
826
Master Hierarchy Mask
Comprising all Primary and Secondary Key Fields. Field values that are in the Master Hierarchy Mask but not in the Security Hierarchy Mask can be changed and saved.
Security Hierarchy Mask
A subset of the Master Hierarchy Mask. The data values for this field cannot be changed by a user once they are saved.
828
822
Module Display

Fig. 8



Server: BIZLT0001\BIZEXP

Database:

User Name: admin

Password:

Fig. 9

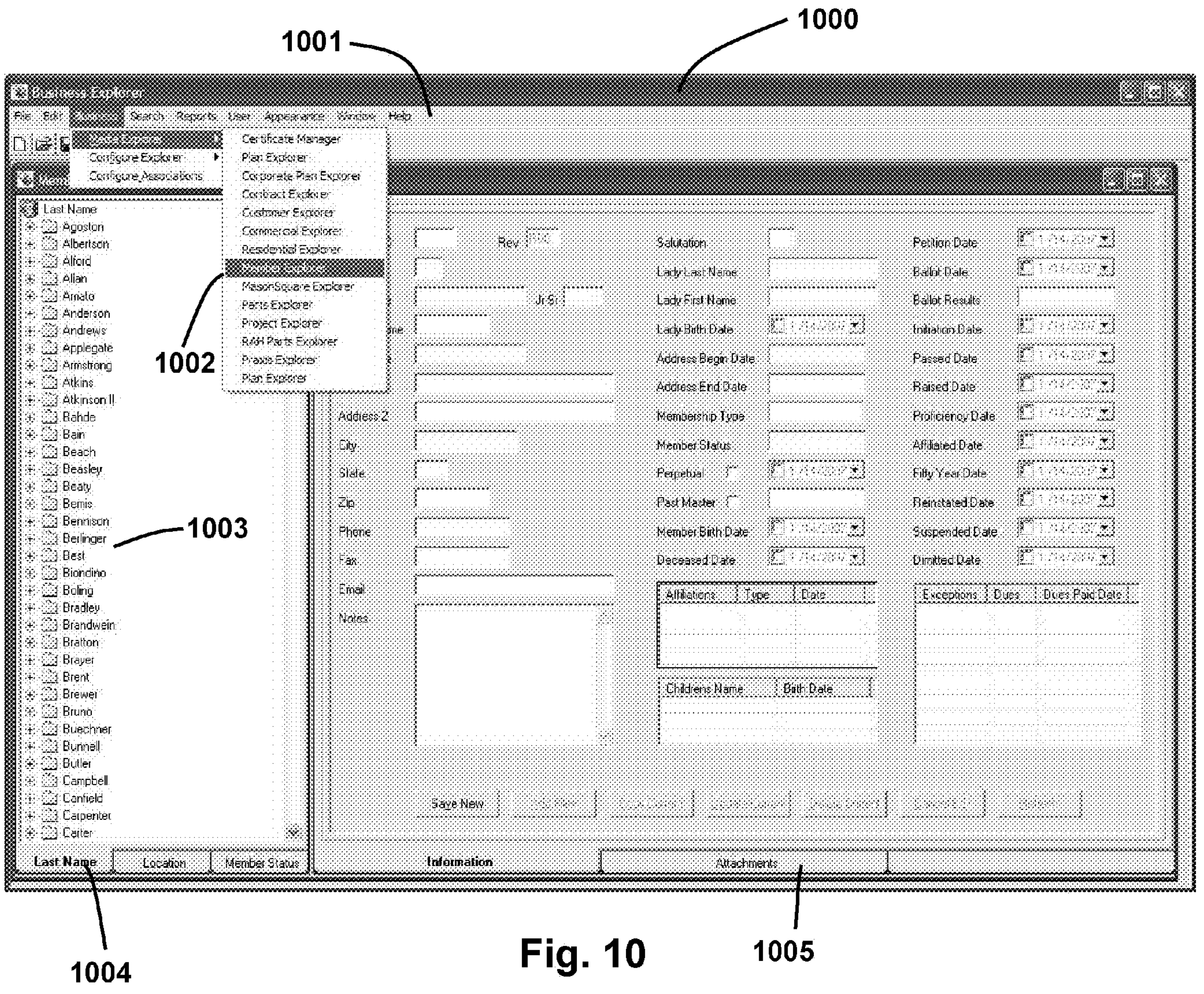

1000
1001
1002
1003
1004
1005
Business Explorer
File Edit Search Reports User Appearance Window Help
Configure Explorer
Configure Associations
Certificate Manager
Plan Explorer
Corporate Plan Explorer
Contract Explorer
Customer Explorer
Commercial Explorer
Residential Explorer
MasonSquare Explorer
Parts Explorer
Project Explorer
RAH Parts Explorer
Praxis Explorer
Plan Explorer
Last Name
Agoston
Albertson
Alford
Allan
Amato
Anderson
Andrews
Applegate
Armstrong
Atkins
Atkinson II
Bahde
Bain
Beach
Beasley
Beaty
Bemis
Bennison
Berlinger
Best
Biondino
Boling
Bradley
Brandwein
Bratton
Brayer
Brent
Brewer
Bruno
Buechner
Bunnell
Butler
Campbell
Canfield
Carpenter
Carter
Last Name
Location
Member Status
Rev.
Jr Sr
Address 2
City
State
Zip
Phone
Fax
Email
Notes
Salutation
Lady Last Name
Lady First Name
Lady Birth Date
Address Begin Date
Address End Date
Membership Type
Member Status
Perpetual
Past Master
Member Birth Date
Deceased Date
Affiliations
Type
Date
Childrens Name
Birth Date
Petition Date
Ballot Date
Ballot Results
Initiation Date
Passed Date
Raised Date
Proficiency Date
Affiliated Date
Fifty Year Date
Reinstated Date
Suspended Date
Dimitted Date
Exceptions
Dues
Dues Paid Date
Save New
Information
Attachments

Fig. 10

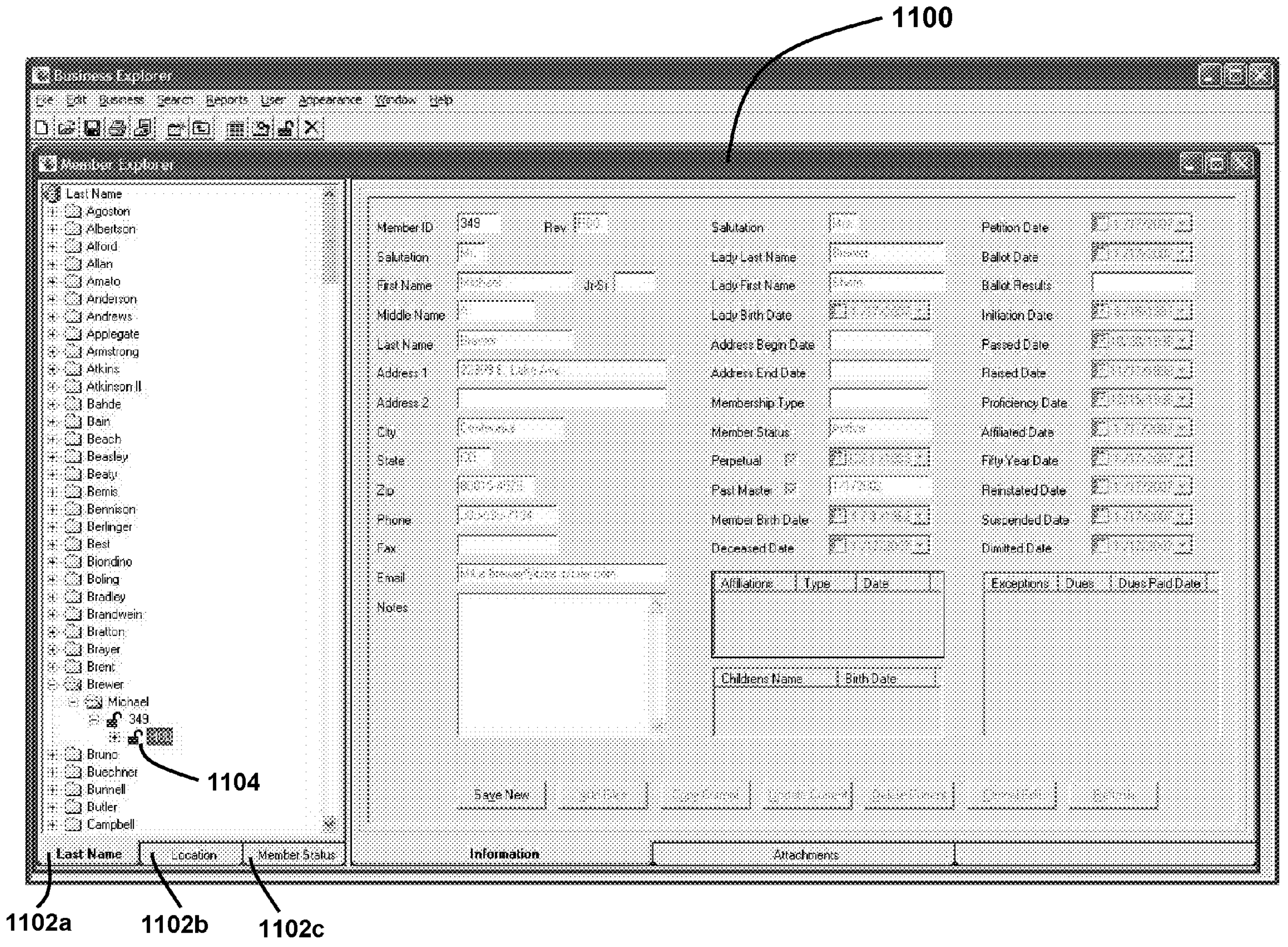

1100
Business Explorer
File Edit Business Search Reports User Appearance Window Help
Member Explorer
Last Name
Agoston
Albertson
Alford
Allan
Amato
Anderson
Andrews
Applegate
Armstrong
Atkins
Atkinson II
Bahde
Bain
Beach
Beasley
Beaty
Bemis
Bennison
Berlinger
Best
Biondino
Boling
Bradley
Brandwein
Bratton
Brayer
Brent
Brewer
Michael
349
Bruno
Buechner
Bunnell
Butler
Campbell
1104
Last Name
Location
Member Status
1102a
1102b
1102c
Member ID
349
Rev
Salutation
First Name
Jr-Sr
Middle Name
Last Name
Address 1
Address 2
City
State
Zip
Phone
Fax
Email
Notes
Salutation
Lady Last Name
Lady First Name
Lady Birth Date
Address Begin Date
Address End Date
Membership Type
Member Status
Perpetual
Past Master
Member Birth Date
Deceased Date
Affiliations
Type
Date
Childrens Name
Birth Date
Petition Date
Ballot Date
Ballot Results
Initiation Date
Passed Date
Raised Date
Proficiency Date
Affiliated Date
Fifty Year Date
Reinstated Date
Suspended Date
Dimitted Date
Exceptions
Dues
Dues Paid Date
Save New
Information
Attachments

Fig. 11

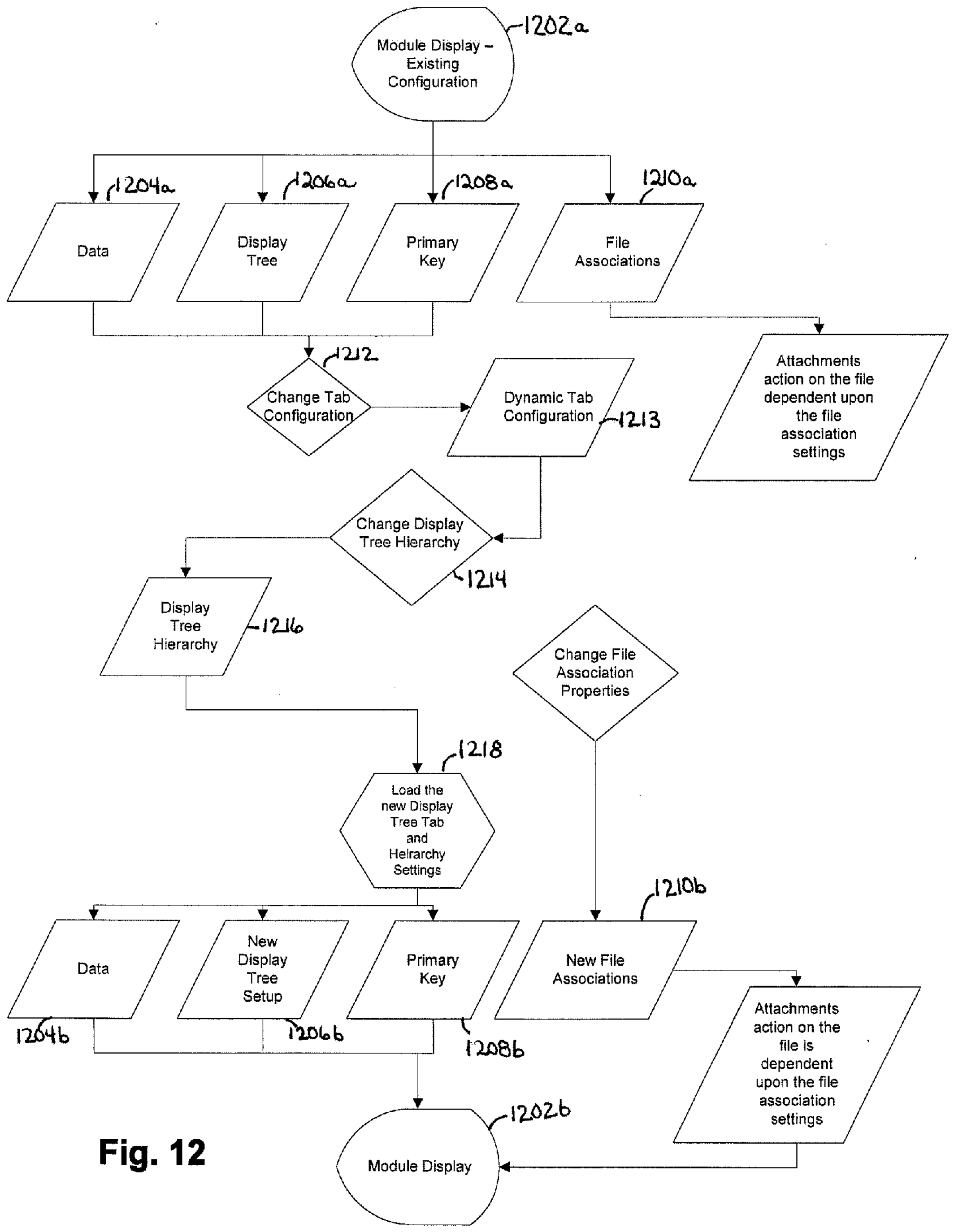
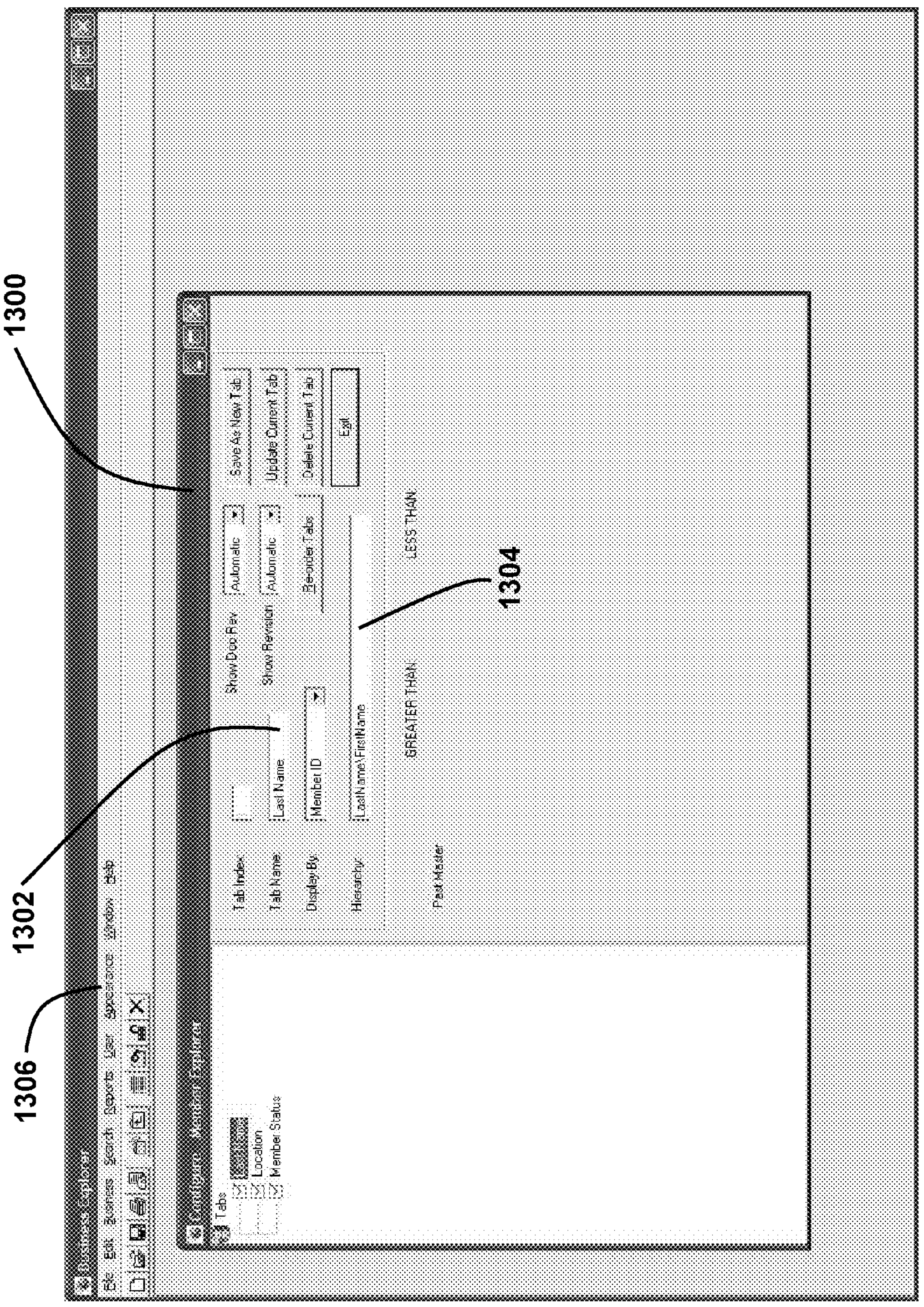

Module Display – Existing Configuration
1202a
Data
1204a
Display Tree
1206a
Primary Key
1208a
File Associations
1210a
Attachments action on the file dependent upon the file association settings
Change Tab Configuration
1212
Dynamic Tab Configuration
1213
Change Display Tree Hierarchy
1214
Display Tree Hierarchy
1216
Change File Association Properties
Load the new Display Tree Tab and Heirarchy Settings
1218
Data
1204b
New Display Tree Setup
1206b
Primary Key
1208b
New File Associations
1210b
Attachments action on the file is dependent upon the file association settings
Module Display
1202b

Fig. 12



Tab Index:
Tab Name: Last Name
Display By: Member ID
Hierarchy: LastName\FirstName
Show Doc Rev: Automatic
Show Revision: Automatic

Fig. 13

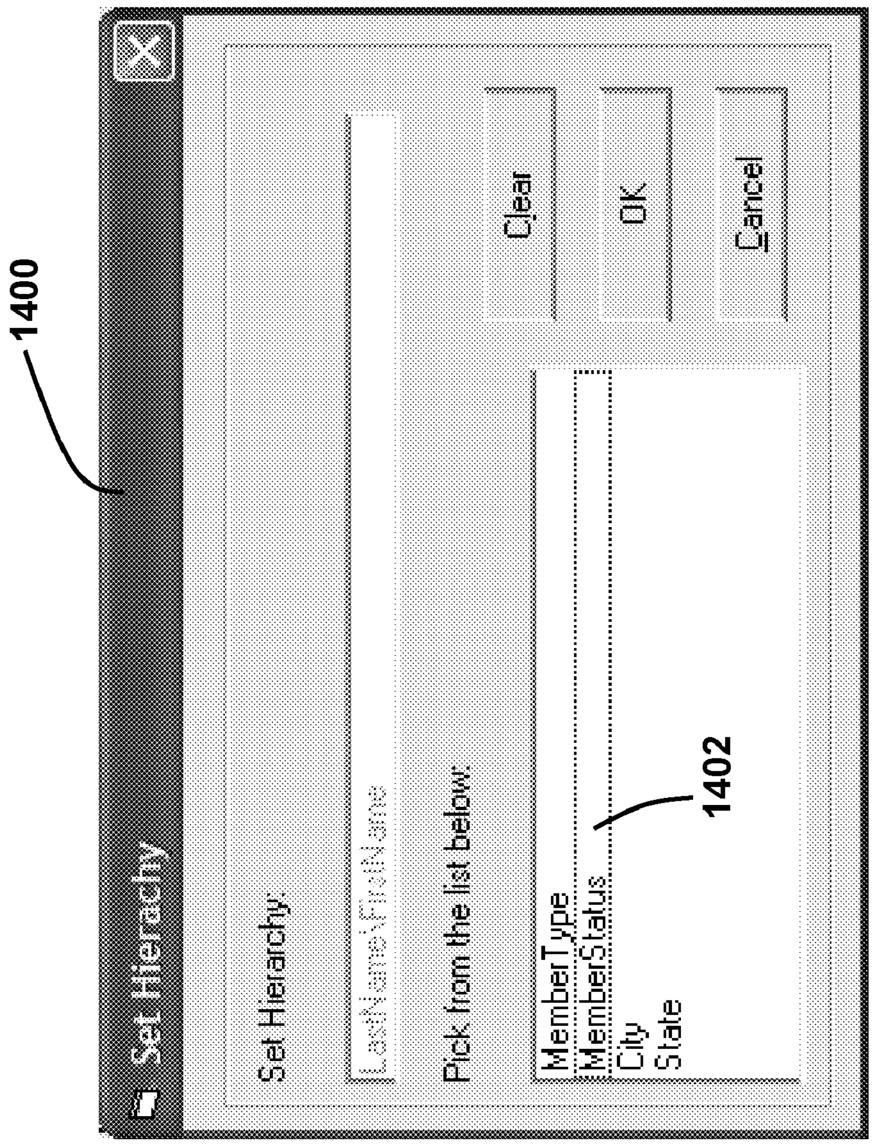
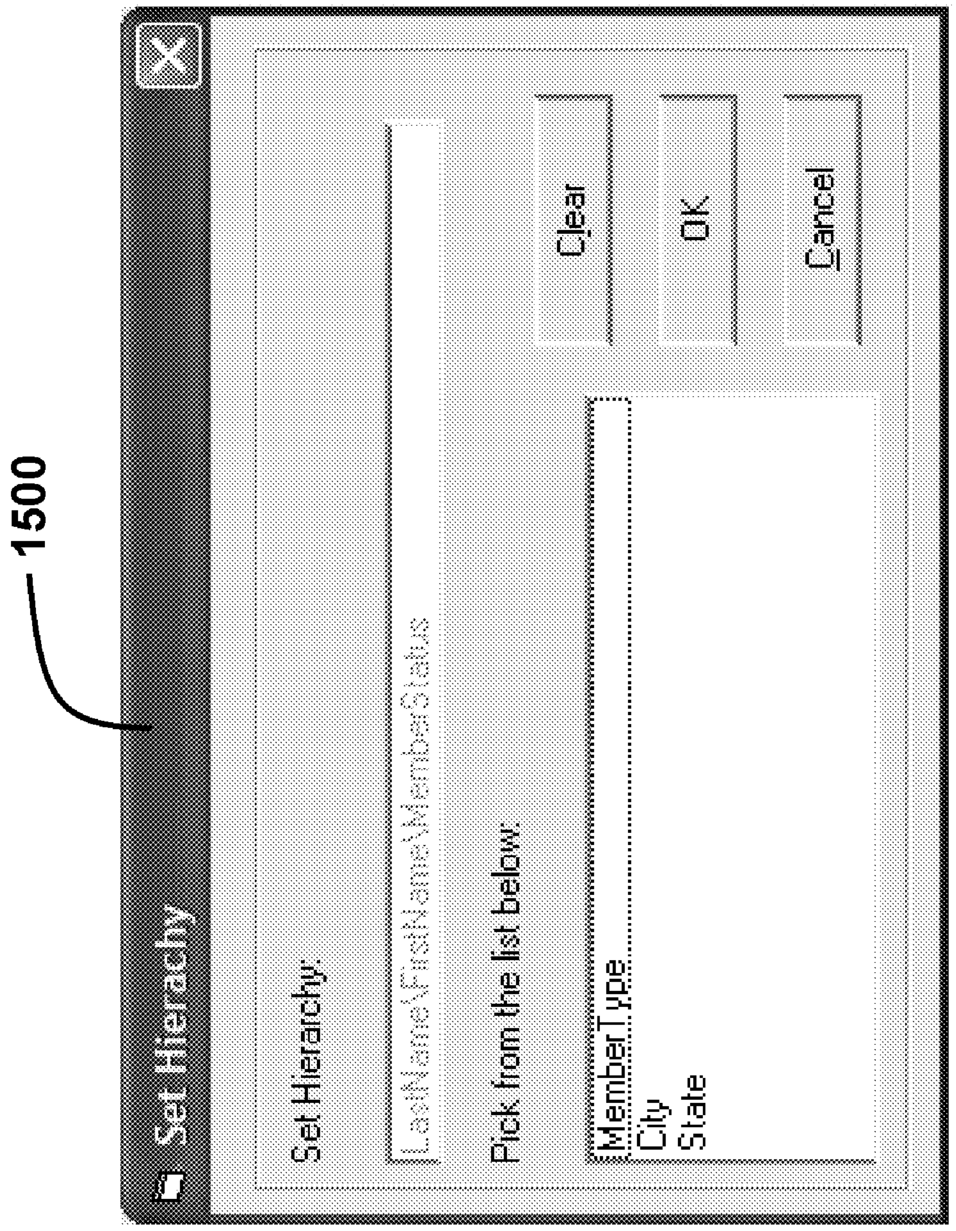

1400
Set Hierachy
Set Hierachy:
LastName\FirstName
Pick from the list below:
MemberType
MemberStatus
City
State
1402
Clear
OK
Cancel

Fig. 14

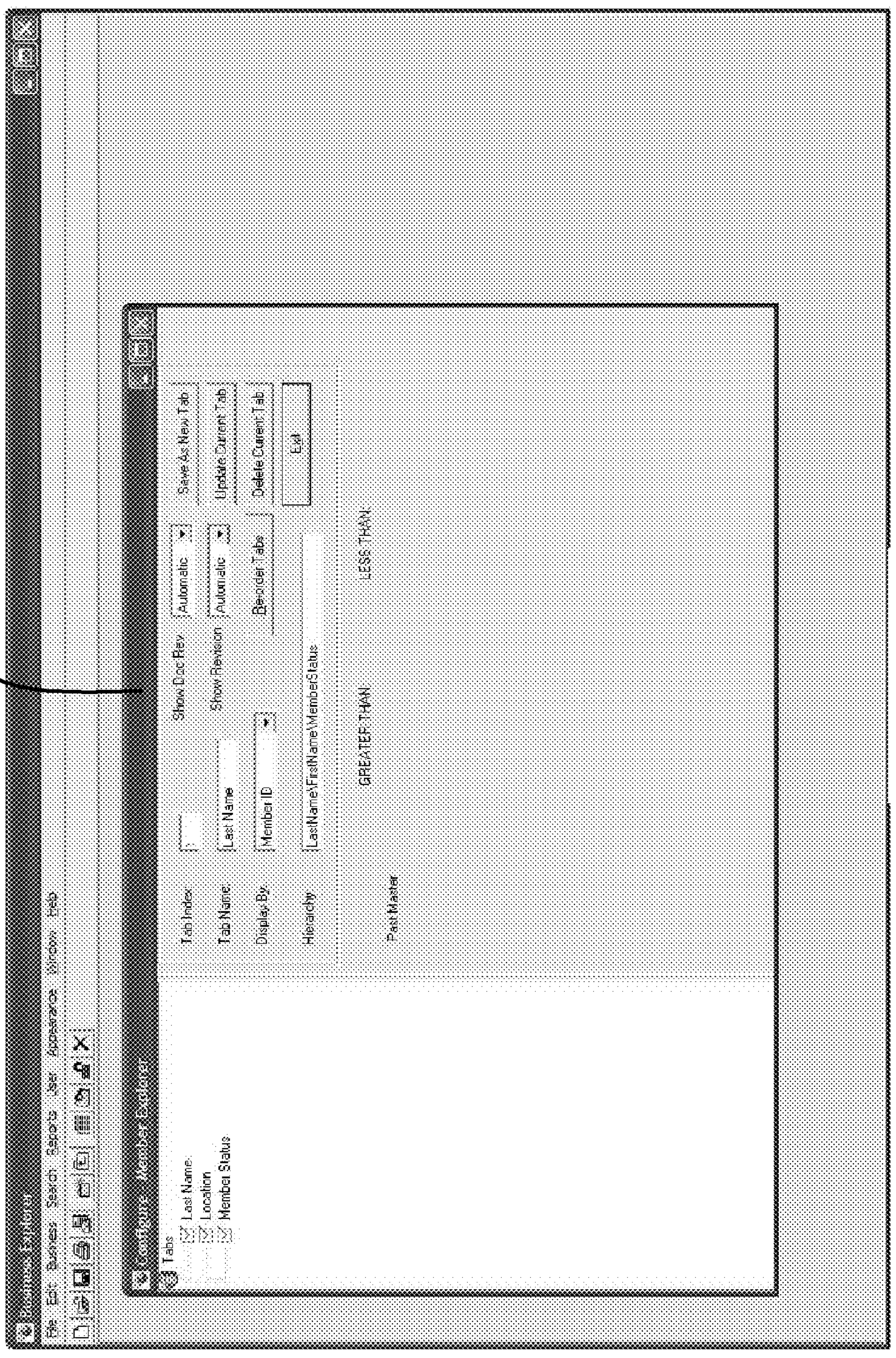


LastName\FirstName\MemberStatus

MemberType
City
State

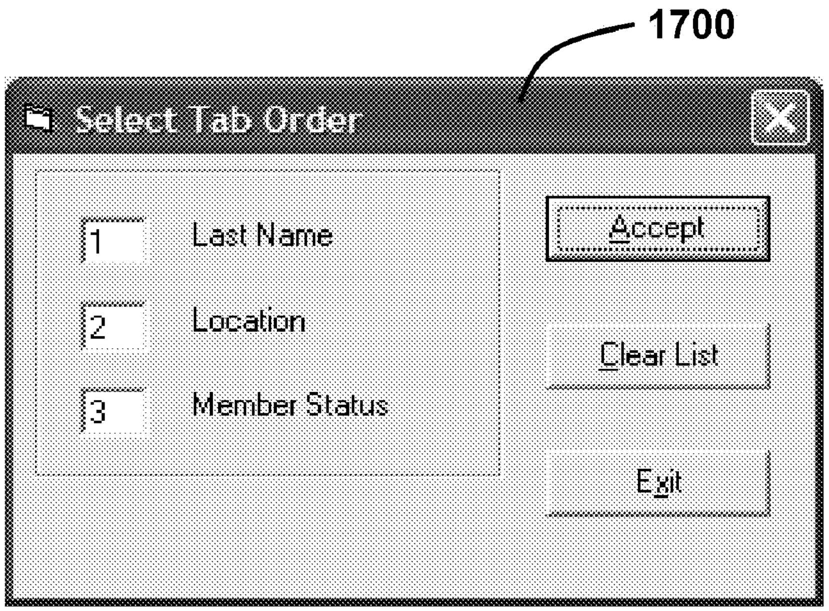
Fig. 15

1600



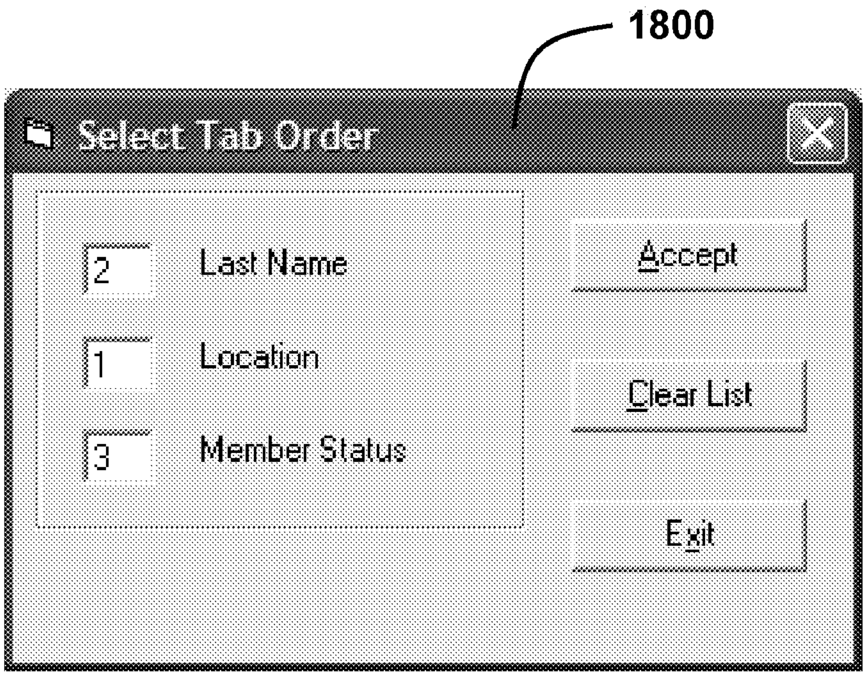
Tab Index:
Tab Name: Last Name
Display By: Member ID
Hierarchy: LastName\FirstName\MemberStatus
Show Doc Rev: Automatic
Show Revision: Automatic
GREATER THAN
LESS THAN
Past Master

Fig. 16



1 Last Name
2 Location
3 Member Status

Fig. 17



2 Last Name
1 Location
3 Member Status

Fig. 18

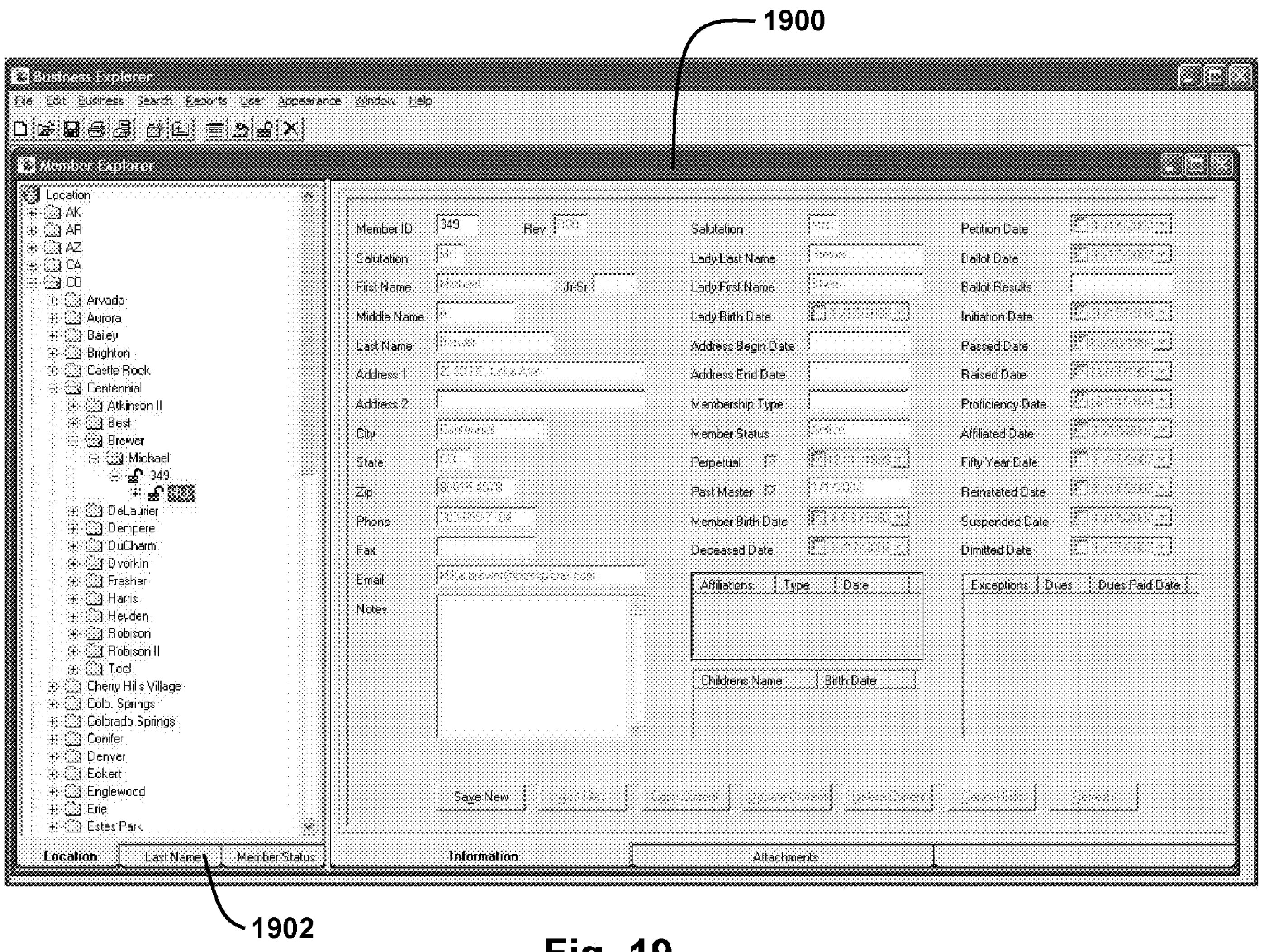

1900
Business Explorer
File Edit Business Search Reports User Appearance Window Help
Member Explorer
Location
AK
AR
AZ
CA
CO
Arvada
Aurora
Bailey
Brighton
Castle Rock
Centennial
Atkinson II
Best
Brewer
Michael
349
DeLaurier
Dempere
DuCharm
Dvorkin
Frasher
Harris
Heyden
Robison
Robison II
Tool
Cherry Hills Village
Colo. Springs
Colorado Springs
Conifer
Denver
Eckert
Englewood
Erie
Estes Park
Location
Last Name
Member Status
Member ID 349
Rev
Salutation
First Name
Jr/Sr
Middle Name
Last Name
Address 1
Address 2
City
State
Zip
Phone
Fax
Email
Notes
Salutation
Lady Last Name
Lady First Name
Lady Birth Date
Address Begin Date
Address End Date
Membership Type
Member Status
Perpetual
Past Master
Member Birth Date
Deceased Date
Affiliations
Type
Date
Childrens Name
Birth Date
Petition Date
Ballot Date
Ballot Results
Initiation Date
Passed Date
Raised Date
Proficiency Date
Affiliated Date
Fifty Year Date
Reinstated Date
Suspended Date
Dimitted Date
Exceptions
Dues
Dues Paid Date
Save New
Information
Attachments
1902

Fig. 19

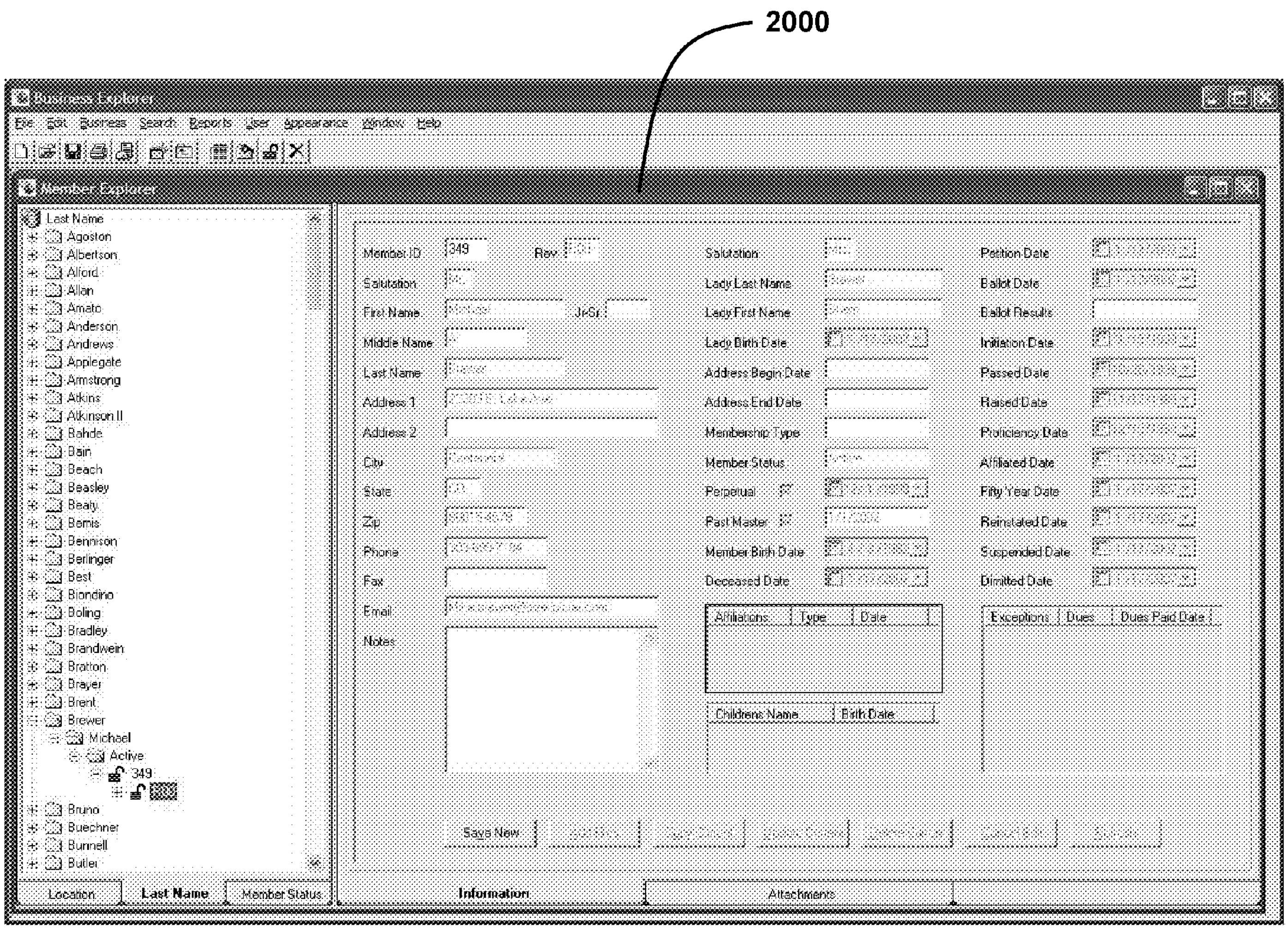

2000
Business Explorer
File Edit Business Search Reports User Appearance Window Help
Member Explorer
Last Name
Agoston
Albertson
Alford
Allan
Amato
Anderson
Andrews
Applegate
Armstrong
Atkins
Atkinson II
Bahde
Bain
Beach
Beasley
Beaty
Bemis
Bennison
Berlinger
Best
Blondino
Boling
Bradley
Brandwein
Bratton
Brayer
Brent
Brewer
Michael
Active
349
Bruno
Buechner
Burnell
Butler
Location
Last Name
Member Status
Member ID
349
Rev
Salutation
First Name
Jr-Sr
Middle Name
Last Name
Address 1
Address 2
City
State
Zip
Phone
Fax
Email
Notes
Salutation
Lady Last Name
Lady First Name
Lady Birth Date
Address Begin Date
Address End Date
Membership Type
Member Status
Perpetual
Past Master
Member Birth Date
Deceased Date
Affiliations
Type
Date
Childrens Name
Birth Date
Petition Date
Ballot Date
Ballot Results
Initiation Date
Passed Date
Raised Date
Proficiency Date
Affiliated Date
Fifty Year Date
Reinstated Date
Suspended Date
Dimitted Date
Exceptions
Dues
Dues Paid Date
Save New
Information
Attachments

Fig. 20

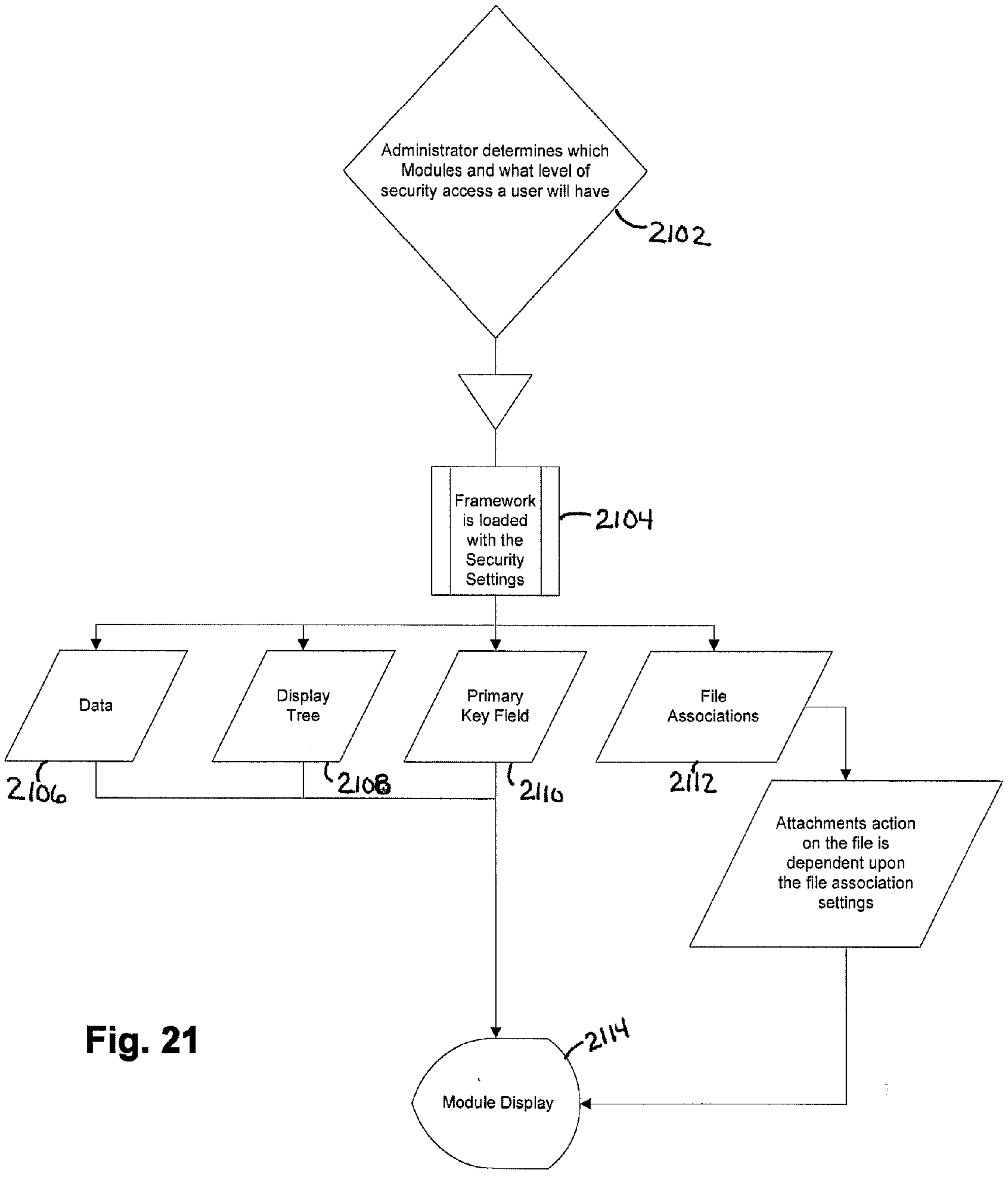
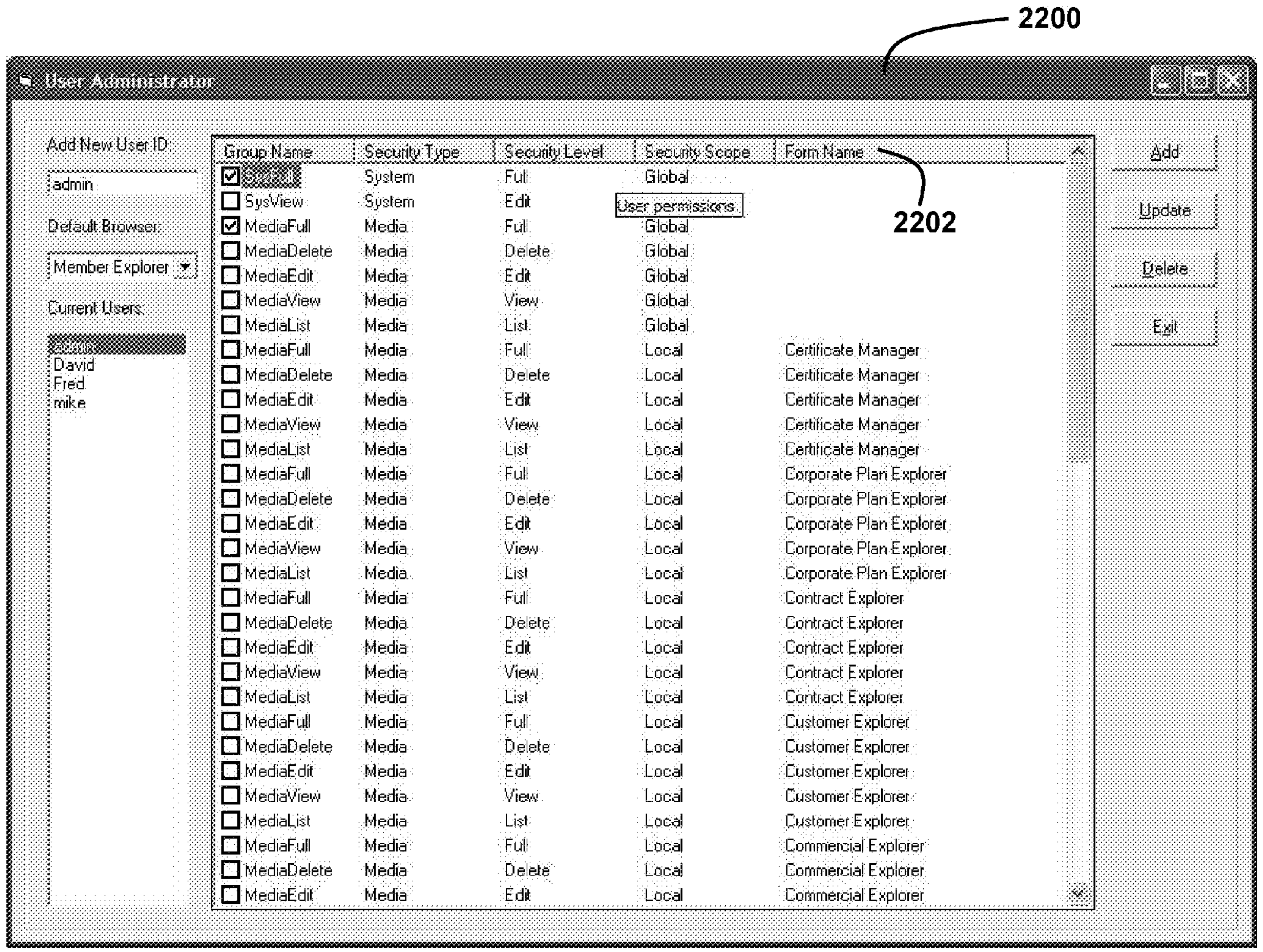

Administrator determines which Modules and what level of security access a user will have
2102
Framework is loaded with the Security Settings
2104
Data
2106
Display Tree
2108
Primary Key Field
2110
File Associations
2112
Attachments action on the file is dependent upon the file association settings
Module Display
2114

Fig. 21



Add New User ID:
admin
Default Browser:
Member Explorer
Current Users:
[illegible]
David
Fred
mike

	Group Name	Security Type	Security Level	Security Scope	Form Name
☑	[illegible]	System	Full	Global	
☐	SysView	System	Edit		
☑	MediaFull	Media	Full	Global	
☐	MediaDelete	Media	Delete	Global	
☐	MediaEdit	Media	Edit	Global	
☐	MediaView	Media	View	Global	
☐	MediaList	Media	List	Global	
☐	MediaFull	Media	Full	Local	Certificate Manager
☐	MediaDelete	Media	Delete	Local	Certificate Manager
☐	MediaEdit	Media	Edit	Local	Certificate Manager
☐	MediaView	Media	View	Local	Certificate Manager
☐	MediaList	Media	List	Local	Certificate Manager
☐	MediaFull	Media	Full	Local	Corporate Plan Explorer
☐	MediaDelete	Media	Delete	Local	Corporate Plan Explorer
☐	MediaEdit	Media	Edit	Local	Corporate Plan Explorer
☐	MediaView	Media	View	Local	Corporate Plan Explorer
☐	MediaList	Media	List	Local	Corporate Plan Explorer
☐	MediaFull	Media	Full	Local	Contract Explorer
☐	MediaDelete	Media	Delete	Local	Contract Explorer
☐	MediaEdit	Media	Edit	Local	Contract Explorer
☐	MediaView	Media	View	Local	Contract Explorer
☐	MediaList	Media	List	Local	Contract Explorer
☐	MediaFull	Media	Full	Local	Customer Explorer
☐	MediaDelete	Media	Delete	Local	Customer Explorer
☐	MediaEdit	Media	Edit	Local	Customer Explorer
☐	MediaView	Media	View	Local	Customer Explorer
☐	MediaList	Media	List	Local	Customer Explorer
☐	MediaFull	Media	Full	Local	Commercial Explorer
☐	MediaDelete	Media	Delete	Local	Commercial Explorer
☐	MediaEdit	Media	Edit	Local	Commercial Explorer

Fig. 22

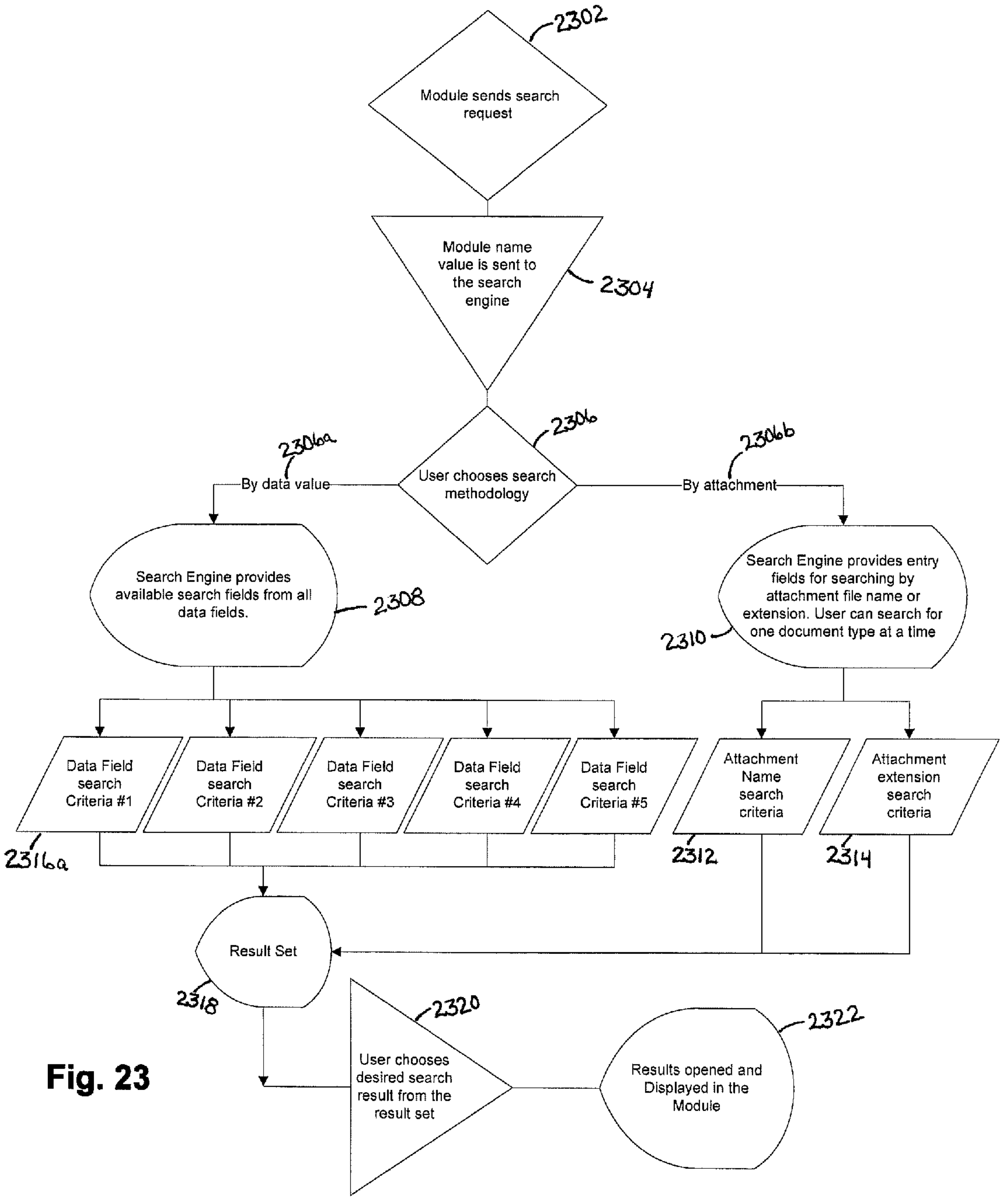

2302
Module sends search request
Module name value is sent to the search engine
2304
2306
User chooses search methodology
2306a
By data value
2306b
By attachment
Search Engine provides available search fields from all data fields.
2308
Search Engine provides entry fields for searching by attachment file name or extension. User can search for one document type at a time
2310
Data Field search Criteria #1
Data Field search Criteria #2
Data Field search Criteria #3
Data Field search Criteria #4
Data Field search Criteria #5
Attachment Name search criteria
Attachment extension search criteria
2316a
2312
2314
Result Set
2318
2320
User chooses desired search result from the result set
2322
Results opened and Displayed in the Module

Fig. 23

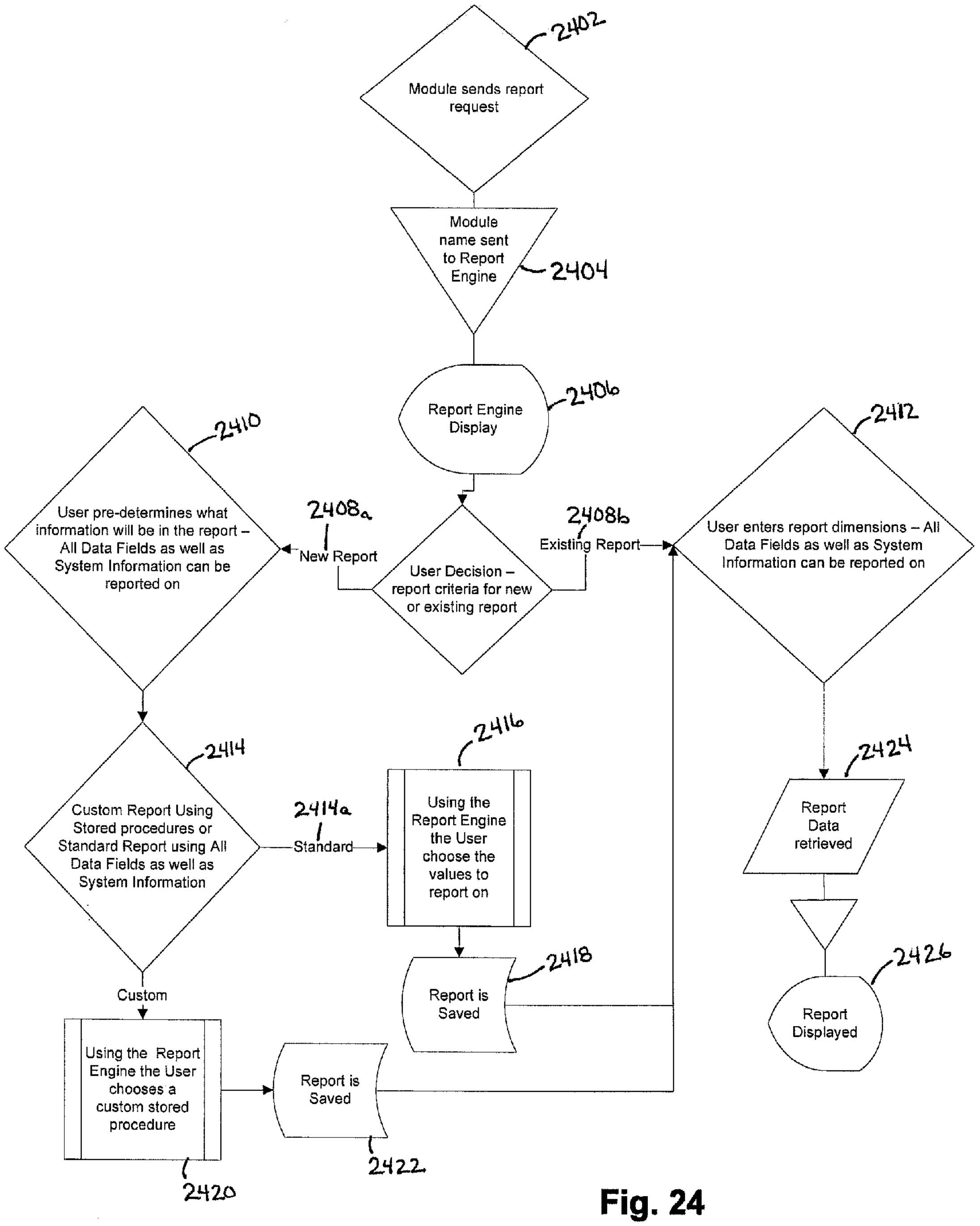

Module sends report request
2402
Module name sent to Report Engine
2404
Report Engine Display
2406
User pre-determines what information will be in the report – All Data Fields as well as System Information can be reported on
2410
2408a
New Report
User Decision – report criteria for new or existing report
2408b
Existing Report
User enters report dimensions – All Data Fields as well as System Information can be reported on
2412
Custom Report Using Stored procedures or Standard Report using All Data Fields as well as System Information
2414
2414a
Standard
Using the Report Engine the User choose the values to report on
2416
Report is Saved
2418
Report Data retrieved
2424
Report Displayed
2426
Custom
Using the Report Engine the User chooses a custom stored procedure
2420
Report is Saved
2422

Fig. 24

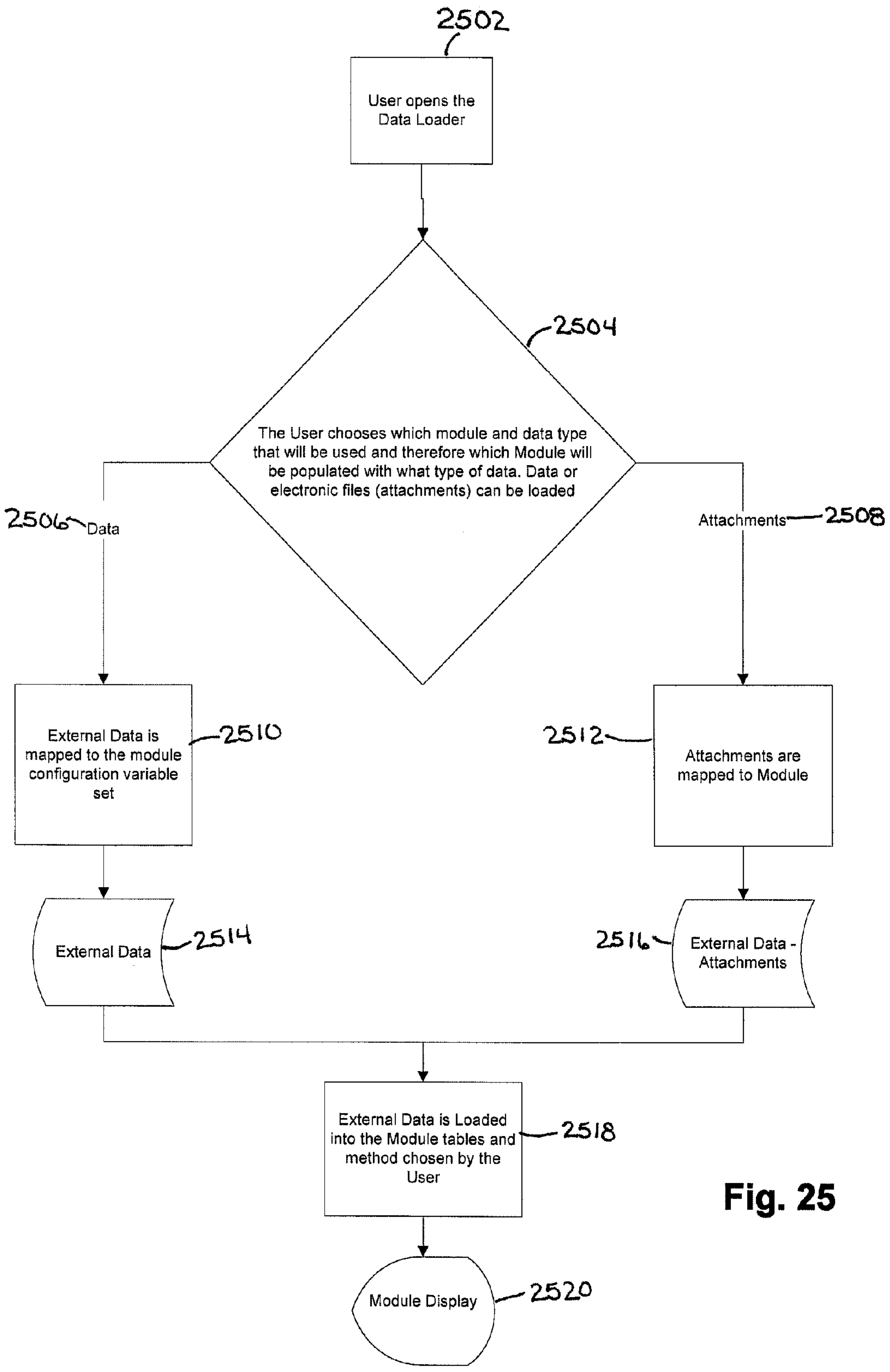

2502
User opens the Data Loader
2504
The User chooses which module and data type that will be used and therefore which Module will be populated with what type of data. Data or electronic files (attachments) can be loaded
2506
Data
Attachments
2508
External Data is mapped to the module configuration variable set
2510
2512
Attachments are mapped to Module
External Data
2514
2516
External Data - Attachments
External Data is Loaded into the Module tables and method chosen by the User
2518
Module Display
2520

Fig. 25

VARIABLE DRIVEN METHOD AND SYSTEM FOR THE MANAGEMENT AND DISPLAY OF INFORMATION

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to U.S. Provisional Application No. 60/946,293 entitled VARIABLE DRIVEN INFORMATION MANAGEMENT SYSTEM AND APPLICATION BUILDER and filed Jun. 26, 2007, the disclosure of which is incorporated herein by reference in its entirety.

BACKGROUND OF THE INVENTION

[0002] 1. Field of the Invention

[0003] The present invention relates to a computer-implemented information management system and application builder and a method for managing and displaying information to a user. More particularly, the present invention relates to a computer-implemented information management and display system where data field configuration variables are used to create data fields. The use of configuration variables to create data fields enables a user to manipulate and display information based on the preferences that are set by that user or for that user.

[0004] 2. Description of Related Art

[0005] Conventional systems for organizing, controlling and manipulating files and other data on a computer system are typically implemented using a hierarchy-based display of the files, commonly referred to as a display tree or an explorer tree. Electronic files are typically stored on a computer system in folders and subfolders that are organized based upon a display tree hierarchy that is established by a system administrator and the established hierarchy cannot be manipulated by an end-user without altering the hierarchy that is displayed to other users of the system. Locating relevant information for collating, manipulating and/or reporting on the information can be a difficult and time consuming task, particularly when the information is located in disparate locations.

[0006] It would be useful to provide a means to allow a user to identify, organize and manipulate files or other data and information in a manner that is more user-friendly to the end user and reduces the amount of time and effort required by the user to identify the information.

[0007] Further, creating a new database and information management system (i.e., an application) for an entity generally requires building a new system from scratch, e.g., by writing new code, and such a system cannot be easily changed once it is created. It would be useful to provide a means to facilitate the rapid and economical creation of new information retrieval and management systems without requiring the writing of new code for each new system.

SUMMARY OF THE INVENTION

[0008] In one embodiment, a method for managing and displaying information is provided. The method can include the step of configuring a plurality of key data fields to create a key data field set, the key data field set including a primary key data field and a plurality of secondary key data fields. The configuration step can include assigning values to a configuration variable set that defines properties of the key data fields. A plurality of unique static data field values are assigned to the primary key data field, where each static data field value identifies a unique database record. The secondary data key fields can be populated with secondary key data field values. Information can then be displayed to a user on a graphical user interface (GUI), where the graphical user interface includes a skewable display tree whereby at least a portion of the secondary key data field values are displayed to the user in the skewable display tree.

[0009] In one aspect, the populating step comprises assigning a dynamic data field value to at least one secondary key data field. A stored procedure for computation of the dynamic data element can be linked to the secondary key data field. In another aspect, the populating step can include assigning a static data field value to at least one secondary key data field.

[0010] In another aspect, the step of displaying can include applying a hierarchy mask to the key field set to restrict the key fields that are displayed to a user on the graphical user interface. The hierarchy mask can be a security hierarchy mask that restricts the key fields that can be edited by a particular user on the graphical user interface.

[0011] In one aspect, the step of populating the secondary key data fields can include accepting input of data field values from a user through the graphical user interface. In another aspect, the step of populating the secondary key data fields with data can include importing data field values from a file.

[0012] In one aspect, the step of displaying information on a graphical user interface can include displaying a plurality of dynamic display tree configuration widgets, whereby a user can select the display hierarchy of secondary key data field values on the graphical user interface by selecting one of the display tree configuration widgets. The display tree configuration widget can be, for example, a tab. The hierarchy of the secondary key data field values can be set by the user and associated with a display tree configuration widget. In a particular aspect, the configuration of the display tree by one user does not affect the display tree configuration of another user who is accessing the same key field set.

[0013] In one aspect, the step of displaying information includes displaying a plurality of text boxes to permit a user to edit data values. The text boxes can include, for example, drop-down menus when the available set of data values is finite.

[0014] In another aspect, the graphical user interface is customizable. In this regard, the step of configuring the key data fields can include assigning data field values to variable configuration fields to assign placement criteria in the graphical user interface to text boxes representing the key fields.

[0015] In yet another aspect, a file can be attached to the database record such that the file can be accessed by a user. Files can include, but are not limited to, word processing files, spreadsheets, images and other documents such as PDF files. The configuration step can include associating a program with the type of attached file, such as particular word processing program or spreadsheet program. The program association can also include specifying the program version that is to be used to open a file.

[0016] The data field values can be stored in a single database file, which can be backed-up. The single database file can also include any attachments, such that the entire application is contained in a single file.

[0017] In another embodiment, a computer-readable medium is provided that includes stored instructions adapted for providing information in a graphical user interface by receiving a plurality of data field configuration variables that define the properties of a plurality of data fields and generating a graphical user interface comprising the data fields,

where the graphical user interface includes a skewable display tree that can be configured by a user interfacing with the graphical user interface.

[0018] In another embodiment, a computer-implemented system is provided for the management and display of information on a graphical user interface. The system can include a configuration component for interacting with a user wherein the user inputs data field configuration variables that are stored in a database, and an application component, wherein the application component reads the data field configuration variables and displays a graphical user interface that is defined by the data field configuration variables.

[0019] In one aspect, the computer-implemented system also includes a data loader component for populating data fields with data.

[0020] These and other embodiments and aspects of the present invention will be apparent from the following description.

DESCRIPTION OF THE DRAWINGS

[0021] FIG. **1** illustrates types of configuration variable values that can be used in a framework configuration variable set according to an embodiment of the present invention.

[0022] FIG. **2** is a flowsheet illustrating the generation of display of information through a hierarchy mask according to an embodiment of the present invention.

[0023] FIG. **3** is a flowsheet illustrating the configuration of a module according to an embodiment of the present invention.

[0024] FIG. **4** is a screen shot illustrating the configuration of a variable key field according to an embodiment of the present invention.

[0025] FIG. **5** is a screen shot illustrating the configuration of a variable primary key field according to an embodiment of the present invention.

[0026] FIG. **6** is a flowsheet illustrating the generation of a variable module field set according to an embodiment of the present invention.

[0027] FIG. **7** is a screen shot illustrating the configuration of associations between a file type and a program according to an embodiment of the present invention.

[0028] FIG. **8** is a flowsheet illustrating the generation of a display of information according to an embodiment of the present invention.

[0029] FIG. **9** illustrates a screen shot of a login interface for accessing a database according to an embodiment of the present invention.

[0030] FIG. **10** is a screen shot illustrating the selection of a module from within a database according to an embodiment of the present invention.

[0031] FIG. **11** is a screen shot illustrating a graphical user interface for the display of information including dynamically skewable display tree and a display of information according to an embodiment of the present invention.

[0032] FIG. **12** is a flowsheet illustrating the individual configuration of a dynamically skewable display tree according to an embodiment of the present invention.

[0033] FIG. **13** is a screen shot illustrating the configuration of a display tree tab according to an embodiment of the present invention.

[0034] FIG. **14** is a screen shot illustrating the configuration of a display tree associated with a display tree tab according to an embodiment of the present invention.

[0035] FIG. **15** is a screen shot illustrating the configuration of a display tree associated with a display tree tab according to an embodiment of the present invention.

[0036] FIG. **16** is a screen shot illustrating a revised configuration of a display tree associated with an explorer tab according to an embodiment of the present invention.

[0037] FIG. **17** is a screen shot illustrating the configuration of display tree tab placement according to an embodiment of the present invention.

[0038] FIG. **18** is a screen shot illustrating a revised configuration of display tree tab placement according to an embodiment of the present invention.

[0039] FIG. **19** is a screen shot illustrating a revised dynamically skewable display tree according to an embodiment of the present invention.

[0040] FIG. **20** is a screen shot illustrating a revised dynamically skewable display tree that can be selected through a display tree tab according to an embodiment of the present invention.

[0041] FIG. **21** is a flowsheet illustrating the application of a security hierarchy mask to a variable field set according to an embodiment of the present invention.

[0042] FIG. **22** is a screen shot illustrating an interface for configuring security access levels for users of a system installation according to an embodiment of the present invention.

[0043] FIG. **23** is a flowsheet illustrating the implementation of a search of data field values according to an embodiment of the present invention.

[0044] FIG. **24** is a flowsheet illustrating the generation of a report on data field values according to an embodiment of the present invention.

[0045] FIG. **25** is a flowsheet illustrating the population of variable fields with data field values and attachments according to an embodiment of the present invention.

DESCRIPTION OF THE INVENTION

[0046] The present invention is directed to a computer-implemented and variable-driven information retrieval and management method and system that is adapted to create and implement a client-specific application for managing information. In this regard, data fields of interest can be dynamically configured by the implementation of data field configuration variables that are assigned a unique set of configuration values to define, for example, what the application can do, how the application is interfaced by a user, what users can interface with the application, what those users can do within the application, and what data values are stored in the application.

[0047] The configured application can include a set of variable data fields that includes one or more module data field sets, each of which defines a unique module within a system installation. Each module data field set includes and is defined around a primary key data field, where the primary key data field is the primary constraint for the other data field values contained in that module. Each module data field set can also include a plurality of secondary key data fields. The secondary key data fields, along with the primary key data field, comprise the variable data fields that can be utilized in a dynamically skewable display tree to display information to a user and to facilitate the rapid location of information by a user.

[0048] The variable-driven and computer-implemented information management system, sometimes referred to herein as a system installation, can include: i) an application component for accessing and manipulating electronic information through one or more modules, such as data, objects and attachments that are contained in the module(s); ii) configuration component for configuring the data fields using configuration variables, including at least one complete module data field set to define a unique module within the system installation; and optionally iii) a loader component for populating the data fields in the module(s) with, for example, data field values, objects and attachments.

[0049] Different pieces of electronic information (e.g., data field values or files) can be stored in the same system installation. Further, each system installation can include one or more module data field sets, and each module data field set within a system installation can be created by a unique set of data field configuration variables and can include its own unique set of data field values, attachments and/or electronic objects. Each module, defined by a module data field set, can also be searched and reports can be generated using the data field values and the electronic files and objects that are associated with that module. A module within a system installation can also be capable of connecting to other data sources and objects, such as through ADO (ActiveX Data Objects) connectivity, to extract data, files, objects and other information from those data sources.

[0050] Once a data field set is configured, such as by using a configuration component, and is populated with data field values, such as by using a loader component, the entire system installation can advantageously be saved into a single backup file, which when restored can restore all of the necessary components for the system installation, including the data field configuration variables and the most recent data field values. In this manner, the system installation can be easily backed-up, and can be easily copied and moved from one location to another, for example from one server to another server. Further, the system installation can be client-server based and/or can be accessed through a web-browser environment (e.g., HTML-based) using the same configuration variable set.

[0051] The method and system of the present invention utilize variables, referred to as configuration variables, to define and create an application for managing information. By using variable configuration values, new applications can be rapidly created and existing applications can be readily changed to meet the demands of the end-user client. Thus, a system installation can include a set of data field configuration variables that are stored by the system in configuration variable fields. The configuration variables can define the overall configuration of a given system installation, and can be unique for every system installation. The configuration variables can be categorized under two configuration variable sets: a framework configuration variable set and a module configuration variable set. The framework configuration variable set can include the core variables that define what the overall system installation is and what it does, and the framework configuration variable set can be fully customized for each system installation. Every module in a given system installation can be constrained by the same framework configuration variable set.

[0052] In comparison, each module configuration variable set within the system installation defines what each module is and what each module can do within the parameters of the framework configuration variable set. A system installation will typically have one framework configuration variable set and multiple module configuration variable sets, each module configuration variable set defining a unique module within the system installation. These configuration variable sets, taken together, can define all aspects of how the system installation functions.

[0053] The configuration variable sets can be created using a configuration component. The configuration component of the system installation will typically only be accessed by a system administrator, programmer or similar person, as the configuration component is used to define the parameters of the application, to set permissions for users, and the like.

[0054] As is noted above, each system installation will include at least one module, and typically will include multiple modules. Each module can be constrained by the framework configuration variable set, which is the standard configuration that is created for a given system installation. Thus, the framework configuration variable set is populated with configuration values for potential use by all modules within the given system installation. Some of the different constraints and parameters that can be implemented through a framework configuration variable set are illustrated in FIG. **1**.

[0055] The framework configuration variable set comprises a set of values that define how the application component interacts with the module data fields within the system installation, and the framework configuration variable set can be unique for a given system installation. For example, the framework configuration variable set can include configuration variables **102** that determine which instances (e.g., independent database engines) of the system installation are available to the users of the system installation. That is, different instances of a system installation can reside on different database installations, or can have different system configurations within the same database installation. This configuration variable **102** can thus be changed to prevent access to certain information for all or some users of the system installation.

[0056] The framework configuration variable set can also include variables **106** that determine which of the modules within the system installation are available to users. A given system installation will have a minimum of one module, and there is no practical upper limit on the number of installed modules for a given system installation.

[0057] The framework configuration variable set can also include variables **104** that set the number of users that are permitted to access a module within a system installation. This can be useful, for example, for complying with licensing provisions of a software installation that restrict the number of users that can use the software. If the permitted number of users changes, a system administrator can implement the change by simply changing this variable.

[0058] The framework configuration variable set can also include variables that set certain default values for modules in a system installation. Some of these default settings can subsequently be changed during module configuration or by a user of the application for that user from within a module. For example, default tab configuration variables **108** can be set as a default for each user. The default menu items **112** that will appear on the graphical user interface can also be set. Framework configuration variables can also be used, for example, to set other aspects of the default GUI appearance and available navigation options for the users. Preferably, a user can later dynamically create and select new tabs for skewable display tree navigation, such as by altering the display tree hierarchy, as is discussed below.

[0059] The framework configuration variable set can also be used to configure security groups that are available in the system installation for the modules contained in that installation. Different security groups can be set with different levels of security within a system installation (e.g., ability to access, view, edit or delete information) and with respect to each module contained within the system installation. Users can then be assigned to a security group, such as by a system administrator. For example, if a user is not a member of a security group that has permission to access a given module, then that user will not see the name of that module when they log in to the system installation. The framework configuration variable set can also include variables that determine the security levels that a user can operate within each module, if they have permission to access that module.

[0060] The framework configuration variable set can also include default file association variables **110** that set the default file-program associations within a module for accessing and/or manipulating different types of electronic files. The file association determines what program and/or what program version will be used to open a selected file when that file is checked out of the database. In one embodiment, individual users can override the default association set by the framework configuration variables and set the program or program version that will be used to open a particular file type. A file association configuration can include program version settings for each user when users have different versions of the same program. If a user has more than one version of the same program (e.g., Microsoft Word 2000 and Microsoft Word 2003), the file can be opened with the version of the program that is appropriate for the selected file, such as to maintain the compatibility of saved documents across users with different program versions.

[0061] The framework configuration variable set can also include the incorporation of a help system **114** for the system installation. The framework configuration variable set can also include variables that set the default location of certain data fields on the GUI when a module is accessed, such as the location of the primary key data field.

[0062] The foregoing represents only an illustrative description of parameters and settings that can be set through the use of framework configuration variables.

[0063] FIG. **2** is a flowsheet that illustrates steps for configuring a new module or editing an existing module within a system installation using a configuration component. Modules are contained within a system installation and each module is defined by the configuration variables that are used to create the module, including the module data fields. The creation of different modules within a system installation enables users to rapidly locate and manage different types of information that are relevant to their tasks. The module data fields can include variable key data fields, all of which are required to be populated with data, and can also include variable regular data fields. The key data fields can include a primary key data field and one or more secondary key data fields. The key fields are the data fields that will be available for display in a hierarchal display tree for that module. Each module includes a single primary key data field which is assigned a static data field value, where the primary key data field value is different for each database record in the module. The secondary key fields can include static values or dynamic (i.e., calculated) values.

[0064] To configure or create a module in a system installation, a user such as a system administrator opens the configuration component **202**. The user decides **203** to either edit an existing module or to create a new module. If the user chooses to edit an existing module, then the user opens the existing module **208**, which is identified by a unique module name within the system installation and is created around a unique primary key data field. As is discussed above, each module has one primary key data field that is unique to that module within the system installation, and the primary key field is preferably assigned a static data field value (i.e., the value is not dynamically calculated).

[0065] When an existing module is opened using the configuration component, the pre-existing configuration variable set is retrieved **210**, such as by retrieving the data from a SQL server. Thereafter, the user can edit the configuration variables **215** for the module, such as by manually editing the configuration variables, to create a revised module configuration variable set. The revised variable set can then be saved in the system installation database **216**. Data field properties that can be configured through the configuration variable set can include, but are not limited to, the name of a data field, the size of a data field, the placement of a data field name and text box on the graphical user interface, the type of data and the source of the data that will populate a data field, and whether or not a data field is a primary key data field, a secondary key data field or a regular data field. The setting of data field configuration variables are discussed in more detail below with respect to FIG. **4**.

[0066] Alternatively, a user can decide to create and configure a new module by selecting a unique name **212** to identify the module. The user then creates and configures a new configuration variable set **214** by assigning values to the configuration variables. The new module configuration variable set can be saved in the system installation database **216** where it can be readily identified by its unique name. At a minimum, the new module configuration variable set includes the primary key data field around which the module is configured.

[0067] FIG. **3** is a flowsheet that illustrates how a GUI display of information can be generated through the implementation of a module configuration variable set **301**. As noted above, the module configuration variable set **301** is unique for each module within a system installation. The variable set **301** comprises variables that define the key data fields **302**, including a primary key data field **304** and one or more secondary key data fields **306**. For every module, the primary key data field **304** is the field through which all other data fields for that module are related, and it is the primary constraint on the information that can be accessed, manipulated and displayed using that module. The primary key data field **304** can be populated with a unique static value for every database record in the module, the value being a primary piece of information that is useful for searching and reporting functions within a module field set. By way of example, if the module relates to the information and data maintained by a professional services firm, such as an accounting firm or law firm, the primary key field could be the client number, and each database record could then be identified by a unique client number value (e.g. 001, 002, 003, etc.) in the primary key field. Thus, all of the other data field information in that database record would relate to that client number. Each system installation can include multiple modules. By way of illustration, and continuing the previous example, a different module could be created for the professional services firm that related to the individuals who are employed with the firm. In this case, the primary key field for that module could be, for example, the last name of the employees. Secondary key data fields and regular data fields that could be associated with this primary key field could include, for example, a home address, a birth date, a telephone number or the like.

[0068] Thus, the primary key data field **304** is the primary constraint for data entry into the other data fields within the module. The primary key data field **304** should therefore be carefully selected and well-defined when configuring a module using a configuration variable set. Each unique module is associated with a unique primary key data field, and hence a unique data field set within the system installation.

[0069] As is noted above, the module data field set also includes secondary key data fields **306**. The secondary key data fields **306** and the primary key data field **304** together comprise the key data fields, and therefore the master hierarchy mask **316**. The master hierarchy mask **316** can include all of the key data fields that are available for display in a hierarchal display tree for that module. The data field values for the secondary key data fields can optionally be changed by a user and the revision to the database record can be saved. As with the primary key data field **304**, the secondary key data fields **306** can be used for all searching and reporting functions within a module and are required to have a value.

[0070] The module data field set can also include regular data fields **312**. Regular data fields **312** make up the rest of the data fields in a module data field set, and are not part of the master hierarchy mask **316**. As a result, the regular data fields **312** do not appear in a hierarchal display tree and therefore cannot be dynamically skewed by a user, as is described below. However, regular data field values **312** can be used for searching and reporting functions within the module.

[0071] The properties of each data field within a module data field set are defined during the configuration of the module through the configuration variables. The data field properties can include what the type of data is and what the source of the data is for that field. For example, the data can be static or dynamic, and if it is dynamic the data field configuration can include instructions as to how the data is to be calculated and what and where the source data values are for the calculation. For example, a data field could be defined as "Member Age" where the data is the age of a person. The value for this field could be calculated using an algorithm by extracting the person's birthdate from a data table and the current date from a computer system. Thus, the age of the person would be dynamically calculated and automatically inserted into that field.

[0072] The last known data field values are stored in the system installation database and are refreshed when a module is opened and a field value is accessed. If a change is made to the value, such as by user input at the GUI, the new value is revisioned and stored in the system installation database. Optionally, the system installation database can save the prior revisions of a database record such that the prior revisions of the information are available for display. The prior revisions can also be locked to prevent further editing of the prior revisions.

[0073] Further, data field values can be acquired and can populate the module data fields from different systems that are resident outside of the system installation database. Thus, disparate systems including different types of data can be tied together to yield data field values that combine systems that would otherwise not be combinable. Data field values for a module can also be extracted from other module field sets within the system installation or a separate system installation.

[0074] Through the configuration component, the module field set can be created by creating a plurality of data fields, including the primary key field, secondary key fields and regular data fields. Each data field can be defined with respect to the type of field, the data type which will populate the field, the size of the field and the display properties for the field, such as the position of the field name and field text box on the GUI when the module is accessed.

[0075] Different modules that share a common master hierarchy mask can advantageously share data field values from one module to another. That is, master hierarchy mask field values in one module can advantageously be copied to a different module to ease the creation of multiple modules that use overlapping data within a given system installation.

[0076] Referring back to FIG. **3**, the module can also include a security hierarchy mask **320**, which is defined when the module is configured. The key fields in the security hierarchy mask **320** are also members of the master hierarchy mask **316**. The security hierarchy mask **320** is composed of the key fields in the master hierarchy mask **316** that are available for display **318** in a display tree, but whose field values cannot be changed by one or more end-users, depending on the security settings applied to that module with respect to that user.

[0077] The module can also include primary key field entry revision information. That is, as changes are made to a secondary key data field or a regular data field in a module, the user can have the option to save each change under the primary key field, making a revision of the information contained in the module and saving the revision. As each module is based upon a unique primary key field, it is the primary key field around which all revisions to the secondary key fields and regular data fields are made.

[0078] When accessed by a user, the module data fields are populated with data. Through the configuration of the module, data source information **314** and **307** is applied to populate the secondary key fields and the regular data fields. The primary key data field information **308** includes the location of the static values for the primary key field. Thus, each data field in a module data field set can be defined during configuration, such as to its data type and data source. Examples of the data definition include whether the data is static or dynamic (e.g., calculated), and if dynamic, how it's calculated and what the source data values are for the calculation. The module can also be configured to periodically execute live updates, whereby data in the database is compared to the source of the data to determine if any changes have been made.

[0079] The module can also include configuration variables that identify attachments **310** (e.g., electronic files and objects) and a module can have the ability to attach files and objects to its primary key data field. The attachments are stored in the system installation database, and when a user desires to access a file, such as a word-processing document or other file to edit or otherwise manipulate the file, the file is "checked-out" of the system installation database and the user can modify the file outside of the system installation. When the user has completed editing or otherwise manipulating the file, the file is checked back into the system installation database and stored as a binary large object (BLOB). This enables a user to check-in and check-out the electronic attachments **310**. The GUI can also identify if another user of the system is currently accessing the attachment **310** and the system can lock-down that attachment until the other user has checked the attachment back in to the database. This can prevent one user from over-writing the work of another user.

[0080] The module field set can also include revision information for an attachment **310**. As file attachments are modified, the system can recognize that a revision to the file has been made and the revision information can be stored along with a separate copy of each revised file as a separate BLOB. Thus, when a document or other file is attached to the primary key field, one or more new variables can be stored to track the document and the revision information.

[0081] FIG. **4** illustrates a screenshot **400** from a configuration component where a user such as a system administrator sets the configuration variables for a module in a system installation. The configuration variables can be used to configure, for example, the placement of a field on the display screen and the field data value definitions for the data fields.

[0082] As illustrated in FIG. **4**, the module being configured is identified at the top of the Properties dialog box **401** by the FormID "LM", the name of the module. The name of the field that is being edited through the configuration component in FIG. **4** is "MemberID", and the Properties dialog box **401** illustrated in FIG. **4** permits the configuration of the MemberID data field. This properties dialog box **401** can be opened by clicking in the text box **408** in the Preview dialog box **403**, and clicking on a different text box (e.g., First Name) will bring up a Properties dialog box for that data field. The Preview dialog box **403** also previews the appearance of that page of the module field set as changes are made to the appearance variables in the Properties dialog box **401**.

[0083] The configuration variables in the left-hand column **404** of the dialog box **401** generally relate to the appearance of the data field name and data entry widget on the GUI. For example, numerical values can be entered that set the size of the data field (i.e., the maximum number of characters), the width and height of a text box or similar widget, the position of the text box and the position of the text box label.

[0084] In the right-hand column **406** of the properties dialog box **401**, configuration variables that set other constraints and parameters for the selected field can be entered. These can include a configuration variable that defines whether the field is the primary key field. In this instance, MemberID is identified as the primary key field for the module by setting the "Required" variable **402** to True. The "Group By" variable **412** indicates whether the field is a secondary key field. In this case, since the data field MemberID is set as the primary key field, it is not a secondary key field. If both variable values **402** and **412** are set to False, then the data field would be a regular data field. The data type **410** is set to "char" (i.e., character data), and other types of character data can include, for example, date/time data, monetary data, and the like. In addition, the type of widget that is used to accept the value for the data field can also be changed. In the example illustrated in FIG. **4**, the type of widget is a text box. Other types of widgets could include, for example, a drop-down menu, a check box, a calendar pop-up window for selecting a date, or the like.

[0085] Other information entered in the Properties dialog box **401** can include the source of the data that will populate the selected field, such as the Source Server, Source Database and/or the Source Table. A Stored Procedure for calculating a dynamic field value can also be identified.

[0086] The tool dialog box **420** illustrated in FIG. **4**, is provided to initiate the creation or editing of a field. Further, data fields can be displayed on different display pages of the module interface and the display page being edited can also be selected from the drop down-list **424**.

[0087] After each desired data field (e.g., Salutation, First Name, Middle Name, etc. . . .) is configured, the configuration variable set can be saved. Thereafter, when a user accesses the LM module, the input configuration variables will be applied to create and display the module and populate the data fields with data field values.

[0088] FIG. **5** illustrates a screenshot of a dialog box **500** for defining a module, in this instance, for a module with the name "LM" (e.g., the module illustrated in FIG. **4**). The dialog box **500** can be accessed by selecting from a drop down list that appears under the Edit menu entry in FIG. **4**. The name of the primary key data field is MemberID **510**, and other primary key data fields can be accessed through the drop-down menu.

[0089] As is illustrated in FIG. **5**, different data fields can be added to, or deleted from, a Security Hierarchy Mask **508** based on the key data fields available in the Master Hierarchy Mask **506**. The data fields placed in the Security Hierarchy Mask (LastName, FirstName, State and City) will not be able to be changed by a user of the LM module without access to the configuration component. The key fields that are in the Master Hierarchy Mask **506**, but are not in the Security Hierarchy Mask **508** (e.g., MemberStatus, MemberType) can be revised by a user accessing the LM module.

[0090] The MediaTable **502** indicates the media table that will be accessed to retrieve the field data values for that module, and each module can have its own media table. The DocTable **504** indicates the data table where files, typically in the form of BLOBs, are stored for that module. The Media Key **510** represents the name given to the primary key field for that module and the FormName **512** (e.g., Member Explorer) is the name given to the module that will appear in a drop-down list of available modules to the user, such as from the top menu bar.

[0091] Thus, by using a configuration component to define a module (FIG. **5**) and to input configuration variables to define the data fields within the module (FIG. **4**), one or more unique modules can be created within the system installation.

[0092] Thus, the data field properties for each module are stored by the system as configuration variables. Because the data field properties are stored as variables, a user can manipulate the data fields within a module in a manner that is most convenient for that user. For example, according to one aspect of the present invention, a user can advantageously configure how information (e.g., data field values) is displayed within a module through the use of dynamic tab configurations and a dynamically skewable hierarchal display tree, and can also configure how attachments will be opened. In this regard, FIG. **6** is a flowsheet illustrating the generation of a module display for an individual user.

[0093] For example, stored file association configurations **604** determine how an attachment is opened from within the module, such as with what program and/or program version. Typically, the configuration component will establish a set of default association values, as is noted above. After accessing the module, a given user can be given the option to select a different program and/or program version to open a selected file type. Upon opening, the file will be checked-out of the system, and display page **606** will display the open attachment **608** by utilizing the selected program.

[0094] As an example, FIG. **7** is a screen shot **700** illustrating the setting of file associations by a user working in a module labeled Member. Using the Association Manager dialog box **702**, the user can configure the association manager to open files with an extension **708** of "SPD" with Version 13 **706** of the program SoftPlan.exe **704**. The user can also provide the external path to that program.

[0095] Referring back to FIG. **6**, a skewable display tree can also be configured and viewed by a user. As used herein, a skewable display tree is a hierarchal display tree, similar to the common folders view of folders and sub-folders in a Windows operating system, but where the data values and the hierarchy of the of data values that are displayed in the tree can be reconfigured by a user, and preferably only for use by that user. The reconfiguration of the display tree can include adding or removing key data field values from the tree hierarchy, or changing the hierarchy of the key data field values. The re-skewing of the tree hierarchy by a user preferably does not change the key data field values and/or the hierarchy of the key data field values that are displayed to another user. The viewable key data field values can include both static data **612** and dynamic data **614**. The viewable key data fields can optionally be displayed in a web-based application that runs through a web browser.

[0096] The GUI can also include one or more GUI widgets, such as radio buttons or tabs, that can be configured by a user to dynamically display a pre-defined skewable display tree. Preferably, the GUI widget is a configurable tab that can be selected by the user to dynamically generate the skewable display tree. Dynamic tab configurations **610** enable a user to quickly view information in a manner that is most useful to that individual user. By adjusting the dynamic display tree tab configurations **610**, a skewed (reconfigured) display tree **618** can be instantly generated and viewed by the user.

[0097] FIG. **8** is a flowsheet illustrating the access to a module by a user and the display of information by the module. A user inputs log-in security credentials **802**, which are passed on to the system installation. Based upon the security credentials input by the user, selected modules are loaded into the application for access by the user. Each module is loaded **804** based upon the security credentials that are input by the user. As is discussed above, each module includes a unique set of data fields **808**, including secondary key data fields **810** and a primary key data field **814**. As is discussed above, the data fields are populated based upon the configuration of the data fields through the configuration variable set. The primary key data field **814** and secondary key data fields **810** make up the master hierarchy mask **826**, including the security hierarchy mask **828**.

[0098] FIG. **9** illustrates a screenshot **900** of a log-in interface for a system installation. A user logging into a system installation can select a server **902** (e.g., BIZLT0001\BIZEXP) and/or a database **904** (e.g., BE333) to access. Access to the servers and databases can be controlled through the security credentials for the user, who inputs a username and password to access the selected database.

[0099] FIG. **10** illustrates a screenshot **1000** of the main GUI that can appear when a module within a system installation is accessed by a user. The selections available by accessing the menu bar **1001** at the top of the screen generally represent options set by the framework configuration variable set for that system installation. For example, as illustrated in FIG. **10**, the user has opened the list of available modules for that system installation, as determined by the user's security credentials, and has selected a module labeled Member **1002**. Once the module is selected and opened, a skewable display tree **1003** on the left-hand side of the GUI will be populated with the data field values from the key fields for that module, and in the default display configuration of the module, or the revised display configuration that was last utilized by the user for that module. As illustrated in FIG. **10**, the skewable display tree **1003** is displayed with a top hierarchy field of the last name. Dynamic configuration tabs **1004** are located below the skewable display tree **1003**. Selecting one of the dynamic display tree tabs will immediately change the hierarchy of the skewable display tree **1003**. It should also be noted that the GUI also includes paging tabs **1005** at the bottom of the data display to change the page that is being viewed.

[0100] FIG. **11** illustrates a screenshot **1100** of skewable display tree for the Member module. As illustrated in FIG. **11**, the display tree is set with a hierarchy of Last Name\First Name\MemberID. That is, the display tree is first arranged alphabetically by last name, and under each last name entry the tree is arranged by a first name. Under the first name, the identification number (MemberID) for that member is displayed. In this instance, MemberID is the primary key field for the Member module. Under the Member ID value, the revision number of that database record (R00) is displayed. Revision R00 is unlocked (i.e., is not being accessed and revisioned by another user) as indicated by the unlocked icon **1104** in the display tree. If information in the database record is revisioned in any way by the user, that revisioned record can be saved as a new record (e.g., R01).

[0101] The GUI also includes dynamic display tree configuration tabs **1102***a*, **1102***b* and **1102***c* that permit the user to quickly and easily view the display tree in a different hierarchy or using different key data fields by simply selecting a different display tree tab. It will be appreciated that other GUI widgets could be used, such as radio buttons.

[0102] As illustrated in FIG. **11**, the user has selected MemberID **349**, Revision R00, and the right-side information pane of the screen shows the information that is associated with that MemberID through implementation of the configuration variables for the module and the population of the data fields for that module. A user having sufficient security credentials can update the database record (e.g., as identified by MemberID **349**) by editing the information displayed in the data fields.

[0103] As is noted above, one aspect of the present invention is directed to a dynamically skewable display tree that is individually configurable by a user. FIG. **12** is a flowsheet that illustrates the configuration of a dynamic display tree according to an embodiment of the present invention. A display **1202***a* displays the existing configuration when initially accessed by a user (e.g, as in FIG. **11**). The existing display configuration is generated based upon the current field data values **1204***a*, the display tree **1206***a*, the primary key field **1208***a* and the file association variables **1210***a*. The display of each of the data **1204***a*, the display tree **1206***a* and the primary key field **1208***a* can be manipulated by changing the display tree hierarchy **1216** associated with each dynamic display tree tab. This can be accomplished by changing the display tree hierarchy **1214** for a selected tab using the tab configuration, and thus effecting the display tree hierarchy **1216** which is then loaded **1218** as the new display tree tab configuration. Thus, the data **1204***b*, display tree setup **1206***b* and primary key field **1208***b* will be reconfigured and redisplayed on the module display **1202***b*.

[0104] FIG. **13** illustrates a screen shot of an explorer tab configuration dialog box **1300**. The dialog box **1300** can be accessed by selecting Appearance **1306** from the top level menu. By selecting to change the display tree hierarchy **1304** (e.g., by mouse-clicking in the Hierarchy text box) for the selected tab **1302** (e.g., Last Name), the user can access a Set Hierarchy dialog box **1400**, as is illustrated by FIG. **14**. The dialog box **1400** displays to the user the existing configuration for the display tree tab Last Name and provides a list of key fields from the master hierarchy mask that can be added to the hierarchy configuration. For example, if the user selects MemberStatus **1402** (FIG. **14**), then MemberStatus is added to the display tree hierarchy (FIGS. **15** and **16**).

[0105] Further, the order of the tabs can also be changed by the user, such as by selecting the Re-Order Tabs radio button (FIG. **13**). In this case, a Select Tab Order dialog box (FIGS. **17** and **18**) is displayed and the user can change the order in which the display tree tabs are displayed. As illustrated in FIG. **17**, the display tree tab order is LastName\Location\MemberStatus, as is illustrated in FIG. **11**. In FIG. **18**, the user has changed the order to Location\LastName\MemberStatus, as is illustrated in FIG. **19**. Note that as illustrated in FIG. **19**, the Location display tree tab hierarchy is set to State\City\LastName\FirstName. Should the user then wish to display the information by Last Name at the top of the display tree hierarchy, the user can simply select the Last Name tab **1902** and the display tree will be instantly and dynamically redisplayed accordingly, as is illustrated in FIG. **20**. Since all of the key data field information is stored by the system as variables, these display reconfigurations advantageously do not affect the underlying data and do not affect the way that other users of the same system view the data. The configuration changes made by a user are unique to that user and are stored by the system in relation to that user.

[0106] As is noted above, the present invention can also provide a security hierarchy to control access to electronic information within the system. FIG. **21** is a flowsheet that illustrates the implementation of a security hierarchy configuration. After a determination **2102** by a system administrator as to which module(s) and what level of access a given user will have, the system installation is loaded with those security settings. The security settings can control, for example, the available data **2106**, the display tree **2108**, the primary key field **2110** and the file associations **2112** that a user is able to access. As a result of these security settings, the display **1214** will be configured for that user and some of the data fields will not be able to be modified or deleted by an individual user.

[0107] FIG. **22** illustrates a screenshot **2200** of an interface that can be used by, for example, a system administrator setting the security access for individual users. Options include security groups that can be selected for the user. The access that is configured for a user can be different for each module in the system installation (referred to in FIG. **22** as Form Name **2202**).

[0108] Another feature that can be included in the information management system is a method for searching and reporting on data within a module. A search hierarchy is illustrated in FIG. **23** and a report hierarchy configuration is illustrated in FIG. **24**.

[0109] Referring to the search hierarchy illustrated in FIG. **23**, the user causes a module to send a search request **2302** based upon the module field set. The module name is sent to the search engine **2304**. Thereafter, a user can choose **2306** to either search the module for data values **2306***a* or for character strings related to an attachment **2306***b*. When searching for a data value **2306***a*, the search engine can search upon the available search fields from all of the meta-data fields, including the primary key field, secondary key fields and general data fields. For example, as is illustrated in FIG. **23**, up to five data fields can be searched simultaneously to obtain a result set **2318**.

[0110] When searching attachments **2306***b*, the search engine provides entry fields for searching by attachment file name or extension, or portions thereof. The full-text of an attachment (e.g., a word-processing document) can also be searched. Thus, a search can be conducted by attachment name **2312** or by attachment extension **2314** to produce a result set **2318**. The user can then choose the desired search result **2320** from the result set and the results can be displayed in the module **2322**.

[0111] FIG. **24** illustrates a report hierarchy configuration. A module sends a report request **2402** to a report engine, which displays the report **2406**. The user can then decide **2408** to create a new report **2408***a* or modify an existing report **2408***b*.

[0112] As is noted above, the system installation can optionally include a data loader to initially populate a module with data. A flowsheet illustrating the use of a data loader is illustrated in FIG. **25**. A user opens a data loader **2502** and then chooses **2504** which module will be populated and with what type of data. Meta-data **2506** and attachments **2506** can be loaded using the data loader. For meta-data **2506**, external data **2510** is mapped to the module field set based upon the provided FormId for attachments **2508**, the attachments are mapped **2512** to the module key field set also based on the FormId and the primary key field value. The external meta-data **2514** and the external data attachments **2516** are then loaded into the module data tables based on the FormId and the method chosen by the user so that the data can be displayed **2520**.

[0113] Each individual module, with its unique primary key field and secondary key field set, is searchable and reportable as the system installation knows which module it is working with by virtue of the field definitions associated with that module.

[0114] In accordance with the foregoing, the method and system of the present invention advantageously can provide one or more of the following advantages:

[0115] the generation of dynamically skewable display trees, optionally with security settings that are determinable by user group;

[0116] configurable dynamic display tree tabs that can be configured by an individual user without affecting the configuration used by another user;

[0117] search capabilities that are dynamically related to the field data values;

[0118] different module key field sets that automatically bring different search parameter options;

[0119] module key field set that determines the search parameters options that are displayed;

[0120] the entire workings of the application can be stored in a SQL server database as variables—it is because the entire application is stored as variables that the application can be dynamically changed;

[0121] the entire framework for the system installation can be contained within the database;

[0122] all of the customization that makes the module key field sets function individually can be stored within the database;

[0123] all security, user data, attachments, reports, and functional intelligence can be contained within the database;

[0124] the ability to pull data values from multiple data sources to populate the fields in the graphical user interface;

[0125] ability to join disparate types of systems and data utilizing common ODBC using Microsoft SQL Server linked servers for live updates;

[0126] non-live updates can be performed through file conversions and loading through a data loader component that does not use Microsoft SQL Server linked servers for the data flow or an ETL (extract, transfer, load) tool;

[0127] enables software version and revision control;

[0128] revisions of attached files can be made and saved;

[0129] different versions of the same software can be managed to open the original file type correctly, as set by a user;

[0130] file protection can be assured by using a secured "check-in/check-out" system that integrates with third-party software applications. This enhances file management and security. When a file is changed by a user, the system can automatically save the revised file with a new filename while also saving the prior version(s) of the document. The prior version(s) can optionally be "locked" to prevent further editing once the newer version has been saved;

[0131] can keep third-party application users from overwriting their work by checking the file out to a user and then blocking all other users from checking out the same file there is no issue of over-writing work;

[0132] can open third-party applications only in the mode intended and that the user has permissions for; and

[0133] can provide group level security on all third-party application files and informational objects.

[0134] While various embodiments of the present invention have been described in detail, it is apparent that modifications and adaptations of those embodiments will occur to those skilled in the art. However, is to be expressly understood that such modifications and adaptations are within the spirit and scope of the present invention.

What is claimed is:

1. A method for managing and displaying information, comprising the steps of:

configuring a plurality of key data fields to create a key data field set, the key data field set comprising a primary key data field and a plurality of secondary key data fields, and the configuring comprising assigning values to a configuration variable set that defines properties of the key data fields;

assigning a plurality of unique static data field values to the primary key data field, where each static primary data field value identifies a unique database record;

populating the secondary key data fields for each unique database record with secondary key data field values; and

displaying information on a graphical user interface, including secondary key data field values, the graphical user interface comprising a skewable display tree whereby at least a portion of the secondary key data field values are displayed in the skewable display tree.

2. A method as recited in claim **1**, wherein the populating step comprises assigning a dynamic data field value to at least one secondary key data field.

3. A method as recited in claim **2**, wherein the configuring step comprises linking a stored procedure for computation of the dynamic data field value to the at least one secondary key data field.

4. A method as recited in claim **1**, wherein the populating step comprises assigning a static data field value to at least one secondary key data field.

5. A method as recited in claim **1**, wherein the step of displaying information comprises applying a hierarchy mask to the key data field set to restrict the key data fields that are displayed to a user on the graphical user interface.

6. A method as recited in claim **1**, wherein the step of populating the secondary key data fields comprises accepting manual input of data field values from a user through the graphical user interface.

7. A method as recited in claim **1**, wherein the step of populating the secondary key data fields comprises importing data field values from a file.

8. A method as recited in claim **1**, wherein the step of displaying information on a graphical user interface comprises displaying a plurality of display tree configuration widgets, whereby a user can dynamically select the display tree hierarchy of secondary key data field values by selecting one of the display tree configuration widgets.

9. A method as recited in claim **8**, wherein the hierarchy of the secondary key field data values can be configured by the user and associated with a display tree configuration widget.

10. A method as recited in claim **1**, wherein the step of displaying information comprises displaying a plurality of text boxes to permit a user to manually edit data field values.

11. A method as recited in claim **1**, wherein the step of configuring a plurality of key data fields comprises assigning values to data field configuration variables to set the position of data field text boxes in the graphical user interface.

12. A method as recited in claim **1**, wherein the step of configuring a plurality of key data fields comprises assigning values to data field configuration variables to set the type of data that will populate the data fields.

13. A method as recited in claim **1**, wherein the step of configuring a plurality of key data fields comprises assigning values to data field configuration variables to set the type of manual data input interface for data fields in the graphical user interface.

14. A method as recited in claim **1**, further comprising the step of attaching a file to the database record.

15. A method as recited in claim **14**, wherein the configuring step comprises associating a program with the type of attached file.

16. A method as recited in claim **15**, wherein the configuring step further comprises associating a version of the program with the type of attached file.

17. A method as recited in claim **1**, wherein the step of displaying information comprises applying a security hierarchy mask to the key data field set to restrict the key data field values that can be edited by a user on the graphical user interface.

18. A method as recited in claim **1**, wherein the key data field values are stored in a single database file.

19. A method as recited in claim **18**, wherein data field configuration values are stored in the single database file.

20. A computer-readable medium including stored instructions adapted for providing information in a graphical user interface by:

receiving a plurality of data field configuration variables that define the properties of a plurality of data fields;

generating a graphical user interface comprising the data fields, where the graphical user interface includes a skewable display tree that can be configured by a user interfacing with the graphical user interface.

21. A computer-implemented system for the management and display of information on a graphical user interface, the system comprising:

a configuration component for interacting with a user wherein the user inputs data field configuration variables that are stored in a database;

an application component, wherein the application component reads the data field configuration variables and displays a graphical user interface that is defined by the data field configuration variables.

22. A computer-implemented system as recited in claim **21**, wherein the system further comprises a data loader component for populating data fields with data.

* * * * *