PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 9/44

A1

(11) International Publication Number: **WO 99/27444**

(43) International Publication Date: 3 June 1999 (03.06.99)

(21) International Application Number: PCT/US98/22592

(22) International Filing Date: 23 October 1998 (23.10.98)

(30) Priority Data:
08/978,343 25 November 1997 (25.11.97) US

(71) Applicant *(for all designated States except US)*: ENTELOS, INC. [US/US]; Suite 200, 2nd floor, 4040 Campbell Avenue, Menlo Park, CA 94025 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: PATERSON, Thomas, S. [US/US]; 810 Sovereign Way, Redwood City, CA 94065–1762 (US). BANGS, Alex, L. [US/US]; 32 Ventura, P.O. Box 520, La Honda, CA 94020 (US).

(74) Agents: HALVORSON, David, R. et al.; Blakely, Sokoloff, Taylor & Zafman LLP, 7th floor, 12400 Wilshire Boulevard, Los Angeles, CA 90025 (US).

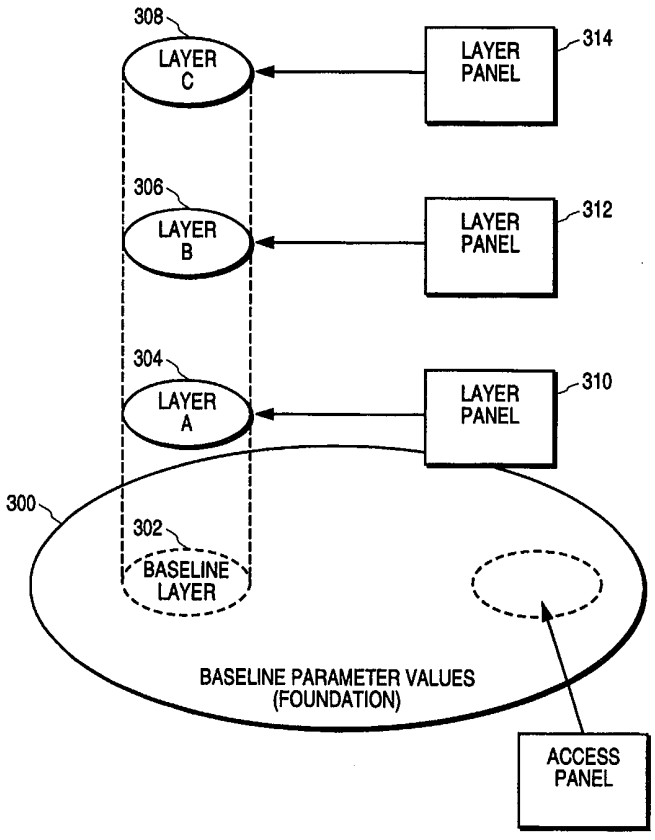
(81) Designated States: AL, AM, AT, AT (Utility model), AU (Petty patent), AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, CZ (Utility model), DE, DE (Utility model), DK, DK (Utility model), EE, EE (Utility model), ES, FI, FI (Utility model), GB, GD, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SK (Utility model), SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: A METHOD OF MANAGING OBJECTS AND PARAMETER VALUES ASSOCIATED WITH THE OBJECTS WITHIN A SIMULATION MODEL

(57) Abstract

A method of managing objects within the simulation model commences with the storage of a first value for a particular parameter within an object of the simulation model. A panel is then displayed within the user interface. The relevant parameter of the above-identified object may then be identified for inclusion within this panel. In response to the identification of the parameter, an identifier (such as an icon) associated with the parameter is displayed within the panel. A second value for the relevant parameter of the object may then be inputted by a user via the panel. The second value for the parameter is then stored separately from the first value for the parameter, and these first and second values can be selectively applied to the parameter.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

A METHOD OF MANAGING OBJECTS AND PARAMETER VALUES ASSOCIATED WITH THE OBJECTS WITHIN A SIMULATION MODEL

RELATED APPLICATIONS

This application is related to the following co-pending applications:

"METHOD OF GENERATING A DISPLAY FOR A DYNAMIC SIMULATION MODEL UTILIZING NODE AND LINK REPRESENTATIONS", in the name of Paterson et al., serial number 08/962,524, filed October 31, 1997.

"A METHOD OF PROVIDING ACCESS TO OBJECT PARAMETERS WITHIN A SIMULATION MODEL", in the name of Paterson et al., serial number 08/979,270, filed November 25, 1997.

"A METHOD OF MONITORING VALUES WITHIN A SIMULATION MODEL", in the name of Paterson et al., serial number 08/977,848, filed November 25, 1997.

FIELD OF THE INVENTION

The present invention pertains generally to the field of simulation modeling. More specifically, the present invention relates to methods of representing, accessing, inputting and monitoring parameters of objects within a simulation model.

BACKGROUND OF THE INVENTION

Simulation modeling is commonly used to model systems to perform "what-if" analyses to optimize system performance and to identify problems within systems. Graphical simulation modeling allows a complex system to be modeled in an intuitive and visually comprehensible manner, and has found application in wide range of fields, from business to biological analysis.

The construction of a simulation model typically involves identifying various objects within the system, which are then represented by variables, equations or both embodied in an "object". A simulation model may be constructed using a graphical user interface (GUI) in which the various objects are represented by user-selected icons or other appropriate graphical representations, and in which the inter-relationships between the objects are represented by links.

A simplified representation of a typical prior art graphical user interface (GUI) for a graphical simulation model is shown **Figure 1**.

Specifically, the prior art GUI of **Figure 1** includes a diagram window 10, within which are displayed node representations for various objects of a modeled system. Each of the various objects of the modeled system is shown to be either an entity object 12, an input object 14 or a link object 16. Each of the objects typically includes at least one parameter which has a parameter name, an assigned value and parameter documentation which describes the parameter.

Known simulation modeling tools include the Process Charter from Scitor Corporation of Menlo Park, California; PowerSim from Modell Data AS in Bergen, Norway (http://www.powersim.com); Ithink and Stella from High Performance Systems Incorporated of Hanover, New Hampshire (http://hps-inc.com); and Extend +BPR from Imagine That! Incorporated of San Jose, California (http://www.imaginethatinc.com). **Figure 2** illustrates a simulation model 30 as generated utilizing the Ithink product from High Performance Systems, Inc. The simulation model represents a work-in/work-out system within a business. The simulation model 30 is shown to include an object 32 that represents "work backlog", the object 32 being fed by arriving work orders 34 and depleted by filled work orders 36. The rate at which work orders are fed to the backlog object 32 is determined by an object 39, which functions as a "valve" with respect to a pipe by which work orders are fed to the object 32. Similarly, the rate at which work orders are dispensed from the object 32 is dependent upon an object 38 which functions as a "valve" for the pipe by which work orders are dispensed from the object 32. The object 38 is shown to receive as inputs the number of workers within the system, as represented by object 42, and the weekly productivity of each of these workers, as represented by the input parameter 40. The weekly productivity of the workers is further a function of hours per week per worker, represented by object 44. The production per hour worked, represented by object 46, is further shown to influence the weekly productivity per worker. Productivity per hour worked is in turn influenced by an average burnout factor, which is represented by an object 48. Various other factors are shown to influence the object 48.

While the simulation model 30 shown in **Figure 2** provides a satisfactory representation of the work-in/work-out system, the model 30 suffers from a number of inefficiencies. Specifically, the mathematical structure underlying the model 30 is not readily apparent from a viewing of the icons, and can only be guessed at as a result of the labels which are attached to the various nodes shown in the simulation model 30. Further,

the numerous icons that are used to represent objects, inputs, pipes and links (as well as the labels associated with each of these icons) result in a cumbersome and cluttered depiction of the modeled system. The input and observation of parameter values associated with objects of the models of the **Figures 1** and **2** is further tedious as a window must be generated for each variable of each object, and a specific value then entered or viewed in a field within the window. In large simulation models, which may have hundreds or even thousands of objects included therein, this process of inputting and viewing values for parameters becomes inconvenient.

SUMMARY OF THE INVENTION

According to the invention, there is provided a method of managing objects within a simulation model. A first value for a parameter of an object of the simulation model is stored. A panel is displayed within a user interface, and the parameter of the object of the simulation model is identified for inclusion within the panel. In response to the identification of the parameter, an identifier for the parameter is displayed within the panel. A second value for the parameter of the object is then received via the panel. The second value for the parameter of the object is then stored separately from the first value for the parameter of the object .

Other features of the present invention will be apparent from the accompanying drawings and from the detailed description that follows.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example and not limitation in the figures of the accompanying drawings, in which like references indicate similar elements and in which:

Figure 1 illustrates a graphical user interface (GUI) utilized in the prior art to generate a display for a simulation model.

Figure 2 illustrates a representation of a simulation model generated using a prior art modeling tool.

Figure 3 is a diagrammatic representation of simulation modeling software according to one embodiment of the present invention.

Figure 4 is a diagrammatic representation of the interaction between the software illustrated in **Figure 3** and GUIs operating on various software platforms.

Figure 5 illustrates a parameter window which allows a user to view and input information pertaining to a parameter of a simulation model according to one embodiment of the present invention.

Figure 6 illustrates two exemplary node representations which may be utilized to represent objects of a simulation model according to the present invention.

Figure 7 illustrates exemplary link representations which may be utilized, according to one embodiment of the present invention, to illustrate a relationship condition between state or function nodes of a simulation model.

Figure 8 illustrates a modifier representation which, according to one embodiment of the present invention, may be utilized to represent the influence of a third node on a relationship condition which exists between first and second nodes of a simulation model.

Figure 9 illustrates an exemplary representation of a simulation model constructed utilizing the graphical elements defined according to one embodiment of the present invention.

Figure 10 is a flow chart illustrating a method, according to one embodiment of the present invention, of generating a display of a simulation model.

Figure 11 is a flow chart illustrating a method, according to one embodiment of the present invention, of selecting a link representation to represent the relationship condition between two objects within a simulation model.

Figure 12 is a flow chart illustrating a method, according to one embodiment of the present invention, of displaying a modifier representation which represents the influence of an object on a relationship condition between a pair of objects.

Figure 13 illustrates access panels, according to one embodiment of the invention, by which a user can access parameter working values.

Figure 14 is a flow chart illustrating a method, according to one embodiment of the invention, of creating an access panel and of accessing parameters of a simulation model utilizing such a panel.

Figure 15 is a diagrammatic representation of a baseline foundation, a baseline layer, and three alternative layers which may be substituted for the baseline layer.

Figure 16 illustrates two exemplary layer panels, according to one embodiment of the invention, which may be utilized to specify values to be included within a layer object.

Figure 17 illustrates a configuration panel, according to one embodiment of the present invention, which may be utilized to apply and de-apply layers within a simulation model.

Figure 18 is a flow chart illustrating a method, according to one embodiment of the invention, of managing objects within a simulation model.

Figure 19 illustrates two exemplary monitor panels, according to one embodiment of the present invention, which allow a user to monitor at least two parameters of objects within a simulation model.

Figures 20a and **20b** illustrate GUIs which each display a diagram window, a chart panel, a monitor panel and a configuration panel.

Figures 21a and **21b** illustrate methods, according to respective embodiments of the present invention, of monitoring parameters within a simulation model.

Figure 22 illustrates an exemplary representation or display of a simulation model utilizing the graphical elements and notation illustrated in **Figures 6-8.**

Figure 23 illustrates a GUI including a diagram window and a pair of access panels according to one embodiment of the present invention.

Figure 24 illustrates a GUI, according to one embodiment of the present invention, including a diagram window, a pair of layer panels, and a configuration panel.

Figure 25 is a diagrammatic representation of a computer system within which software, for performing the methodologies discussed above, may reside and be executed.

DETAILED DESCRIPTION

A method of managing objects within a simulation is described. In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be evident, however, to one skilled in the art that the present invention may be practiced without these specific details.

The present specification describes exemplary methods for representing simulation models of systems, and for accessing and managing objects that describe such systems, utilizing graphical simulation modeling

software. Referring to **Figure 3**, there is provided a diagrammatic representation of one exemplary embodiment of simulation modeling software 50 according to the present invention. Specifically, the modeling software 50 comprises a core 52, which may be coded using an object-oriented language such as the C++ or Java programming languages. Accordingly, the core 52 is shown to comprise classes of objects, namely diagram objects 54, access panel objects 56, layer panel objects 58, monitor panel objects 60, chart objects 62, and configuration objects 64. As is well known within the art, each object within the core 52 may comprise a collection of parameters (also commonly referred to as instances, variables or fields) and a collection of methods which utilize the parameters of the relevant object. The functioning and purposes of each of the various classes of objects shown in **Figure 3** will become apparent from the description that follows. An exploded view of the contents of an exemplary diagram object 66 is provided, from which it can be seen that the diagram object 66 includes documentation 68 which provides a description of the diagram object, a collection of parameters 70, and methods 72 which may define an equation or class or equations. The diagram objects 54 each define a feature or object of a modeled system which is displayed within a diagram window presented by a graphical user interface (GUI) which interacts with the core 52. According to one exemplary embodiment of the invention, the diagram objects 54 may include state, function, modifier and link objects which are represented respectively by state nodes, function nodes, modifier icons and link icons within the diagram window. The objects 56-64, in one exemplary embodiment, each define respective windows (or panels) which are overlaid on a diagram window to present selected information regarding the modeled system, and to facilitate user interaction with the model. For the purposes of this specification, the term "panel" shall be given a broad interpretation and taken to refer to any designated or identified area with a display, regardless of shape, size or demarcation. Accordingly, the term "panel" shall be taken to encompass, but not be limited to, a "window" with a GUI.

Figure 4 provides a diagrammatic representation of the core 52, which is shown to be capable of interaction with any one of a number of GUIs. Specifically, the core 52 is shown to interface with a GUI 80 operating on the Macintosh platform developed by Apple Computer, Inc. of Cupertino, California, a GUI 82 operating on either the Windows '98 or Windows NT platforms developed by Microsoft Corporation of Redmond,

Washington, or a platform-independent GUI 84 coded in Hyper-Text Markup Language (HTML) or the Java language developed by Sun Microsystems of Mountain View, California. Each of the GUIs interacts with the core 52 to present a diagram window in which icons representative of the diagram objects 54 are displayed, and in which panels (or windows) representative of panel objects 56, layer objects 58, monitor objects 60, chart objects 62 or configuration objects 64 may be displayed.

Parameters

As discussed with reference to **Figure 3**, each object defined within the software core 52 may have at least one parameter associated therewith which quantifies certain characteristics of the object, and which is used during simulation of the modeled system. It will also be appreciated that not all objects must include a parameter. In one exemplary embodiment of the invention, several types of parameters are defined. Firstly, system parameters may be defined for each subject type. For example, a system parameter may be assigned an initial value for a state object, or a coefficient value for a link object. Other parameter types include object parameters and diagram parameters that facilitate easy manipulation of values in simulation operations. Specifically, diagram parameters are available to all objects, while each object parameter is available only to a single object.

Referring now to **Figure 5**, there is shown a parameter window 90, that may be generated by any one of the GUIs shown in **Figure 4**, and that provides a user with information regarding a parameter and allows the user to input or specify a value to be attributed to the respective parameter. The manner by which a parameter window 90 is invoked and constructed within a GUI will be described below. The exemplary parameter window 90 is shown to include four sections, namely a definition section 92, a units section 94, a settings section 96 and a range section 98. The definition section 92 displays an identifier (or symbol) for the parameter, as well as an appropriate definition of the parameter. In the illustrated parameter window 90, the parameter is represented by the symbol "C", which is defined as the coefficient of an equation within a specific object. The units section 94 displays units used internally and for assessment. The units used for assessment are used for a value that is designated as a "working" value in the settings section 96. If the internal and assessed units are different, a conversion may also be indicated in the units section 94. In the illustrated

example, the assessed value is reciprocated and multiplied by a constant K that converts from hours assessed to a half-life calculation value.

The settings section 96 includes three values for the relevant parameter, namely a working value, a baseline value and a comparison value. It is possible to designate either the baseline or comparison value as the working value. For example, the user may be able to drag-and-drop the baseline or comparison value, using a pointing device such as a mouse, into the block 100. The working value will be updated accordingly. The baseline value is specified by the original builder of a simulation model in which the relevant parameter is utilized, and cannot be modified directly by a user. The settings section 96 further includes description blocks into which the user may enter descriptions for each of the working, baseline and comparison values.

The range section 98 may optionally be used to define upper and lower value limits that may be assigned to the parameter. Again, description fields for each of the upper and lower limits are provided.

Graphical Elements

Figures 6-8 provide examples of graphical elements which, according to one exemplary embodiment of the invention, may be utilized to construct a display of a simulation model according to the invention. The illustrated graphical elements are made available by a GUI to a model builder, who is then able to construct a representation of a modeled system. In the description which follows, the term "node" is used to reference an icon which is representative of a state or function, and the term "link" used to reference an icon that represents a link object. It will be appreciated that the graphical elements discussed below are exemplary and any distinctive graphical elements may be substituted for the graphical elements discussed below without departing from the spirit of the invention.

The graphical elements discussed below with reference to **Figures 6-8** may be used to construct a simulation model, such as that shown in **Figure 9**. Referring firstly to **Figure 6**, there are illustrated examples of two node types, namely a state node 110 and a function node 112. In order to distinguish between state and function nodes 110 and 112, reference will also be made now to **Figure 9**. **Figure 9** illustrates an exemplary simulation model 150 that represents a predator-prey system. In the model system, the predators comprise wolves and the prey comprise rabbits.

Referring back to **Figure 6**, a state node 110 represents an underlying state object defining a condition or state within the modeled system. In the representation of the simulation model 150 in **Figure 9**, it will be noted that state node represent various conditions within the model. Specifically, a "vegetation" state node 152 represents the state of vegetation within the model 150, while state nodes 156, 158, 162 and 164 respectively represent rabbit and wolf population numbers. The various objects underlying the state nodes may comprise one or more parameters and/or one or more equations (or methods) for which are effected or influenced by links (which may also be termed "arrows") which feed into the respective state nodes (i.e. state and function objects are influenced by link objects). For example, the "adult rabbits" state node 156 represents an object including a parameter indicating the number of adult rabbits within the simulation at any specific time.

Referring to **Figure 6**, a function node 112 represents a function object that defines an element within the simulation model that is purely a function of zero or more objects (or parameters) within the simulation model. Referring again to the simulation model 150 shown in **Figure 9**, two function nodes 154 and 160 are shown respectively to represent objects representing rabbit food supply and wolf food supply. Each of the function nodes 156 and 160 includes a parameter that is a function of parameters of nodes that are shown by links to feed into the relevant function node. For example, the function node 154, which has a parameter whose value represents the quantity of rabbit food available, is shown to be a function of the vegetation node 152. Similarly, the function node 160, which includes a parameter whose value indicates the available quantity of wolf food, is a function of the "adult rabbit" node 156, and the "young rabbit" node 158.

Figure 7 illustrates a set of link representations 118-128 which a model builder may select to represent a relationship condition which exists between two objects, represented by nodes, within simulation model. Each of the link representations 118-128 is associated with and represents a different relationship condition. Referring firstly to a "constant effect" link representation 118, this link representation indicates a relationship condition between first and second objects, represented by the state nodes 115 and 117 respectively, wherein the first object has an effect on the second object, and this effect is independent of any values of parameters associated with the first or second node. In one embodiment the link representation 118 represents the effect as constant over the duration of a simulation

operation. The link representation 118 is distinguished in that the tail portion of the link representation is spaced from the circle behind the arrowhead. A "proportional effect" link representation 120 represents a relationship condition between first and second objects wherein the first object has an effect on the second object, and the magnitude of this effect is dependent on the value of a parameter of the first object, represented by state node 115. The link representation 120 is distinguished in that the tail portion thereof contacts the circle behind the arrowhead. Referring to **Figure 9**, examples of "proportional effect" link representations are indicated at 120a and 120b. Specifically, the "adult rabbit" state node 156 is shown to have a proportional effect on the "young rabbit" state node 158 in that the number of young rabbits will increase in proportion to the number of adult rabbits. The same holds true for the link representation 120b, which represents the effect of the "adult wolves" node 162 on the "young wolves" node 164.

An "interaction effect" link representation 122 represents that a first object, represented by the state node 115, has an effect on a second object, represented by state node 117 and that the effect is dependent on the values of parameters of both the first and second objects. The link representation 122 is distinguished in that the tail portion of the representation engages the circle, and in that an arcuate line extends from the circle to the state node 117. Referring again to **Figure 9**, two examples of "interaction effect" link representations are shown at 122a and 122b. Specifically, the "adult wolves" node 162 is shown to have an effect on the "adult rabbits" node 156. Specifically, as indicated by the "C" in the circle 170, adult wolves consume adult rabbits. The rate or magnitude of this consumption is determined by both the number of adult wolves and the number of adult rabbits, and the effect of the object represented by node 162 on the object represented by node 156 is accordingly dependent on the value of parameters (e.g. population numbers) associated with each of these objects respectively. The same explanation applies regarding the effect of the object represented by the "adult wolves" node 162 on the object represented by the "young rabbits" node 158, between which the "interaction effect" link representation 122b extends.

A "constant conversion" link representation 124, shown in **Figure 7**, represents that instances of a first object represented by the state node 115 are converted to instances of a second object represented by the state node 117. The "constant conversion" link representation 124 further represents

that the number of instances converted is independent of any values of parameters associated with the first or second object. In one embodiment, the link representation 124 denotes this conversion as being constant, and is not effected by external parameters. The link representation 124 is characterized in that a tail portion thereof is thickened relative to the tail portion of the "constant effect" link representation 118, and that this tail portion is spaced from the circle behind the head of the link representation 124.

A "proportional conversion" link representation 126 represents that a number of instances of a first object, represented by the state node 115, are converted to instances of a second object, represented by the state node 117. Further, the link representation 126 indicates that the number of instances converted is dependent on the number of instances of the first object. Referring to **Figure 9**, "proportional conversion" link representations are drawn at 126a and 126b. Specifically, the link representation 126a represents that instances of an object represented by the "young rabbit" node 158 are converted to instances of an object represented by the "adult rabbit" node 156. Further, the number of young rabbits (i.e. young rabbit instances) converted to adult rabbits (i.e. adult rabbit instances) is dependent on the number of young rabbit instances, which exist within the object represented by node 158. Similarly, the number of instances of an object represented by the "young wolves" node 164 that are converted to instances of an object represented by the "adult wolves" node 162 is dependent on the number of instances of the object represented by the "young wolves" node 164. The circles 172 of the "proportional conversion" link representations 126a and 126b are shown to include the letter "S", which indicates that the type of conversion that occurs is a change in state. It will be appreciated that any symbol could similarly be included within the circle of a link representation to provide further information regarding the type of relationship condition which exists between objects represented by nodes between which a link representation extends.

An "interaction conversion" link representation 128 represents that a number of instances of a first object, represented by state node 115, are converted to instances of a second object, represented by state node 117. Further, the "interaction conversion" link representation 128 represents that the number of instances of the first object that are converted to instances of the second object is dependent upon respective numbers of instances of both the first and the second objects. The link representation

128 is distinguished in that the tail portion is thickened relative to the tail portion of the link representation 122, and that an arcuate line extends from a circle within the representation 128 to the state node 117 representing the second object.

From the above description of the link representations 118-128, it will be noted that each link represents a relationship condition between first and second objects as being either an "effect" relationship or a "conversion" relationship. Further, each link representation 118-128 represents the relationship condition as being either constant, proportional or interactive.

The link representations 118-128 shown in **Figure 7** are exemplary, and any appropriate link representations can be used to represent the various relationship conditions described above. It will be appreciated that the relationship conditions that are represented by the various link representations 118-128 are typically defined by the model builder. To this end, examples of mathematical expressions of relationship conditions that may be represented by the various link representations are provided below in Table 1. The mathematical expressions given in Table 1 are for a dynamic simulation model which is driven by the progression of time. The link representations 118-128 could similarly be used in a static model, such as a model for which time may be assigned a user-inputted value.

TABLE 1

LINK TYPE	RELATIONSHIP CONDITION (EQUATION)
Constant Effect Link	$\frac{dT}{dt} = K+...$ where T is the target node and K is a constant.
Proportional Effect Link	$\frac{dT}{dt} = C \bullet S(t)^a +...$ where T is the target node, S is the source node, C is a coefficient, and a is an exponent.
Interaction Effect Link	$\frac{dT}{dt} = C\left(S(t)^a + T(t)^b\right)+...$ where T is the target node, S is the source node, and a and b are exponents. This equation can vary depending on the operation selected in the parameter dialog. The operations available are S+T, S-T, S*T, T/S, and S/T. The equation shown is for S+T.
Constant Conversion Link	$\frac{dT}{dt} = K \bullet R+...$ $\frac{dS}{dt} = -K+...$ where T is the target node, S is the source node, K is a constant, and R is a conversion ratio.
Proportional Conversion Link	$\frac{dT}{dt} = C \bullet R \bullet S(t)^a +...$ $\frac{dS}{dt} = -C \bullet S(t)^a +...$ where T is the target node, S is the source node, C is a coefficient, R is a conversion ratio, and a is an exponent.
Interaction Conversion Link	$\frac{dT}{dt} = R \bullet C\left(S(t)^a + T(t)^b\right)+...$ $\frac{dS}{dt} = -C\left(S(t)^a + T(t)^b\right)+...$ where T is the target node, S is the source node, a and b are exponents, and R is a conversion ratio. This equation can vary depending on the operation selected in the parameter dialog. The operations available are S+T, S-T, S*T, T/S, and S/T. The equation shown is for S+T.

As noted above, each of the link representations 118-128 includes a circle in which a graphical identifier providing further information regarding the relationship condition can be displayed. For example, referring to **Figure 9**, each of the link representations is shown to contain an alphabet letter which provides further information regarding the relationship condition between two respective objects. A GUI may also provide an index table 174, such as shown in **Figure 9**, which provides a key to the identifiers displayed in the circles of the various link representations.

Referring now to **Figure 10**, there is shown a flow chart illustrating a method 180, according to one exemplary embodiment of the invention, of generating a display (or graphical representation) of a simulation model. In one embodiment, the steps of method 180 are performed by a GUI in conjunction with the software core 52 illustrated in **Figure 3**. The method commences at step 182, and proceeds to step 184 where a predetermined set of relationship conditions that may exist between objects in the simulation model are defined. For example, the defined relationship conditions may comprise the six relationship conditions described above with reference to **Figure 7**, and may be expressed in the form of equations. At step 186, a respective link representation is associated with each of the relationship conditions in the set. For example, the link representations 118-128 may each be associated with a distinct relationship condition, as described above with reference to **Figure 7**. Looking now at a minimum construction at step 188, the GUI displays node representations, such as state nodes 115 and 117, for first and second objects. The display of the node representations occurs in response to a user input. At this step a user may also input further information, such as parameter values, equations and documentation, to define the diagram objects (i.e. state and/or function objects) such as those shown at 54 in **Figure 3**.

Having thus generated at least two node representations for display by the GUI, and having defined the objects that underlie these node representations, the user at step 190 then selects a link representation, from the link representations associated with the predetermined set of relationship conditions, to represent a desired relationship condition between the objects represented by the first and second nodes (e.g. to specify an equation or class of equations which define a relationship between the first and second objects). At step 192, a user may then further define the relationship condition between the objects in terms of parameter values. For example, the relationship condition between the objects could be expressed in terms of a dynamic equation such as any one of those provided above in Table 1. Parameters defining the relationship condition may be inputted and defined by using a parameter window 90 such as that shown in **Figure 5**. The user may also optionally specify further information to be included within the link representation. For example, the user may specify an identifier to be incorporated within the circle of any one of the link representations 118-128 shown in **Figure 7**. The method 180 then proceeds to step 194, where the GUI displays the selected link representation to show

the relationship condition that exists between the objects represented by the first and second nodes.

The association of different link representations with each distinct relationship condition of a set is particularly advantageous in that a user viewing a display of a simulation model, such as the simulation model 150 in **Figure 9**, is able immediately to ascertain and understand the relationships between the objects represented by the nodes without having to "drill-down" into the representation or to access additional information windows. The inclusion of identifiers within the respective circles of each of the link representations 118-128, and the provision of a key 174 for each of the identifiers, further enhances understanding of the display.

Figure 11 is a flow chart illustrating an exemplary method 190 of selecting a link representation from a predefined set to represent a relationship condition between objects represented by first and second nodes within a diagram window. The method 190 commences at the step 200 and then proceeds to decision box 202, where a determination is made as to whether the relationship condition between the first and second objects is an "effect" relationship or a "conversion" relationship. If it is determined that the relationship condition is properly classified as an effect relationship, the method proceeds to decision box 204, where a determination is made as to whether the effect is constant. If so, an appropriate link representation (e.g. link representation 118) is selected at step 206. If the effect relationship is not constant, the method 190 proceeds to decision box 208, where a determination is made as to whether the effect of the one object on the other is dependent on the value of a parameter associated with one of the objects. If so, the method proceeds to step 210, where an appropriate link representation (e.g. link representation 120) is selected. Alternatively the method 190 proceeds to decision box 212, where a determination is made as to whether the effect of one object on the other is dependent on values of parameters associated with each of the objects. If so, the method 190 proceeds to step 214, where an appropriate link representation (e.g. link representation 122) is selected. From step 214, the method terminates at step 216.

Returning to decision box 202, if it is determined that the relationship condition between the objects is not an "effect" relationship condition, but rather a "conversion" relationship condition, the method proceeds from decision box 202 to decision box 218, where a determination is made as to whether the "conversion" relationship condition requires that a constant

number of instances of one object be converted to instances of the other object. If so, the method proceeds to step 220, where an appropriate link representation (e.g. link representation 214) is selected. Alternatively, the method proceeds to decision box 222, where a determination is made as to whether the "conversion" relationship condition specifies that the number of instances of the one object that are converted to instances of the other object is dependent on the number of instances of either one of these objects. If so, the method proceeds to step 224, where an appropriate link representation (e.g. link representation 126) is selected. Alternately, the method proceeds to decision box 226, where a determination is made as to whether the "conversion" relationship condition requires that the number of instances of one object that are converted to instances of the other object is dependent on respective numbers of instances in both the first and second objects. If so, an appropriate link representation (e.g. link representation 128) is selected. From step 228, the method 190 proceeds to terminate at step 216.

Returning to **Figure 8**, a model builder may wish to model a situation where a relationship condition, represented by a link representation 138, between objects represented by nodes 130 and 132, is influenced by a third object, represented by node 134. According to one embodiment of the present invention, overlaying the relevant link representation 138 with a modifier representation 136 may represent this influence of the third object on the relationship condition between the first and second objects. The node 134 representing the third object is further shown to feed into the modifier representation 136 by a link representation 137. The modifier representation 136 includes an identifier that provides an indication of the type of effect the third object has on the relevant relationship condition. For example, as it is case in **Figure 8**, the modifier representation 136 may include a "+" symbol, which may indicate that the third node has a stimulating effect on the relevant relationship condition. Referring to **Figure 9**, modifier representations 136a and 136b each include the "+" symbol, which indicates that a third object (i.e. a "rabbit food supply" object) has a stimulating effect on the production by "adult rabbits" object instances of "young rabbits" object instances. The modifier representation 136 represents an analogous situation with respect to the wolf population in the simulation model 150.

In one embodiment of the present invention, a modifier representation may include a "-" symbol to represent that a third object has

an inhibiting effect on the relationship condition between a pair of objects. Further, a modifier representation may include a "=" symbol which indicates that the third object may have either an inhibiting or stimulating effect on the relationship between a pair of objects, dependent on the value of a parameter of the first object, the third object or a modifier object represented by the modifier representation. For example, a parameter of the third object could have a certain threshold below which the third object has a stimulating effect on the relationship condition and above which the third object has an inhibiting effect on the relationship condition.

To this end, reference is now made to **Figure 12** which illustrates a method 230, according to one embodiment of the invention, of displaying a modifier representation which represents the influence of an object on a relationship condition between a pair of objects. The method commences at step 232, and proceeds to step 234 where a node representation for a third object of the simulation model is displayed within a diagram window in response to a user input. The method then proceeds to decision box 236, where determination is made as to whether the third object influences a relationship condition, represented by an appropriate link representation, between two objects represented in the diagram window by respective node representations. If not, the method terminates at step 238. If so, the method proceeds to decision box 240, where a determination is made as to whether this influence is inhibiting. If so, a modifier representation including an appropriate identifier (e.g. "-") is selected at step 242. Alternatively, the method 230 proceeds to decision box 244, where a determination is made as to whether the influence is stimulating. If so, a modifier representation, including an appropriate identifier (e.g. "+") is selected at step 246. Alternatively, the method proceeds to decision box 248, where a determination is made as to whether the influence is dependent on a value of a parameter of the third object. If so, a modifier representation including an appropriate identifier (e.g. "=") is selected at step 252.

The identifiers mentioned above are, of course, merely exemplary and any suitable identifier could be included within a modifier representation to advertise a characteristic of an underlying modifier object. For example, modifier representations including the "A" or "B" could also be used to identify the underlying modifier object as "allowing" or "blocking" a relationship condition (e.g. an effect or conversion relationship condition) between a two objects.

From steps 242, 246 or 252, the method 230 proceeds to step 252, where the selected modifier representation is overlaid on a link representation between nodes for the first and second objects, and a link representation is generated between a node representation for the third object and the modifier representation. The method then terminates at step 238.

The modifier representation represents an underlying modifier object, which may in turn be expressed in terms of an equation. In one embodiment, the modifier object may comprise a function of parameters within the simulation model. An example of an equation expressing a modifier object is provided below in Table 2.

TABLE 2

Modifier	$\frac{dT}{dt} = M \bullet f\left(\frac{S(t)}{N}\right) \bullet linkterm+...$ where T is the target node, M is a multiplier constant, N is a normalization constant, F() is a function, either linear or specified by a transformation curve, and link term is the an link term. Note that modifiers also have an additive or multiplicative setting; this setting specifies whether multiple modifiers on a link should add or multiply their terms together before being multiplied by the link term.

In Table 2, the "link term" expression refers to an equation which is embodied in a link object to express a relationship condition between two objects. Examples of such equations are provided above in Table 1 in the "RELATIONSHIP CONDITION (EQUATION)" column.

Panels - Accessing Selecting Parameters

While **Figure 9** shows a simple simulation model 150 for the purposes of explanation, it will be appreciated that the graphic elements described above may be used to generate a model for a very complex system, such as an aspect of human physiology or a complex manufacturing system. In such a case, the simulation model may include hundreds or even thousands of objects, each of which may include a number of parameters. In order to perform effective "what-if" analyses using a simulation model, it is useful to access and observe the input values of certain key parameters prior to performance of a simulation operation, and also possibly to observe output values for these key parameters at the conclusion of such an operation. As many parameters are included in the expression of, and are effected by, a relationship between two objects, a modeler may also need to

examine certain parameters at either end of such a relationship. For example, a modeler may wish to examine parameters that specify the effects a specific object has on a number of other objects, and also parameters that specify the effects of these other objects upon the specific object. Complex models are also often broken down into a system of sub-models, either using software features or merely by the modeler's convention. It is accordingly often useful for the modeler simultaneously to view selected parameters contained within a specific sub-model. The satisfaction of this need is complicated by the fact that the boundaries of a sub-model may not be mutually exclusive with respect to parameters, i.e., a single parameter may appear in many sub-models. Further, the boundaries of sub-models often change as the model evolves.

A modeler utilizing modeling software may be required to invoke a parameter window for each variable the modeler wishes to access or view. It will be appreciated that the generation of a number of such windows, as may typically occur for a complex model, quickly result in a cluttered GUI. The managing and viewing of such a multitude of parameter windows is also awkward.

In order to accommodate the need to view and access a user-selected group of parameters within a simulation model, the present invention proposes providing a mechanism by which the modeler can create groupings of parameter "aliases" within windows or panels that can be overlaid, or displayed along side, a diagram window and that are distinct from the diagram window. The present invention further proposes allowing a modeler to create multiple access panels, each panel providing access to a user-selected group of parameters collected from an object or group of objects within the simulated model. **Figure 13** provides examples of two access panels 260 and 262, each of which contains information regarding user-selected parameters of objects within the exemplary simulation model 150 shown in **Figure 9**. More specifically, the access panel 260 displays information for various parameters pertaining to the rabbit population modeled by the objects within the simulation model 150. Similarly, the access panel 262 displays information regarding parameters pertaining to the predator population represented by objects in the simulation model 150. Each of the access panels 260 and 262 has four columns of information, namely an object column 264, a symbol column 266, a working value column 268 and a unit column 270. The object column 264 includes both a node representation (or icon) and a text

description for the object for each parameter included within the panel. As is apparent, objects from which parameters may be extracted include state objects, function objects, link objects, and modifier objects, which are collectively designated as being diagram objects 54 in **Figure 3.** The symbol column 266 contains a specified symbol for each of the parameters included in the respective access panel, while the working value column 268 includes a working value for each of the parameters. In one embodiment of the invention, each working value comprises a fixed input value for a parameter, and is thus not effected by performance of a simulation operation. In another embodiment, the working value may comprise a current value for the parameter, and may thus reflect an output value at conclusion of a simulation operation. In this embodiment, as a simulation operation using the simulation model 150 proceeds, the working value displayed within the working value column 268 for each parameter may change and continually be updated. It is furthermore significant that a user is able to access a working value of the parameter in the column 268 by selecting an area within a panel in which the relevant working value appears. The user may then be able to change this parameter, and rerun a simulation operation, thus making the performance of "what-if" analyses particularly easy. The unit column 270 specifies the units for the relevant parameter.

The panels 260 and 262 shown in **Figure 13** are also advantageous in that they allow a user-selected group of parameters to be viewed together. The panels 260 and 262 provide a number of other features to a user to enhance the user-friendliness of a simulation model. Specifically, by selecting (e.g. by double clicking on) the icon for any particular object within the panel, the user is able to invoke a parameter window 90, such as that shown in **Figure 5**, for the relevant parameter.

Each of the access panels 260 and 262 further includes navigation icons 272 and 274, which may be selected by user (e.g. by double clicking thereon) to move between panels. For example, assuming that the "predator" panel 260 is currently active and displayed (i.e. accessible by a user) and that the "rabbit" panel 260 was also defined by the GUI, a user could select either the "backward" navigation icon 272 or "forward" navigation icon 274 to place the panel 262 in an inactive state and to place the panel 260 in an active and displayed state, in which a user can access the working values of the parameters of the panel 260. It will be appreciated that where more than two panels are defined by a GUI, the navigation icons

272 and 274 provide a convenient and user-friendly method of navigating between such panels.

Each panel also includes a list icon 276, which is situated between the navigation icons 272 and 274. Upon user selection of the list icon 276 on the panel, a drop-down menu (not shown) including a list of all panels extant within the GUI is invoked. The list may contain a name for each extant panel, and may provide an identifier, such as a check mark, adjacent the name of the currently active panel. From this list, a user is then able to designate another panel as the active panel.

Each of the access panels 260 and 262 is representative of an underlying panel object 56, such as those shown in **Figure 3.**

Methods of creating an access panel, such as panel 260 in **Figure 13**, and an associated panel object 56, and of accessing parameters within such a panel, are described with reference to **Figure 14** . The method commences at step 282, and then proceeds to step 284, where a panel is displayed within the GUI. The panel represents an underlying panel object 56, which is dedicated to receiving and storing input values for user-designated parameters within the simulation model. The panel is displayed in response to the user selection of an appropriate menu item. At step 286 a modeler (also termed a "user") identifies a parameter of an object for inclusion within the displayed panel. According to one exemplary method, the modeler may identify a parameter to be included within a panel by performing a drag-and-drop operation using the symbol of a parameter from an appropriate parameter window 90, such as that shown in **Figure 5**. In a further method, the modeler may perform a similar drag-and-drop operation utilizing the symbol for a parameter that appears in a previously defined panel. Referring back to **Figure 13**, it will be seen that each of the panels 260 and 262 also includes a transfer icon 278 in the upper left-hand corner. By performing a drag-and-drop operation with respect to the transfer icon 278, a modeler is able to transfer all parameters within a particular panel to another panel in one operation. The transfer icon 278 is advantageous in that it allows duplicates of panels to be created with relative ease. Returning now to **Figure 14**, at step 288, an icon and name for the object to which the parameter belongs, a parameter symbol, a parameter working value and parameter units are displayed within the panel, as explained above with reference to the access panels 260 and 262. At step 290, a value for at least one parameter is received via the panel as the working value for the parameter. In one embodiment, the working value is an

input value which is maintained within the panel, and which is not effected by performance of the simulation operation. It will of course be appreciated that a modeler may input values for any number of parameters within a panel in this manner. At step 292, in response to the value being inputted via the panel at step 290, inputted value is assigned to the appropriate parameter. At step 294, a user then initiates a simulation operation in which the value assigned to the parameter is utilized by the simulation model as an input value in performance of the simulation operation. In one embodiment, the working value is not altered in response to performance of the simulation operation. However, in an alternative embodiment, during and/or on completion of the simulation operation, output values for parameters included within the panel may optionally be displayed in the working value column 268, at step 196. The method then terminates at step 298.

Layers-Storing Values for a Set of Parameters

As discussed above with reference to **Figure 5**, each parameter of each object within a simulation model may have three values associated therewith, namely a working value, a baseline value and a comparison value. These are displayed in the settings section 96 of a parameter window 90 for the relevant parameter. The collections of working, baseline and comparison values for all parameters within a simulation model may conveniently be termed working, baseline and comparison foundations. As described in reference to **Figure 5**, the model builder typically specifies the baseline parameter values that constitute the baseline foundation when a simulation model is constructed.

Taking the simulation model 150 shown in **Figure 9** as an example, it will be appreciated that a particular set of baseline parameter values (of a baseline function) may represent a particular model scenario or condition. For example, parameters associated with the "adult wolves" and "young wolves" objects represented by nodes 162 and 164 respectively may represent a scenario in which the predators comprise wolves. However, by assigning a different set of values to the relevant parameters, a scenario in which the predators comprise coyotes may be represented and modeled. Accordingly, it will be appreciated that a set of values for a particular set of parameters may correspond to a specific model scenario. Such sets of parameters can be viewed as "macros" for configuring the model. In order to allow a user conveniently to substitute one scenario within the simulation model with

another (e.g. substituting a wolf predator scenario with a coyote predator scenario), the present invention proposes a mechanism by which a modeler can create groupings of parameter "aliases" which may conveniently be termed "layers". Each layer comprises a set of parameter values that may be substituted for a set of parameter values within the baseline foundation. In contradistinction to the panels discussed above with reference to **Figures 13** and **14**, a layer comprises a set of parameter values which are stored at a separate location from the baseline values, so as to allow the user to switch between the values contained in one or more layers and the baseline values.

Reference is made to **Figure 15** which shows a diagrammatic representation of a baseline foundation 300 containing a baseline layer 302, which comprises a set of baseline values for a predetermined set of parameters. The baseline layer 302 is shown to be replaceable by any one of three layers 304-308, each of which contains a set of alternative values for the predetermined set of parameters.

Each of the layers 304-308 has a respective layer panel 310-314 associated therewith, which allows a modeler to construct the respective layer and to then view the values attributed to the various parameters which are included in the associated layer. Each of the layer panels 310-314 is similar in appearance and layout to the access panels shown in **Figure 13**, but the propose and functioning of the layer panels are fundamentally different from that of the access panels. Turning now to **Figure 16**, exemplary layer panels 310 and 312 are illustrated. Each of the panels 310 and 312 includes:

(1) An object column 320, which contains an icon and text identifier for the object to which these respective parameter belongs;

(2) A symbol column 322, which contains a symbol for each parameter included in the panel;

(3) A layer value column 324, which contains a layer value for each parameter of the panel; and

(4) A units column 326, which indicates the units in which the respective layer values are expressed.

A modeler is able to include parameters within a layer panel in the same manner utilized to construct an access panel. It will be noted that the layer panels 310 and 312 shown in **Figure 16** each include an identical set of the parameters, but that each layer panel 310 and 312 attributes different layer values to each of these parameters. Specifically, the layer panel 310 attributes values to the relevant parameters which models a scenario within the simulation model 150 shown in **Figure 9** in which the predators comprise wolves. Similarly, the layer panel 310 attributes values to the relevant parameters which models a scenario within the simulation model 150 in which the predators comprise coyotes. For example, panel 310 specifies wolves as having a life expectancy of 240 months, whereas panel 312 specifies coyotes as having a life expectancy of 180 months. Similarly, the reproductive rate of coyotes is specified to be higher than that of wolves and the time required for young wolves to mature into adults is also specified to be longer than the period required for young coyotes to mature. Each of the layer panels represents an underlying layer object 58, as illustrated in **Figure 3**.

Figure 17 shows a configuration panel 330 that may be utilized to apply and de-apply layers within a simulation model. Specifically, the configuration panel 330 includes a layer selection section 332, which lists the layers applicable to a simulation model. Utilizing an apply icon 334 (i.e. the "+" icon) and a de-apply 336 (i.e. the "-" icon), a user is able to apply and de-apply layers to the simulation model. Layers, which a user has the option of applying or de-applying, are listed in the list section 337, which in the illustrated example lists no layers. However, it will be appreciated that any number of layer panels may be a named within the list section 337. The configuration panel 330 further includes a foundation section 338, using which the working, baseline or comparison foundation may be selected as the foundation that the selected layer is to override. A period sector 340 allows a user to specify a time period over which a simulation operation by the simulation model is to be performed, while a delta-time (dt) section 342 allows a user to specify the time increment by which the simulation model progresses through the period specified in the period section 340. A store section 344 allows a user to specify the time increment at which values for parameters within the simulation model are stored. In the illustrated example, the simulation model will run over a 5000 hour period in increments of one hour, and values for parameters of the simulation model will be stored for every ten hour increment.

Figure 18 shows a flowchart illustrating a method 350, according to one embodiment of the present invention, of the managing objects within a simulation model. The method commences at step 352, and then proceeds to step 354, where a first set of values for parameters of objects within the simulation model are stored. A model builder typically specifies these initial values upon construction of the original model. At step 356, a layer panel is displayed within the GUI, typically in response to a user indication. For example, the user may select a menu item from a drop-down menu for the creation of a layer panel. Once the layer panel is displayed within the GUI, at least one parameter of the simulation model is identified for inclusion within the layer panel at step 358. This may be done in the same manner as parameters are included within access panels as described above with reference to the flowchart in **Figure 14**. At step 360, and in response to the user identification of parameters at step 358, object and parameter identifiers, and the first values for the selected parameters, are displayed within the layer panel. At step of the 362, the user then inputs second (alternative) values for the selected parameters into the appropriate blocks within the layer value column 324 of the layer panel. At step 364, the core program 52 creates a layer object 58, and stores the second values for the selected parameters (e.g. on a hard disk), separate from the location at which the first values for the selected parameters are stored. Thus, the initial values for the selected parameters are available for recall and are not overwritten. At step 366, the user selects either the set of first values, or any set of alternative values, (e.g. the set of second values) for use within the simulation model. Using the configuration panel 330, the user may apply or de-apply layers within the simulation model to the thereby model different scenarios, and perform "what-if" analyses in an easy and convenient manner. At step 368, the user then performs a simulation operation utilizing the simulation model including the selected set (or layer) parameter values by selecting the "run" icon 339 provided on the configuration panel 330. The method 350 then terminates at step 360.

Monitors

The present invention further proposes a mechanism by which parameters within a simulation model can be monitored in an intuitive and useful manner. In one embodiment of the present invention, a proposed mechanism is particularly useful for viewing a curve of a transform that expresses a specified relationship between parameters of

objects within the simulation model. The curve of the transform is typically the plot of an equation in which the parameters comprise co-efficients. Alternatively, the curve may be determined in some other manner, for example by experimentation, and then utilized to express the relationship between the relevant parameters. Referring now to **Figures 9** and **19**, **Figure 19** shows monitor panels 380 and 382, which plot curves expressing the relationships between food supply and reproduction for the rabbit and predator populations respectively within the simulation model 150. The relationships between the food supply and reproduction parameters expressed by the plotted curves of the monitor panels 380 and 382 are embodied within the modifier objects represented by the modifier representations 136a and 136b shown in **Figure 9**.

Each of the monitor panels 380 and 382 includes a header 384, and an object identifier 386 which identifies the object that embodies the transform represented in the respective monitor panel. Accordingly, each of the monitor panels 380 and 382 shows a respective modifier icon at this location. Each object identifier 386 behaves in the same manner as described above for the access and layer panels and by double clicking on this icon, the same result is produced as by double clicking on a corresponding object identifier (or icon) within a diagram window

Each of the monitor panels 380 and 382 further includes both X and Y parameter identifiers 388 and 390 which identify the respective X and Y input parameters for the plotted transform and also display current numeric values for each of these parameters at any given time. A monitor line 392 is also provided within each of these the monitor panels 380 and 382, and identifies the X input parameter value within the plotted a transform. Monitor panels may be used in conjunction with a further chart panel, as discussed below with reference to **Figures 20a** and **20b**, to monitor the interaction of various parameters in a graphical manner and coordinated manner. For example, a user could thus monitor the effect of an adult rabbit population increase on predator reproduction rates in a graphical manner. This feature is particularly useful for developing dynamic models which include non-linear functions, as it is often useful to attempt to correlate an event in the model's behavior (e.g. observed in a plot of a model parameter value verses time) with a change from one domain to another domain in a non-linear function. For example, a user may wish to determine whether the non-linearity at a saturation point of a sigmoidal function is responsible for a temporal plateau in an output value

of a parameter. Specifically, such a cause-effect relationship can be identified if the X parameter output value of the plotted function moves into a saturated region of the sigmoid (i.e. the dependent parameter value Y asymptotically approaches a maximum value Ymax as X exceeds a saturation point Xsaturation) at the same time that an output parameter value ceases to vary and remains constant over some time interval. Assuming that X is an input parameter value (i.e., time invariant) for the model, then one simply needs to look at a plot for the X output parameter value verses time and compare that to the Xsaturation during the period of interest for the output parameter value. It will be appreciated that the process is much more difficult when there is not some critical value, such as Xsaturation, in a function but rather a smooth transition between function domains. Difficulty is also increased when a user is attempting to correlate multiple events or multiple non-linear functions, or a function is not solely defined by time variant parameters (e.g., Xsaturation) but is at least partially defined by model parameter values which vary with time. In this case, not only is the independent output parameter value X changing with time, but the curve of the transform is changing with time as well.

To address the above identified needs, the present invention proposes creating a first graphical user interface (GUI) element in the form of a monitor panel, such as that shown in **Figure 19**, which plots a non-linear function over a specified domain of a selected X parameter output value and creating a second graphical user interface element (GUI) in the form of a chart panel 402, such as that shown in **Figure 20a** which plots at least one further model parameter value against time. Utilizing a monitor panel in conjunction with a chart panel it is possible, in a coordinated fashion, to monitor (both of graphically and directly) output values for X and Y parameters on a non-linear curve plot, and simultaneously to observe the value of a further dependent parameter value at a specific time on the chart panel. A monitor line 392 also provides "animation" of the function expressed by the curve plot of a monitor panel as a simulation models advances through time, or as a time parameter is attributed a certain value.

In another embodiment of the present invention, the Y parameter output value may be expressed as the function of a X parameter value and one or more further parameter values of the simulated model, and the curve plotted within the monitor panel may thus vary as time changes during a simulation operation.

To illustrate the above-described aspect of the present invention, reference is made to **Figures 20a** and **20b**, which show screen prints of a GUI 400 diagramming the simulation model 150 shown in **Figure 9**. The GUI 400 is shown to include a diagram window 401 overlaid with a chart panel 402, a monitor panel 404 and a configuration panel 330. As is apparent from the configuration panel 330, a single layer is applied within the simulation model, namely a "wolves" layer. The chart panel 402 plots both "Young Predators" and "Adult Predators" parameters over time, while the monitor panel 400 displays the curve defining the relationship between "Predator Food Supply" and "Reproduction Regulation" parameters. Referring firstly to **Figure 20a**, as indicated within the chart panel 402, the simulation operation has progressed 74.6 months, this time being indicated by a time line 398. It is apparent that the "Young Predator" parameter shown in the chart panel 402 experiences a peak at just after this time within the simulation operation. The peak can be attributed to a drop in the food supply (i.e. rabbit population) to a level that new births of predators cannot be sustained. The monitor panel 400 illustrates the respective X and Y parameter values (i.e. the "Predator Food Supply" and" Reproduction Regulation" parameter value outputs) at this time utilizing monitor line 392, which intersects a point of the plotted curve. The Y parameter output value is shown to have "plateauxed" at this time, as shown in **Figure 20a**, indicating that the birth rate is at its maximum (i.e. the birth rate has become saturated). Referring now to the **Figure 20b**, the GUI 400 is shown 76.8 months into the simulation operation, and the time line 398 is according shown to have advanced beyond the peak of the "Adult Predator" parameter. The monitor line 392 is also shown to have retreated to the left, thus indicating that the X parameter output has no notable influence on the relevant Y parameter output at this time. Specifically, the effect of the drop in food supply levels, and the resulting drop in predator births, is reflected in the monitor panel. The net effect is that the predator birth rate drops to zero, the number of young predators drops as they mature into adult predators, and the adult predator population continues to rise, but then begins to plateau.

Referring now to **Figure 21a**, there is illustrated a method 410 of monitoring parameters within a simulation model, according to one embodiment of the present invention. The method 410 commences at step 412, and then progresses to step 414, where a relationship function between the first and second parameters of objects within the simulation model is

defined. For example, the relationship could be expressed as a mathematical equation, or could alternatively be identified by experimentation. In one embodiment of the present invention, the relationship condition between the first and second parameters is expressed as a non-linear transform. The method 410 then proceeds to step 416, where a first graphical plot is generated within a monitor panel, this the first graphical plot illustrating the defined relationship function between the first and second parameters. The method 410 then proceeds to step 418, where a second graphical plot is generated within a chart panel, the second graphical plot plotting a third parameter value against time. At step 420, a simulation operation is performed utilizing the simulation model, which includes first, second and third parameters. At step 422, the respective values of the first and second parameters are indicated within the first graphical plot, and a corresponding value for the third parameter is indicated on the second graphical plot. As explained above, the values for the first, second and third parameters may be indicated using the monitor and the time lines 392 and 398 respectively. The method 410 then terminates at step 423.

In an alternative embodiment of the present invention, the first graphical plot could represent the relationship among three parameters, and the X-axis of the monitor chart could be a function of two parameters. In this case, the plot shown in the monitor panel would vary as time progressed during a simulation operation.

Figure 21b illustrates a further method 424 of monitoring parameters within a simulation model. The method 424 commences at step 425, and then progresses to step 426, wherein a predetermined relationship function between first and second parameters of objects within the simulation model is defined, for example, as a mathematical equation. At step 427, a first graphical plot of the defined relationship function between the first and second parameters is generated. At step 428, a second graphical plot of a third parameter output value vs. time is generated within a chart panel. At step 429, an output value for either of the first, second or third parameters is specified. For example, a user may advance or retreat a monitor line 392 within a monitor panel to a specific location to thereby specify first and second parameter output values. Alternatively, the user could specify a time value on the second graphical plot and this way specify an output value for the third parameter.

At step 430, respective output values for the first and second parameters are indicated on the first graphical plot within the monitor panel. For example, the monitor line 392 may be advanced or retreated to indicate output values for the first and second parameters. In a coordinated fashion, an output value for the third parameter is similarly indicated on the second graphical plot, for example, by graphically highlighting a location on the second graphical plot. In this way, output values for the first, second or third parameters corresponding to the specified output value are indicated. The method then terminates at step 431.

It will be appreciated that the method 410 described with reference to **Figure 21a** is time-driven, with the indicated output values for the first, second or third parameters being driven by the progression of time or by attributing a selected time value to a time parameter. In contrast, the method 424 described above with reference to **Figure 21b** is parameter-driven, in that the output values for the first, second or third parameters is determined by a specified output value for either the first, second or third parameter.

Exemplary Simulation Model

Figure 22 illustrates a further exemplary simulation model 430, which is constructed utilizing the graphical elements and notation illustrated in **Figures 6-8**, and which again models a predator-prey system. The functioning of the simulation model 430 can readily be ascertained utilizing the meanings attributed to the various graphical elements described above.

Figure 23 shows a GUI 440, which includes a diagram window 442 in which the simulation model 430 is displayed. The GUI 440 further comprises a first access panel 444, which includes parameter information and working values for parameters relating to the birth and death rates of the predators (i.e., wolves) within the simulation model 430. The access panel 444 is further shown to be in an active state. A second access panel 446 is shown to include parameter identification and working values for parameters relating to the starting populations of both predators and prey within the simulation model 430. The access panel 446 is further shown to be in an inactive state. It will be appreciated that by grouping the shown parameters in the respective access panels 444 and 446, access to, and viewing of, related parameter values is simplified and made convenient.

Figure 24 shows GUI 450, which includes a diagram window 452 in which the simulation model 430 is again displayed. The GUI 450 includes a layer panel 454, which displays parameter identification information and layer values which characterize the predators within the simulation model 430 has being "hardy" wolves. The GUI 440 further includes a layer panel 456, which displays parameter identification information and layer values that characterize the predators within the simulation model 430 as being "wimpy" wolves. A configuration panel 460 is also included with the GUI 450, and contains a layer section 462, in which the layer represented by the layer panel 456 (i.e., the "wimpy" wolf layer) is shown as being the applied or selected layer. Accordingly, the layer values shown within the layer panel 456 will be attributed to the relevant parameters on performance of a simulation operation utilizing the simulation model 430.

Computer System

Figure 25 shows a diagrammatic representation of a computer system 500 within which software for performing the methodologies discussed above, and for generating a GUI according to the teachings of the present invention, may operate. The computer system 500 includes a processor 502, a main memory 503 and a static memory 504, which communicate via a bus 506. This system 500 is further shown to include a video display unit 508 (e.g., a liquid crystal display (LCD) or a cathode ray tube (CRT)) on which a GUI according to the present invention may be displayed. The computer system 500 also includes an alpha-numeric input device 510 (e.g. a keyboard), a cursor control device 512 (e.g. a mouse), a disk drive unit 514, a signal generation device 516 (e.g. a speaker) and a network interface device 518. The disk drive unit 514 includes a computer-readable medium 515 on which software 520 for executing each methodology described above and for generating the various GUI elements comprising the invention is stored. The software 520 is also shown to reside, completely or at least partially, within the main memory 503 and/or within the processor 502. The software 520 may further be transmitted or received via the network interface device 518. For the purposes of this specification, the term "computer-readable medium" shall be taken to include any medium which is capable of storing or encoding a sequence of instructions for performing the methodologies of the present invention, and shall be taken to included, but not be limited to, optical and magnetic disks, and carrier wave signals.

Thus, a method of managing objects within a simulation model has been described. Although the present invention has been described with reference to specific exemplary embodiments, it will be evident that various modifications and changes may be made to these embodiments without departing from the broader spirit and scope of the invention. Accordingly, the specification and drawings are to be regarded in an illustrative rather than a restrictive sense.

CLAIMS

What is claimed is:

1. A method of managing objects within a simulation model, the method comprising the steps of:

storing a first value for a parameter of an object of the simulation model;

displaying a first panel within a user interface;

identifying the parameter of the object of the simulation model for inclusion within the first panel;

in response to the identification of the parameter, displaying an identifier for the parameter of the object within the first panel;

receiving a second value for the parameter of the object via the first panel; and

storing the second value for the parameter of the object separately from the first value for the parameter of the object.

2. The method of claim 1 including the step of, in response to a user selection, performing a first simulation utilizing the simulation model inclusive of the first value for the parameter of the object, or performing a second simulation utilizing the simulation model inclusive of the second value for the parameter of the object.

3. The method of claim 1 including the step of displaying an identifier for the object within the first panel.

4. The method of claim 1 wherein the step of identifying the parameter comprises detection of a user selection of the parameter.

5. The method of claim 4 wherein the step of detecting a user selection comprises detecting a drag-and-drop operation performed by the user on an identifier associated with the parameter.

6. The method of claim 1 including the steps of identifying a plurality of parameters of the simulation model for inclusion within the first panel, displaying an identifier for each of the plurality of parameters within the first panel, receiving a second value for each of the plurality of parameters within the first panel, and storing the second value separate from a first value for each of the plurality of parameters within the first panel.

7. The method of claim 1 including the step of defining a plurality of panels within the user interface, identifying a respective parameter for inclusion within each of the plurality of panels, displaying a respective identifier for the respective parameter within each of the plurality of panels, receiving a second value for the respective parameter within each of the plurality of panels, and storing the second value separate from a first value for the respective parameter within each of the plurality of panels.

8. The method of claim 1 including the step of identifying the first panel as being in an applied state in which the second value for the parameter of the object is included in the simulation model, and identifying a second panel as being in an unapplied state in which a further alternative value for the parameter of an object included within the second panel is not included in the simulation model.

9. The method of claim 8 including the step of displaying a navigation icon within the first panel, detecting a user selection of the navigation icon, and transferring the first panel to an unapplied state and transferring the second panel to an applied state in response to user selection of the navigation icon.

10. The method of claim 8 including the step of displaying a list icon within the first panel, detecting a user selection of the list icon, and displaying respective identifiers for the first and second panels in response to the user selection of the list icon.

11. The method of claim 10 wherein the representation of the first panel indicates the first panel as being in the applied state.

12. The method of claim 10 including the steps of detecting a user selection of the identifier of the second panel and placing the second panel in an applied state in response to the user selection.

13. A method of managing objects within a simulation model, the method comprising the steps of:

selecting a panel within a user interface;

selecting a parameter of an object of the simulation model for inclusion within the panel, the selected parameter having a first value assigned thereto;

in response to the selection, displaying an identifier for the parameter of the object within the panel;

inputting a second value for the parameter of the object via the panel; and

storing the second value for the parameter of the object separately from the first value for the parameter of the object

14. The method of claim 13 including the step of, in response to a user selection, performing a first simulation utilizing the simulation model inclusive of the first value for the parameter of the object, or performing a second simulation utilizing the simulation model inclusive of the second value for the parameter of the object.

15. A computer-readable medium having a sequence of instructions stored thereon which, when executed by a processor, cause the processor to perform the steps of:

storing a first value for a parameter of a object of the simulation model;

defining a first panel within a user interface;

identifying the parameter of the object of the simulation model for inclusion within the first panel;

in response to the identification of the parameter, displaying an identifier for the parameter of the object within the first panel;

receiving a second value for the parameter of the object via the first panel; and

storing the second value for the parameter of the object separately from the first value for the parameter of the object.

16. The computer-readable medium of claim 15 having a sequence of instructions stored thereon which, when executed by the processor, cause the processor to perform, responsive to a user selection, a first simulation utilizing the simulation model inclusive of the first value for the parameter of the object, or to perform a second simulation utilizing the simulation model inclusive of the second value for the parameter of the object.

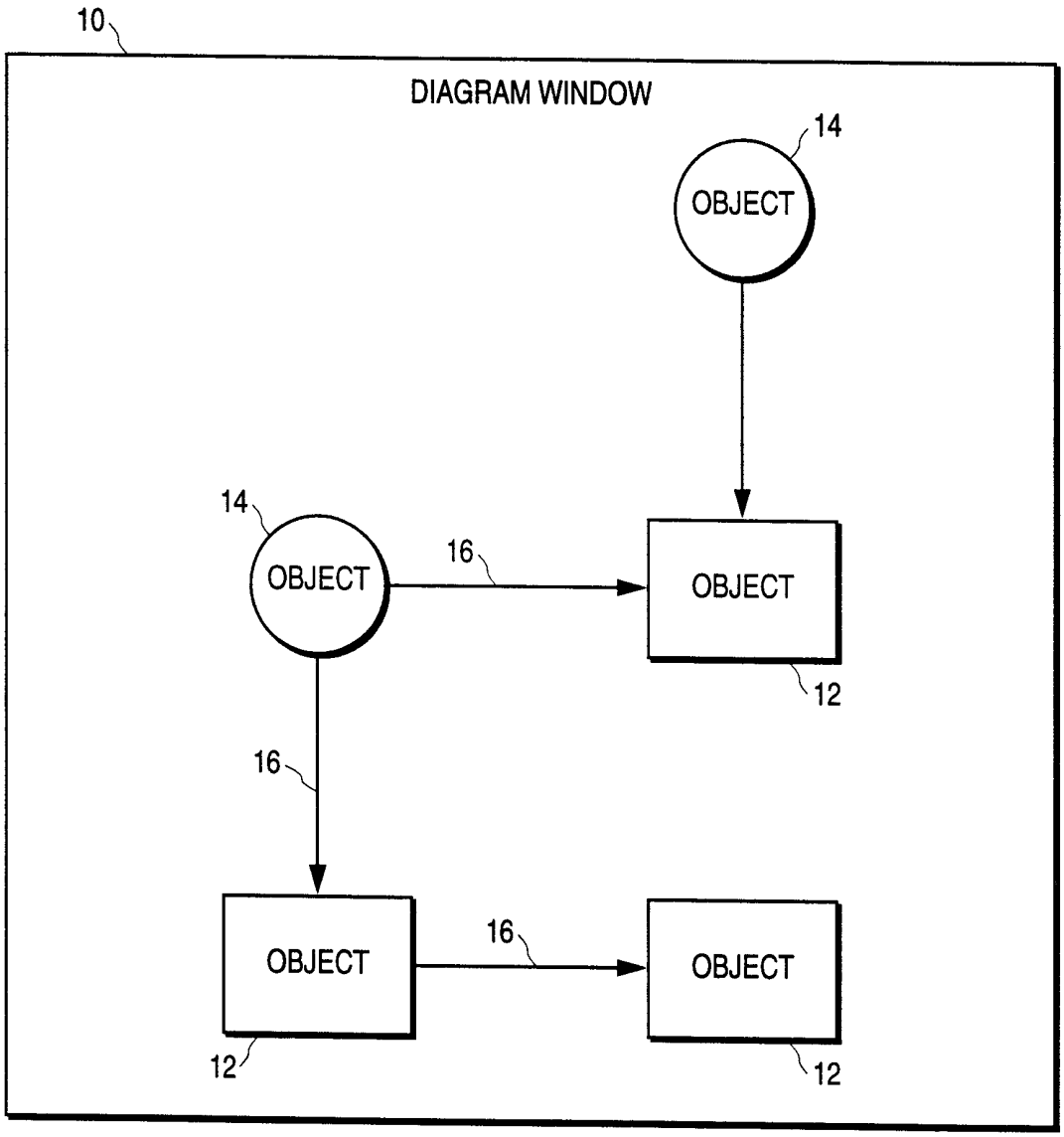


FIG.1 (PRIOR ART)

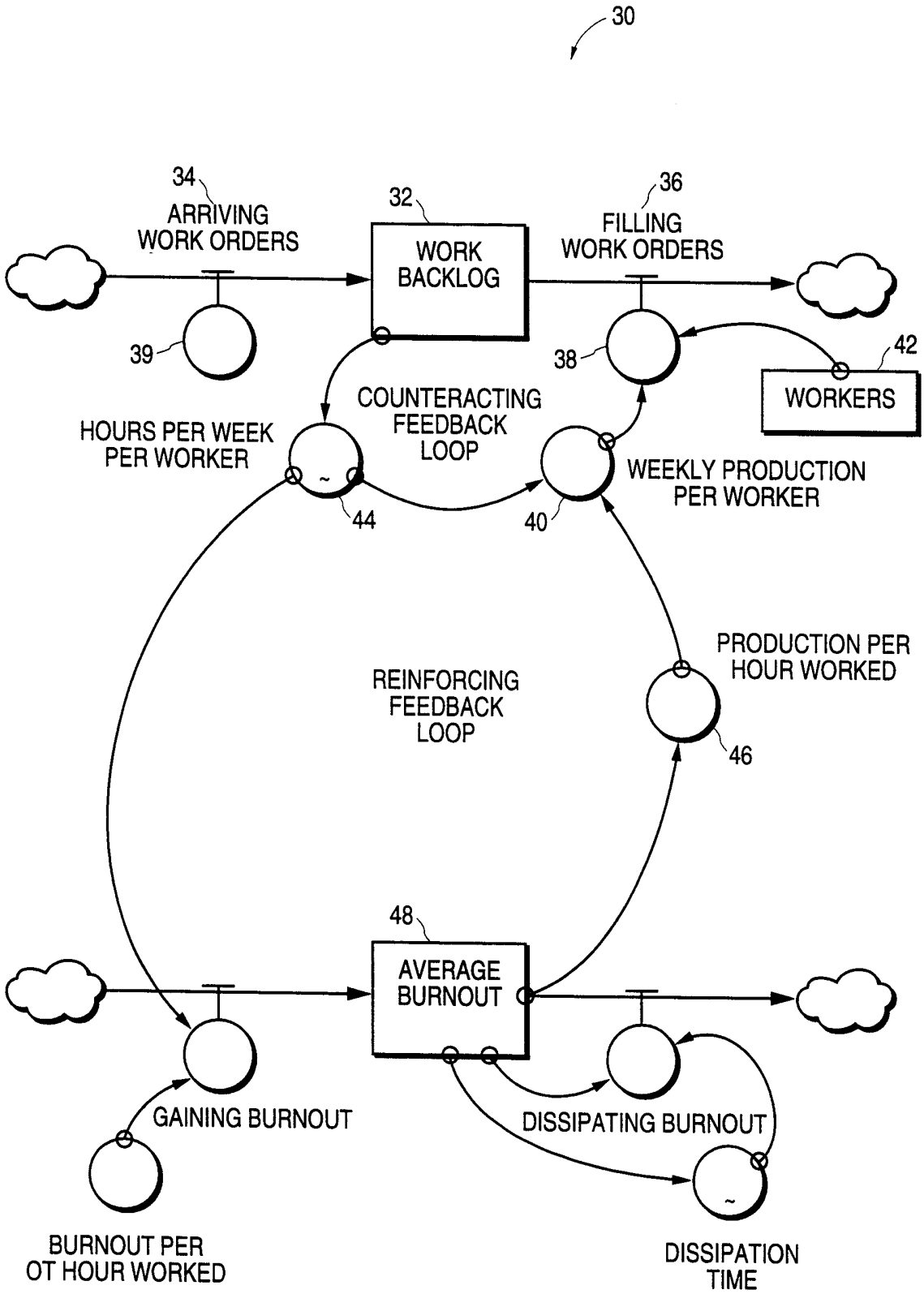


FIG.2 (PRIOR ART)

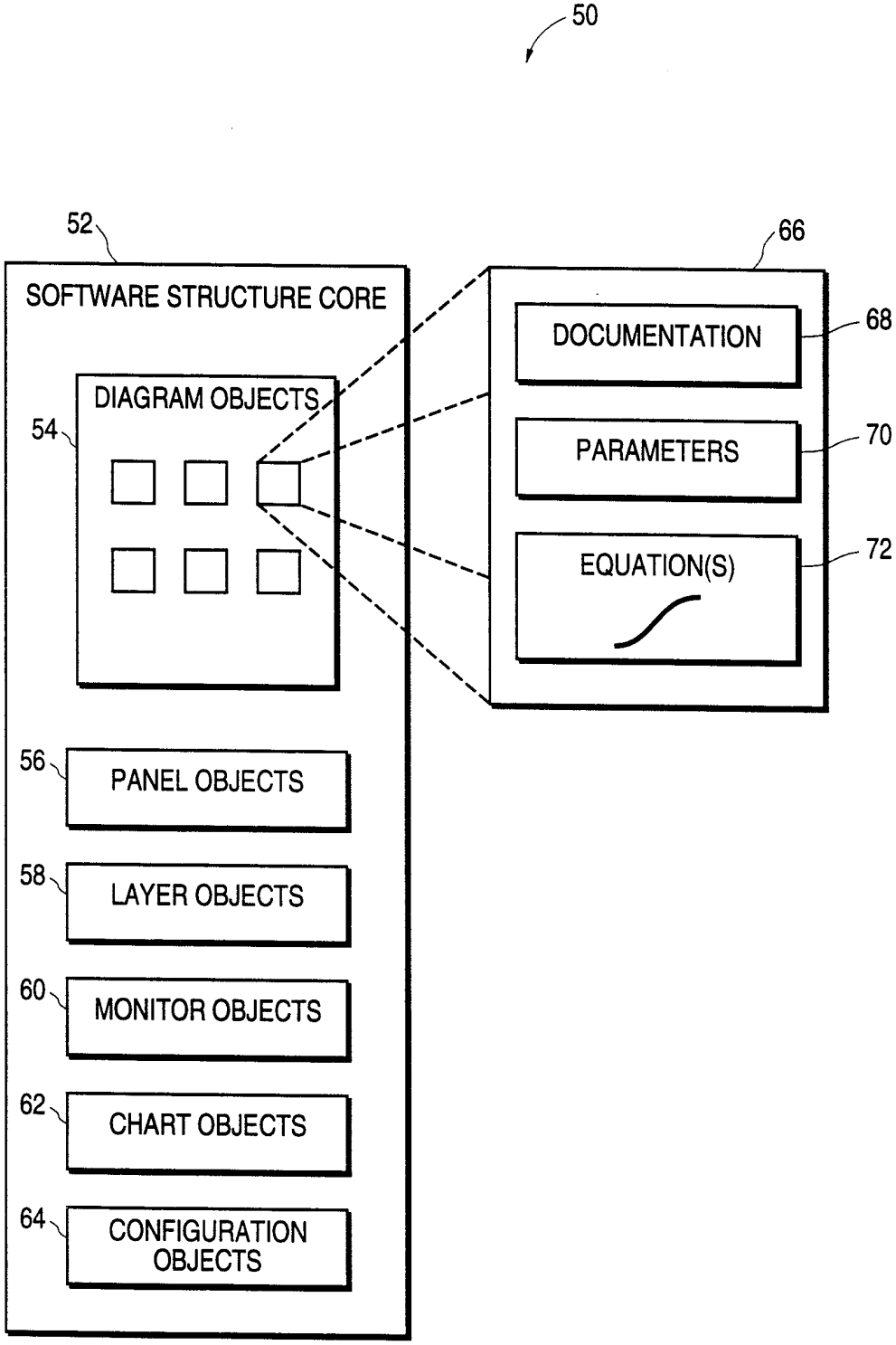

50
52
SOFTWARE STRUCTURE CORE
DIAGRAM OBJECTS
54
56
PANEL OBJECTS
58
LAYER OBJECTS
60
MONITOR OBJECTS
62
CHART OBJECTS
64
CONFIGURATION OBJECTS
66
DOCUMENTATION
68
PARAMETERS
70
EQUATION(S)
72

FIG.3

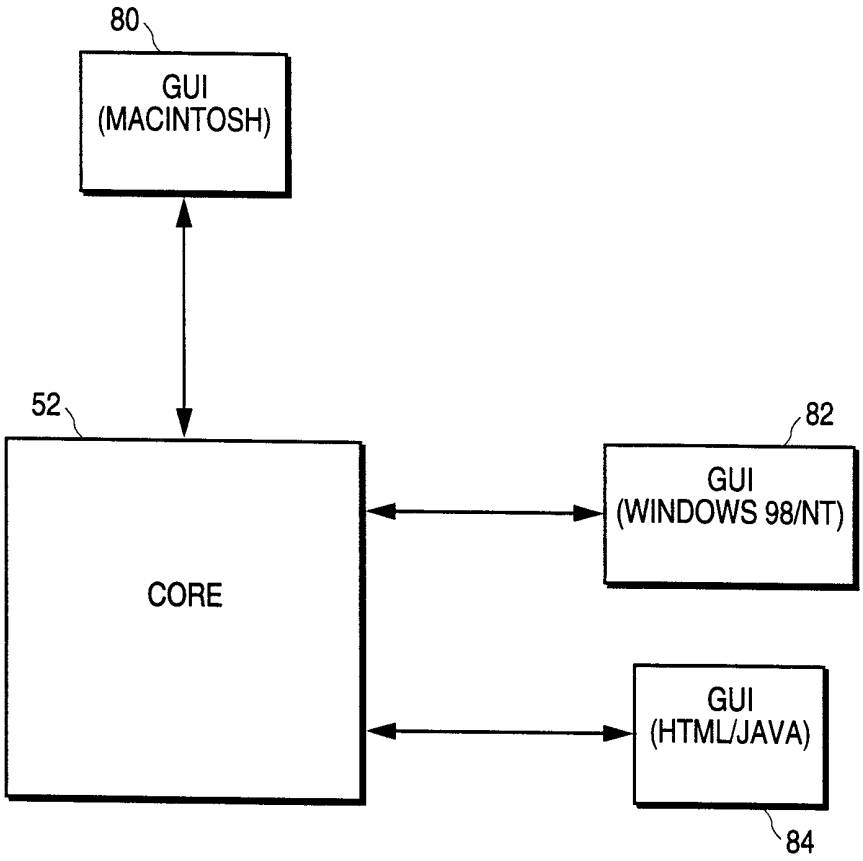


FIG.4

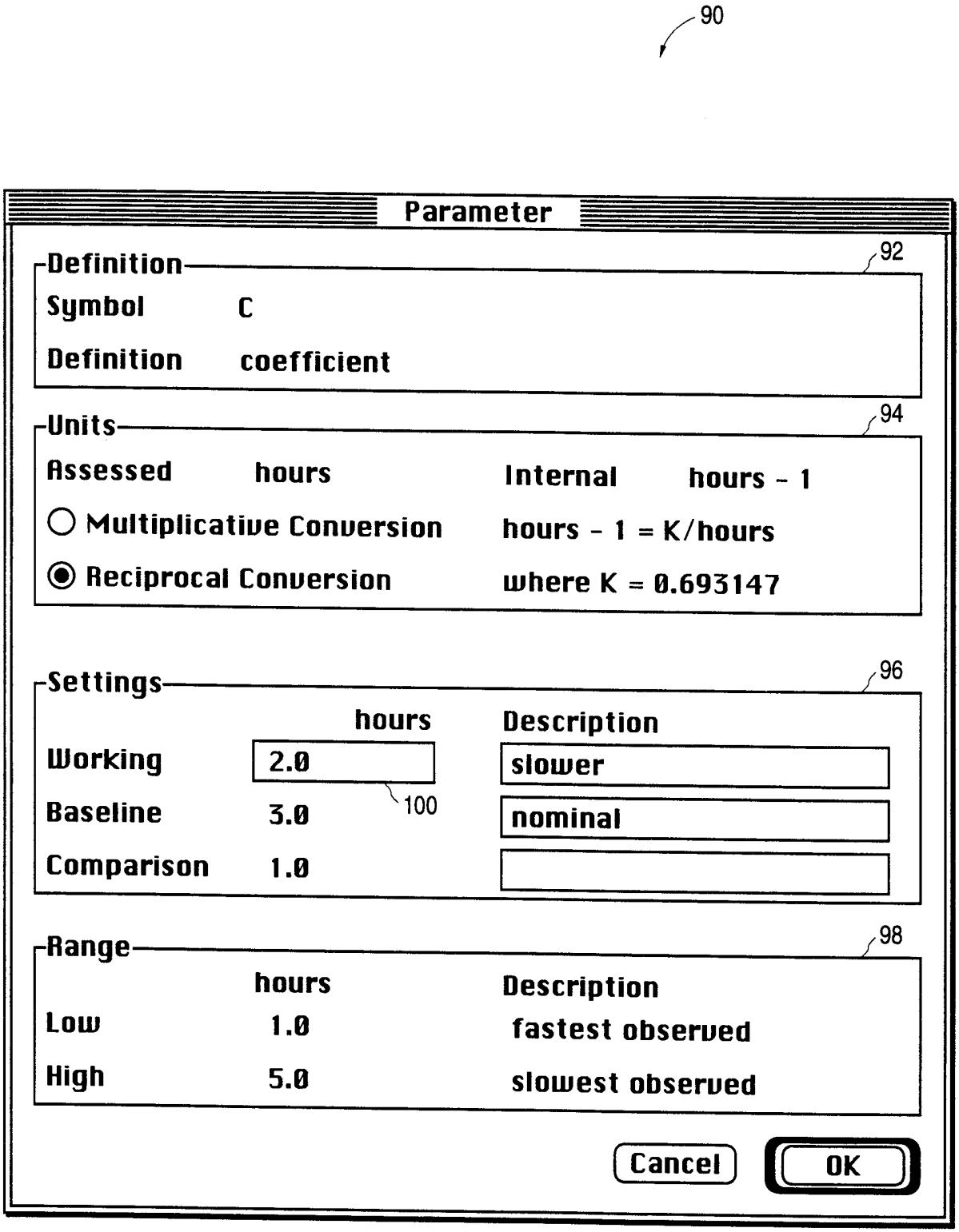

90
Parameter
Definition
92
Symbol C
Definition coefficient
Units
94
Assessed hours
Internal hours - 1
Multiplicative Conversion
hours - 1 = K/hours
Reciprocal Conversion
where K = 0.693147
Settings
96
hours
Description
Working 2.0
slower
100
Baseline 3.0
nominal
Comparison 1.0
Range
98
hours
Description
Low 1.0
fastest observed
High 5.0
slowest observed
Cancel
OK

FIG.5

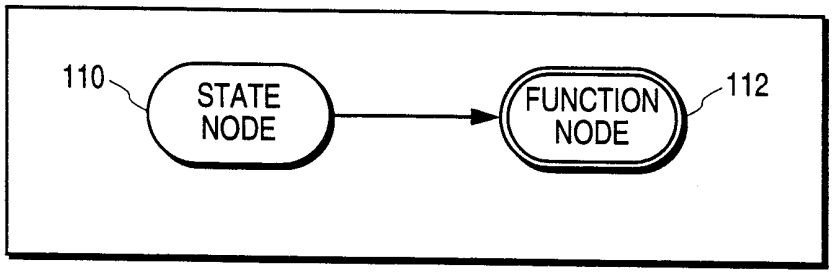


FIG.6

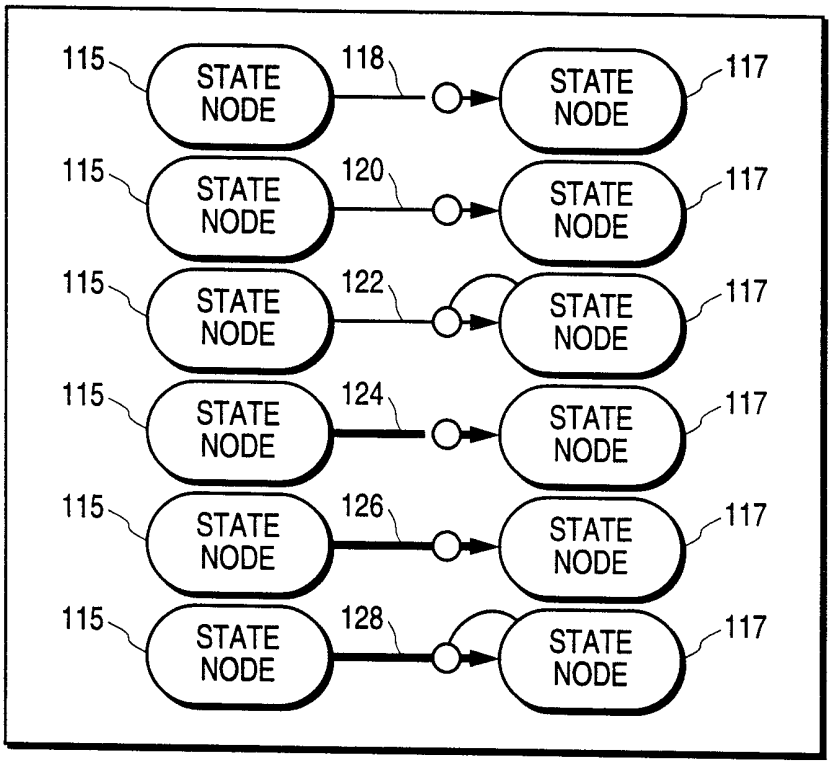


FIG.7

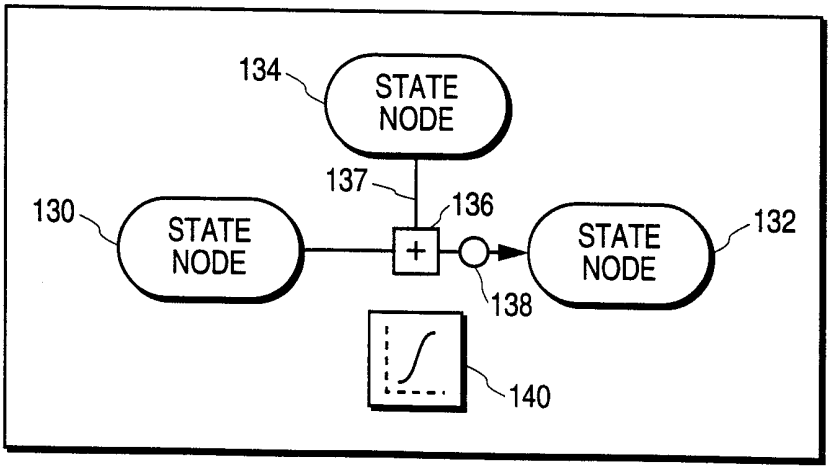


FIG.8

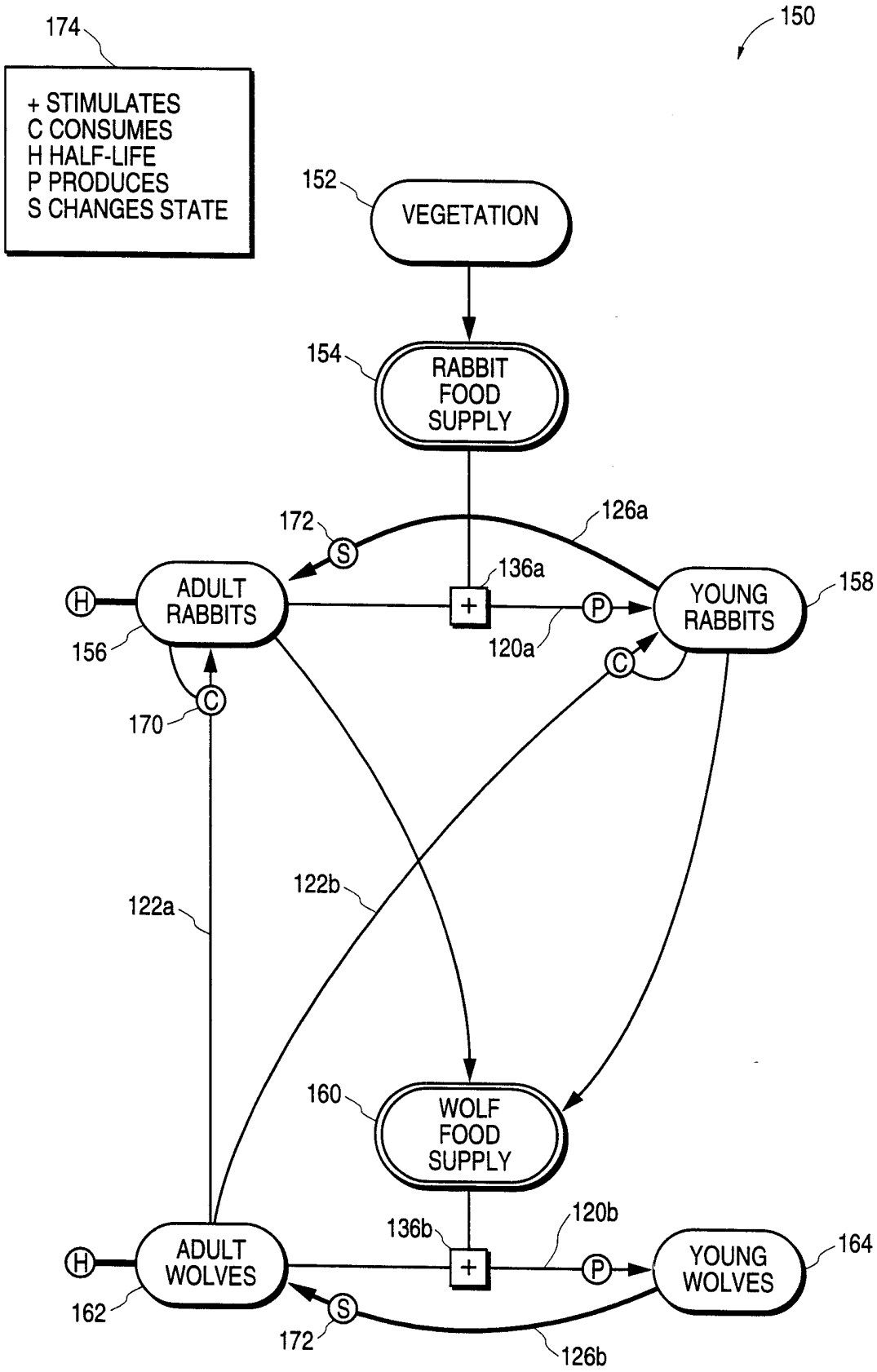

174
+ STIMULATES
C CONSUMES
H HALF-LIFE
P PRODUCES
S CHANGES STATE
150
152
VEGETATION
154
RABBIT FOOD SUPPLY
172
126a
136a
158
ADULT RABBITS
YOUNG RABBITS
156
120a
170
122b
122a
160
WOLF FOOD SUPPLY
136b
120b
164
ADULT WOLVES
YOUNG WOLVES
162
172
126b

FIG.9

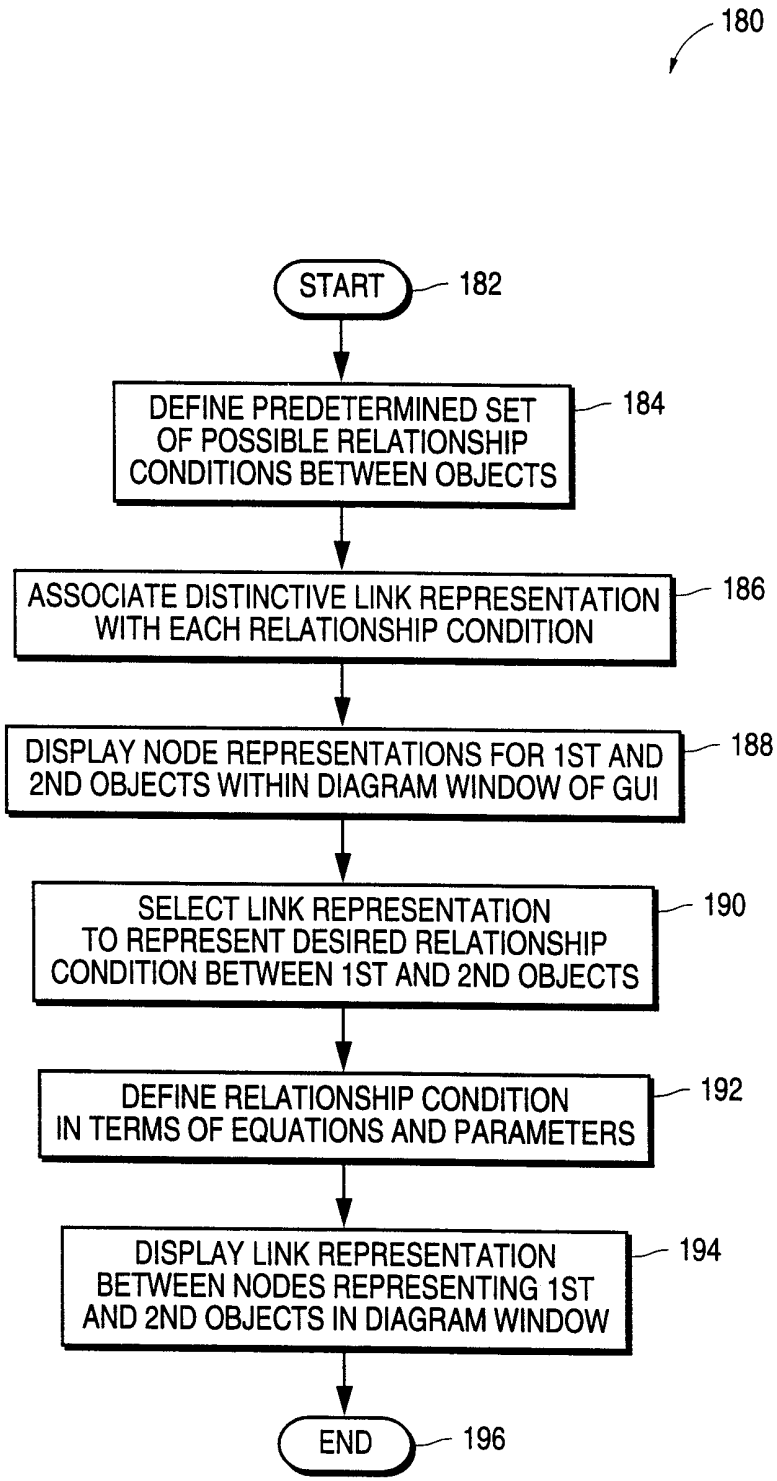

180
START
182
DEFINE PREDETERMINED SET OF POSSIBLE RELATIONSHIP CONDITIONS BETWEEN OBJECTS
184
ASSOCIATE DISTINCTIVE LINK REPRESENTATION WITH EACH RELATIONSHIP CONDITION
186
DISPLAY NODE REPRESENTATIONS FOR 1ST AND 2ND OBJECTS WITHIN DIAGRAM WINDOW OF GUI
188
SELECT LINK REPRESENTATION TO REPRESENT DESIRED RELATIONSHIP CONDITION BETWEEN 1ST AND 2ND OBJECTS
190
DEFINE RELATIONSHIP CONDITION IN TERMS OF EQUATIONS AND PARAMETERS
192
DISPLAY LINK REPRESENTATION BETWEEN NODES REPRESENTING 1ST AND 2ND OBJECTS IN DIAGRAM WINDOW
194
END
196

FIG.10

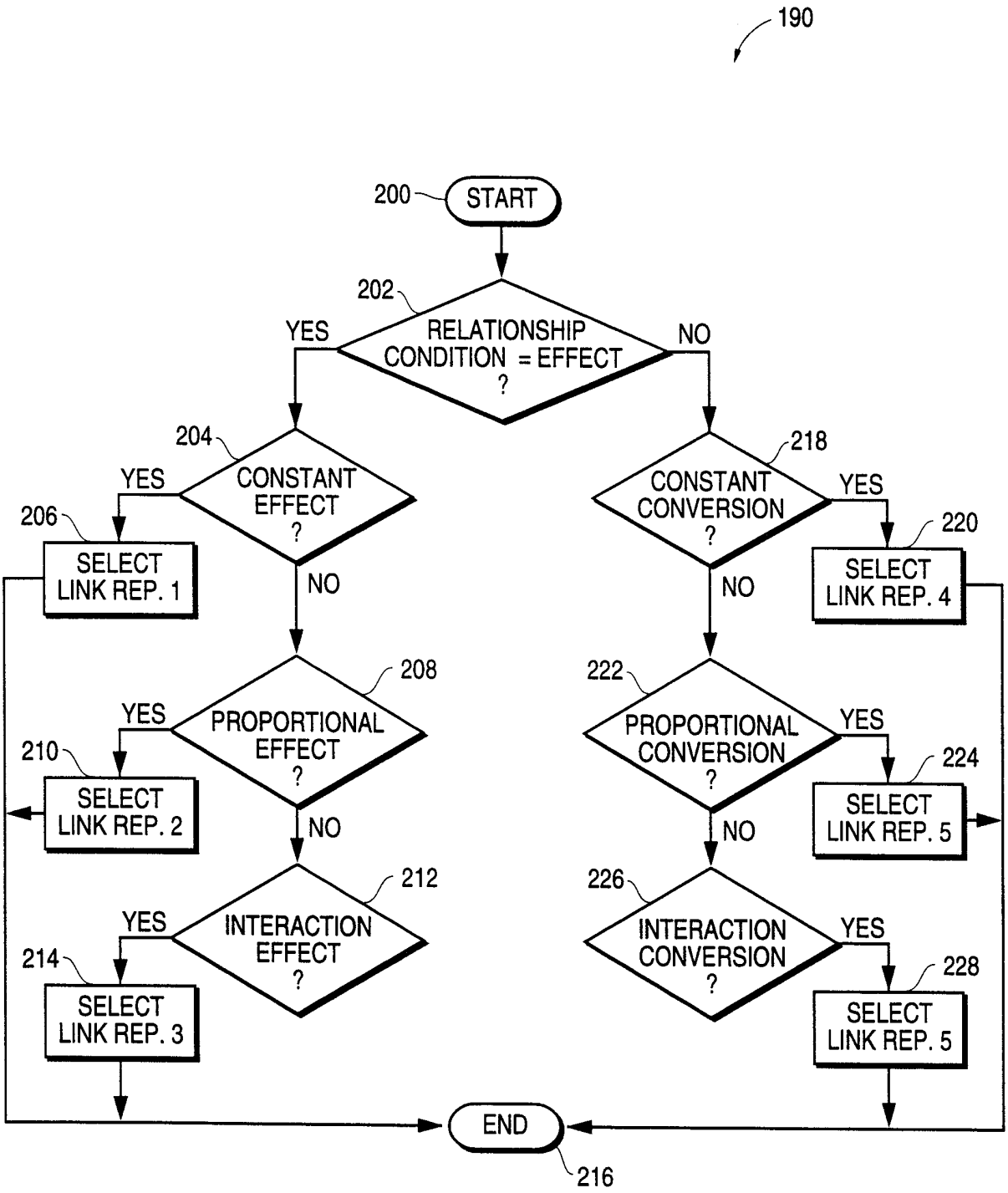

190
200
START
202
RELATIONSHIP CONDITION = EFFECT ?
YES
NO
204
CONSTANT EFFECT ?
YES
206
SELECT LINK REP. 1
NO
208
PROPORTIONAL EFFECT ?
YES
210
SELECT LINK REP. 2
NO
212
INTERACTION EFFECT ?
YES
214
SELECT LINK REP. 3
218
CONSTANT CONVERSION ?
YES
220
SELECT LINK REP. 4
NO
222
PROPORTIONAL CONVERSION ?
YES
224
SELECT LINK REP. 5
NO
226
INTERACTION CONVERSION ?
YES
228
SELECT LINK REP. 5
END
216

FIG.11

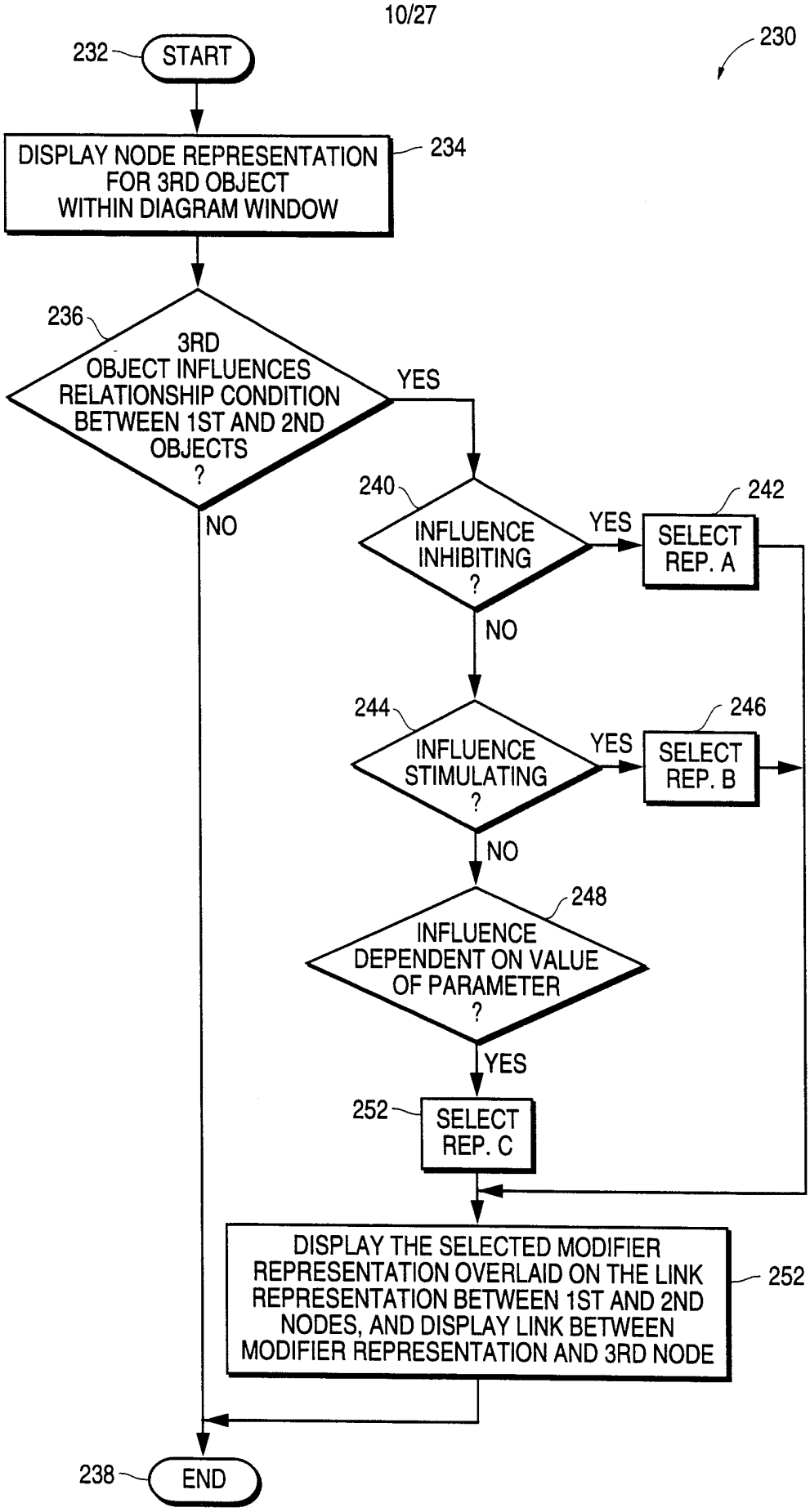

230
232
START
DISPLAY NODE REPRESENTATION FOR 3RD OBJECT WITHIN DIAGRAM WINDOW
234
236
3RD OBJECT INFLUENCES RELATIONSHIP CONDITION BETWEEN 1ST AND 2ND OBJECTS ?
YES
NO
240
INFLUENCE INHIBITING ?
YES
242
SELECT REP. A
NO
244
INFLUENCE STIMULATING ?
YES
246
SELECT REP. B
NO
248
INFLUENCE DEPENDENT ON VALUE OF PARAMETER ?
YES
252
SELECT REP. C
DISPLAY THE SELECTED MODIFIER REPRESENTATION OVERLAID ON THE LINK REPRESENTATION BETWEEN 1ST AND 2ND NODES, AND DISPLAY LINK BETWEEN MODIFIER REPRESENTATION AND 3RD NODE
252
238
END

FIG.12

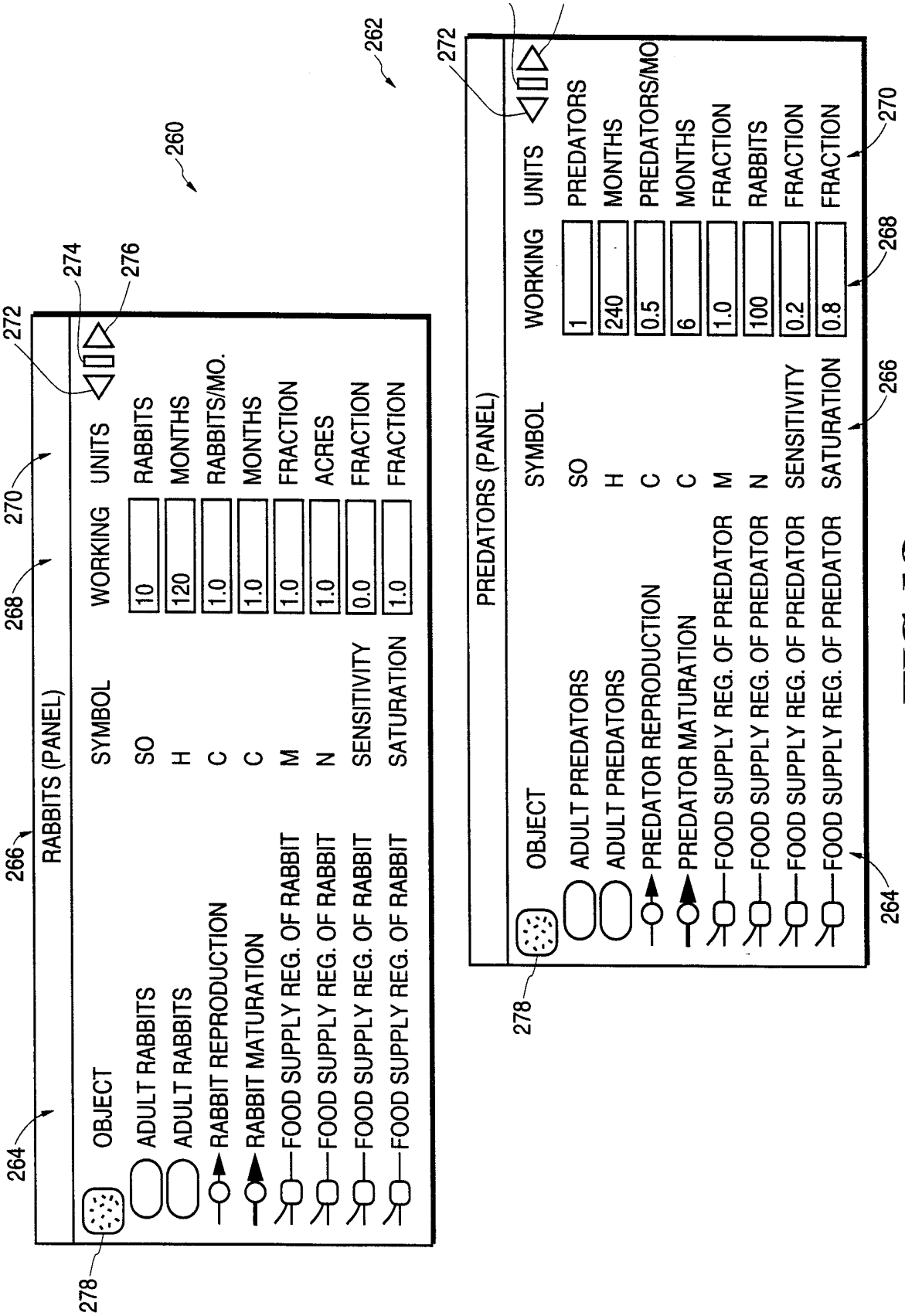

RABBITS (PANEL)
OBJECT
SYMBOL
WORKING
UNITS
ADULT RABBITS
SO
10
RABBITS
ADULT RABBITS
H
120
MONTHS
RABBIT REPRODUCTION
C
1.0
RABBITS/MO.
RABBIT MATURATION
C
1.0
MONTHS
FOOD SUPPLY REG. OF RABBIT
M
1.0
FRACTION
FOOD SUPPLY REG. OF RABBIT
N
1.0
ACRES
FOOD SUPPLY REG. OF RABBIT
SENSITIVITY
0.0
FRACTION
FOOD SUPPLY REG. OF RABBIT
SATURATION
1.0
FRACTION
PREDATORS (PANEL)
OBJECT
SYMBOL
WORKING
UNITS
ADULT PREDATORS
SO
1
PREDATORS
ADULT PREDATORS
H
240
MONTHS
PREDATOR REPRODUCTION
C
0.5
PREDATORS/MO
PREDATOR MATURATION
C
6
MONTHS
FOOD SUPPLY REG. OF PREDATOR
M
1.0
FRACTION
FOOD SUPPLY REG. OF PREDATOR
N
100
RABBITS
FOOD SUPPLY REG. OF PREDATOR
SENSITIVITY
0.2
FRACTION
FOOD SUPPLY REG. OF PREDATOR
SATURATION
0.8
FRACTION
260
262
264
266
268
270
272
274
276
278

FIG.13

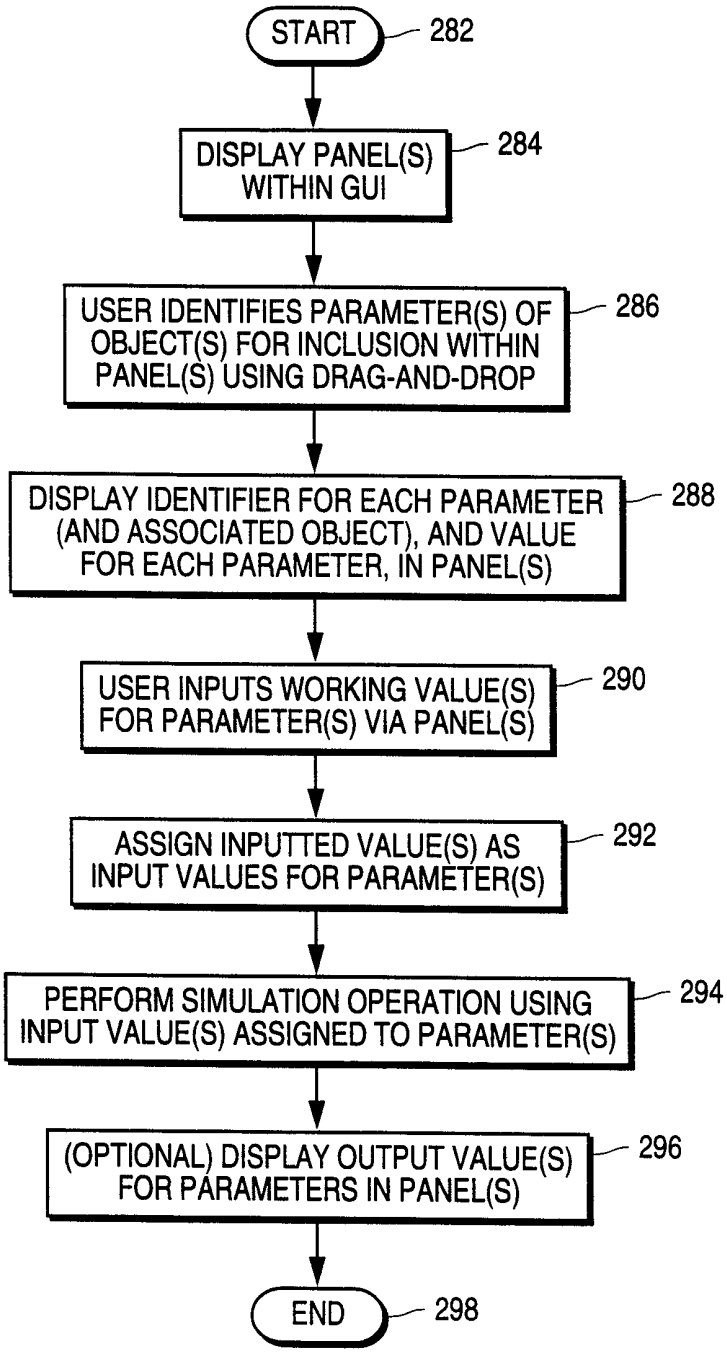

START
282
DISPLAY PANEL(S) WITHIN GUI
284
USER IDENTIFIES PARAMETER(S) OF OBJECT(S) FOR INCLUSION WITHIN PANEL(S) USING DRAG-AND-DROP
286
DISPLAY IDENTIFIER FOR EACH PARAMETER (AND ASSOCIATED OBJECT), AND VALUE FOR EACH PARAMETER, IN PANEL(S)
288
USER INPUTS WORKING VALUE(S) FOR PARAMETER(S) VIA PANEL(S)
290
ASSIGN INPUTTED VALUE(S) AS INPUT VALUES FOR PARAMETER(S)
292
PERFORM SIMULATION OPERATION USING INPUT VALUE(S) ASSIGNED TO PARAMETER(S)
294
(OPTIONAL) DISPLAY OUTPUT VALUE(S) FOR PARAMETERS IN PANEL(S)
296
END
298

FIG.14

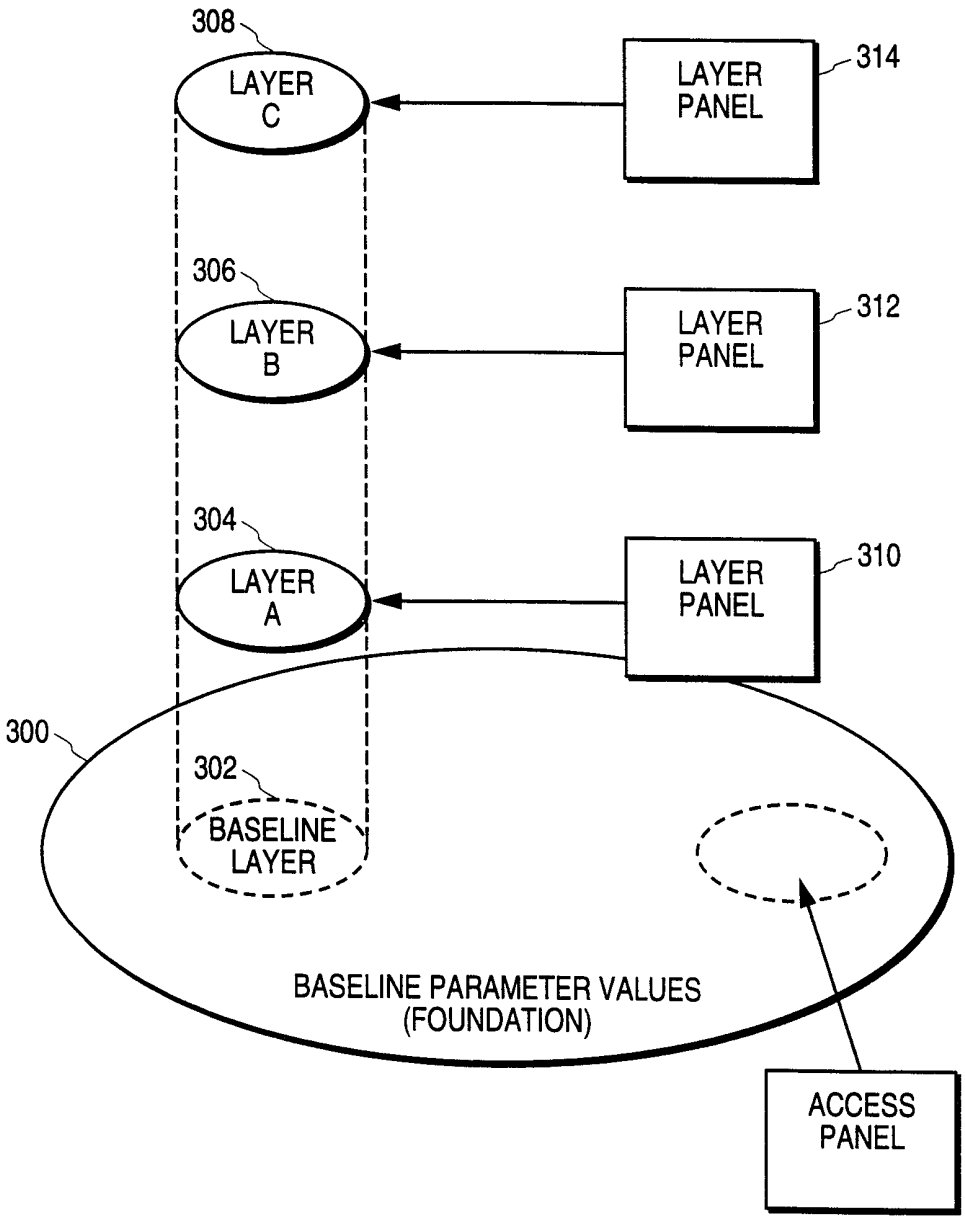

308
LAYER C
LAYER PANEL
314
306
LAYER B
LAYER PANEL
312
304
LAYER A
LAYER PANEL
310
300
302
BASELINE LAYER
BASELINE PARAMETER VALUES (FOUNDATION)
ACCESS PANEL

FIG.15

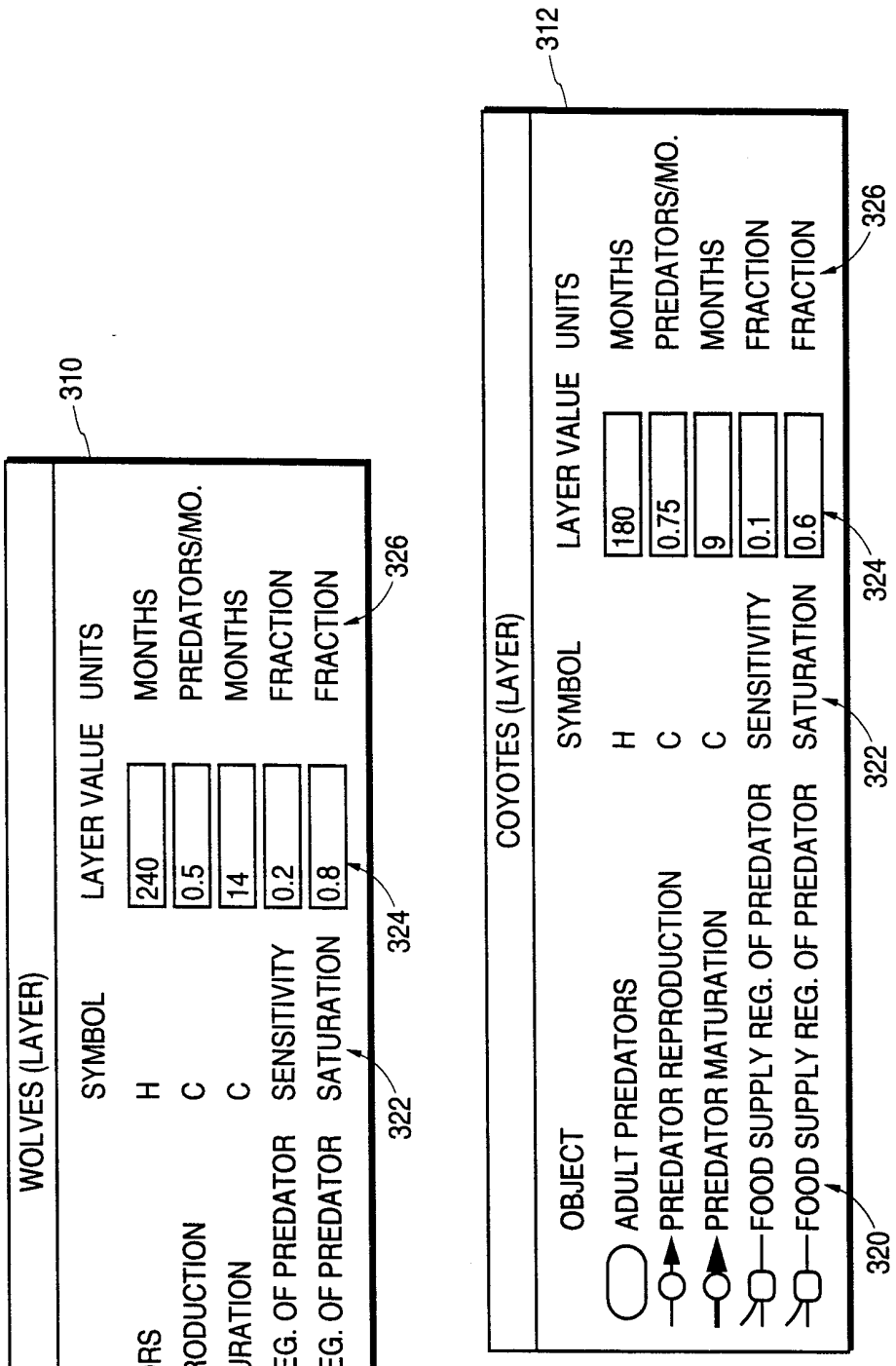

WOLVES (LAYER)
310
OBJECT
SYMBOL
LAYER VALUE
UNITS
ADULT PREDATORS
H
240
MONTHS
PREDATOR REPRODUCTION
C
0.5
PREDATORS/MO.
PREDATOR MATURATION
C
14
MONTHS
FOOD SUPPLY REG. OF PREDATOR
SENSITIVITY
0.2
FRACTION
FOOD SUPPLY REG. OF PREDATOR
SATURATION
0.8
FRACTION
320
322
324
326
COYOTES (LAYER)
312
OBJECT
SYMBOL
LAYER VALUE
UNITS
ADULT PREDATORS
H
180
MONTHS
PREDATOR REPRODUCTION
C
0.75
PREDATORS/MO.
PREDATOR MATURATION
C
9
MONTHS
FOOD SUPPLY REG. OF PREDATOR
SENSITIVITY
0.1
FRACTION
FOOD SUPPLY REG. OF PREDATOR
SATURATION
0.6
FRACTION
320
322
324
326

FIG.16

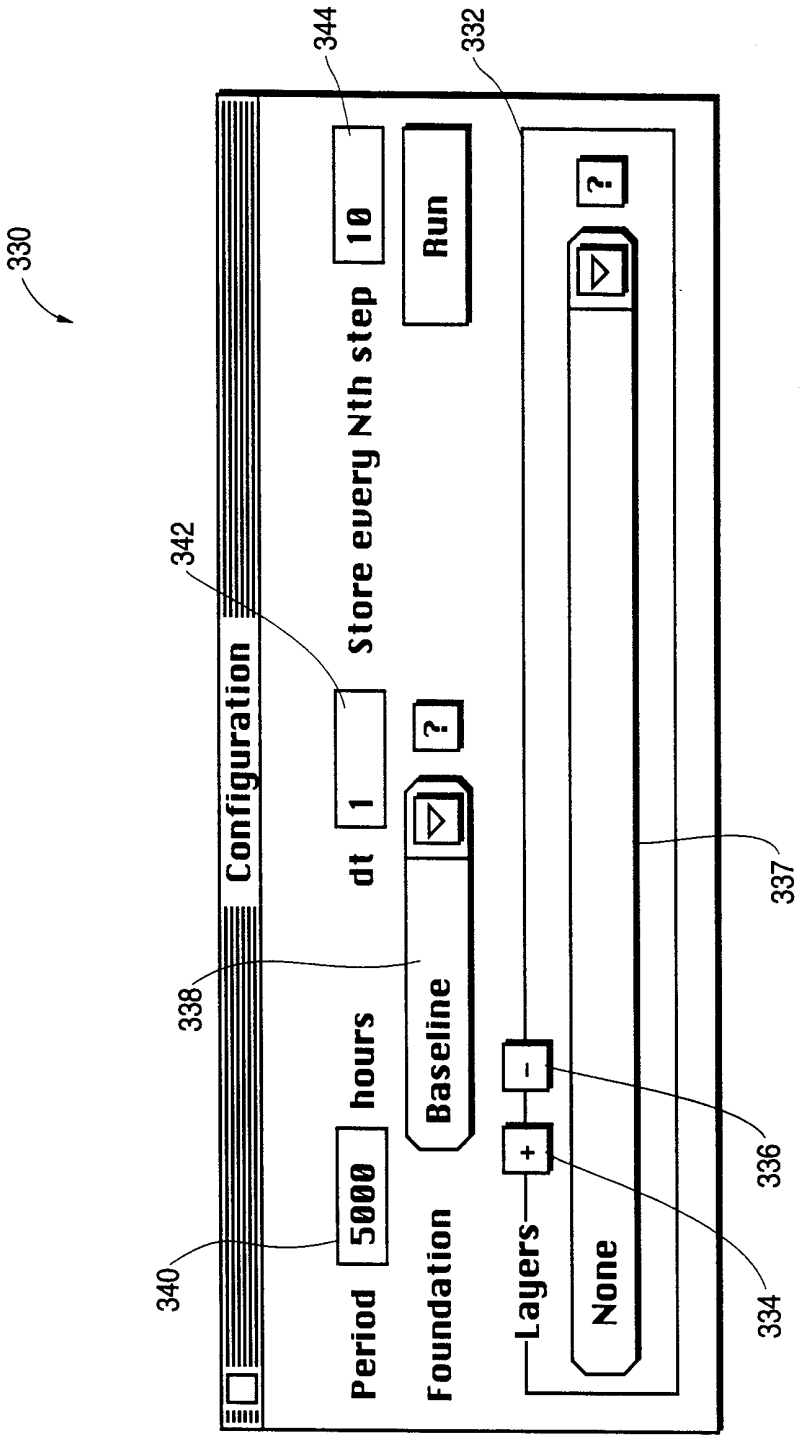

330
344
332
342
338
340
334
336
337
Configuration
Store every Nth step
10
Run
dt
1
?
Period
5000
hours
Baseline
Foundation
Layers
None
?
-
+

FIG.17

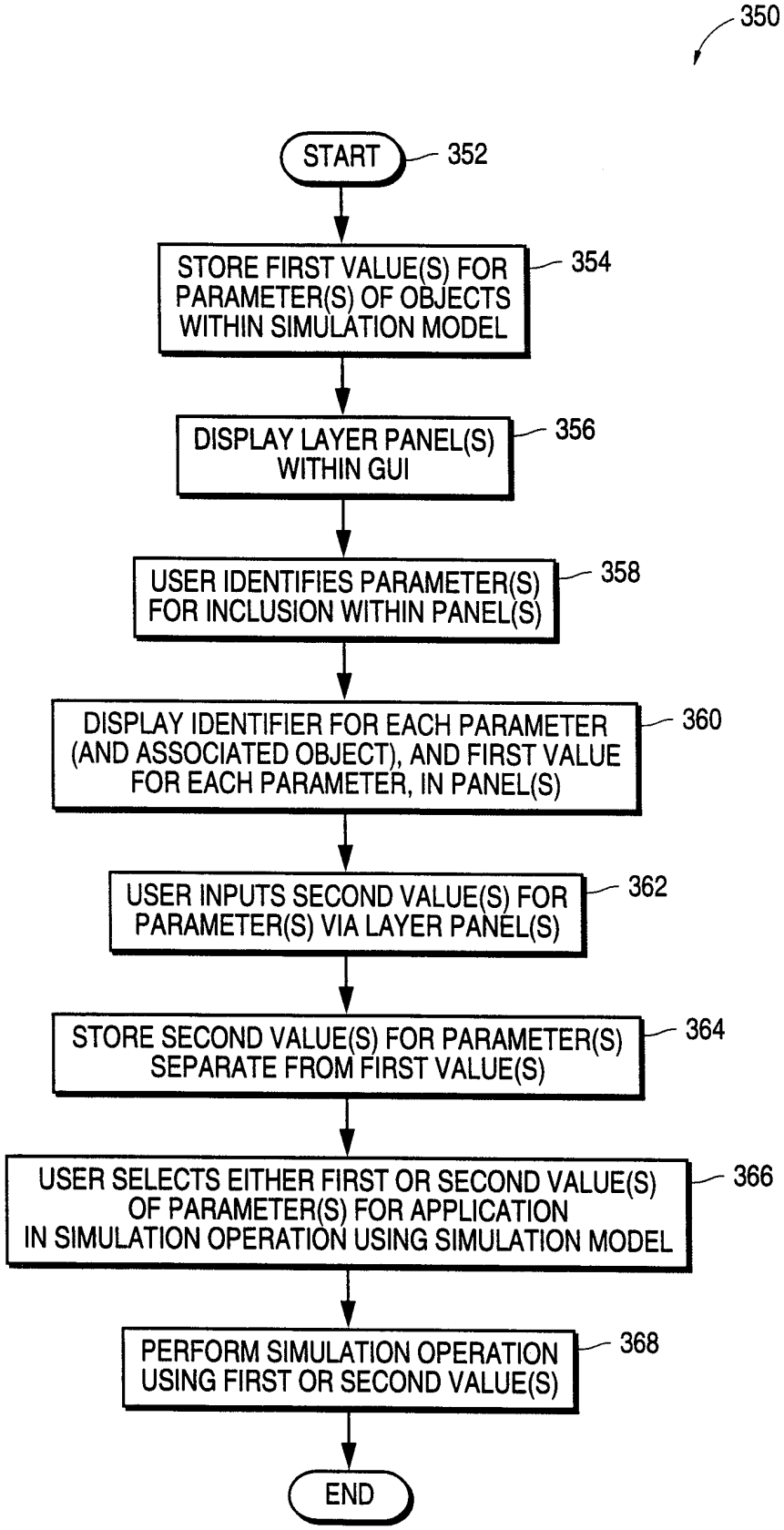

350
START
352
STORE FIRST VALUE(S) FOR PARAMETER(S) OF OBJECTS WITHIN SIMULATION MODEL
354
DISPLAY LAYER PANEL(S) WITHIN GUI
356
USER IDENTIFIES PARAMETER(S) FOR INCLUSION WITHIN PANEL(S)
358
DISPLAY IDENTIFIER FOR EACH PARAMETER (AND ASSOCIATED OBJECT), AND FIRST VALUE FOR EACH PARAMETER, IN PANEL(S)
360
USER INPUTS SECOND VALUE(S) FOR PARAMETER(S) VIA LAYER PANEL(S)
362
STORE SECOND VALUE(S) FOR PARAMETER(S) SEPARATE FROM FIRST VALUE(S)
364
USER SELECTS EITHER FIRST OR SECOND VALUE(S) OF PARAMETER(S) FOR APPLICATION IN SIMULATION OPERATION USING SIMULATION MODEL
366
PERFORM SIMULATION OPERATION USING FIRST OR SECOND VALUE(S)
368
END

FIG.18

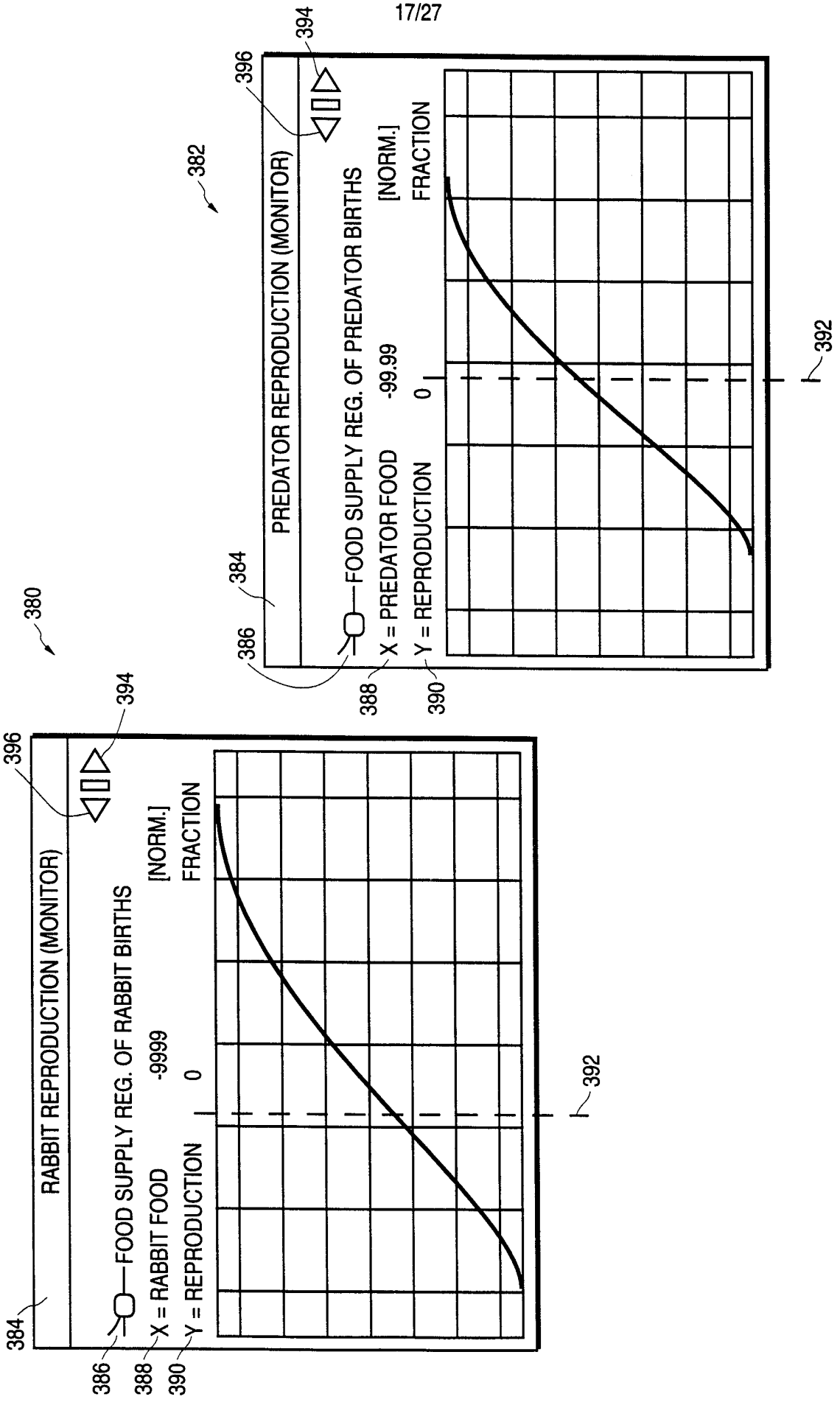

380
382
384
386
388
390
392
394
396
RABBIT REPRODUCTION (MONITOR)
FOOD SUPPLY REG. OF RABBIT BIRTHS
X = RABBIT FOOD
Y = REPRODUCTION
-9999
0
[NORM.]
FRACTION
PREDATOR REPRODUCTION (MONITOR)
FOOD SUPPLY REG. OF PREDATOR BIRTHS
X = PREDATOR FOOD
Y = REPRODUCTION
-99.99
0
[NORM.]
FRACTION

FIG.19

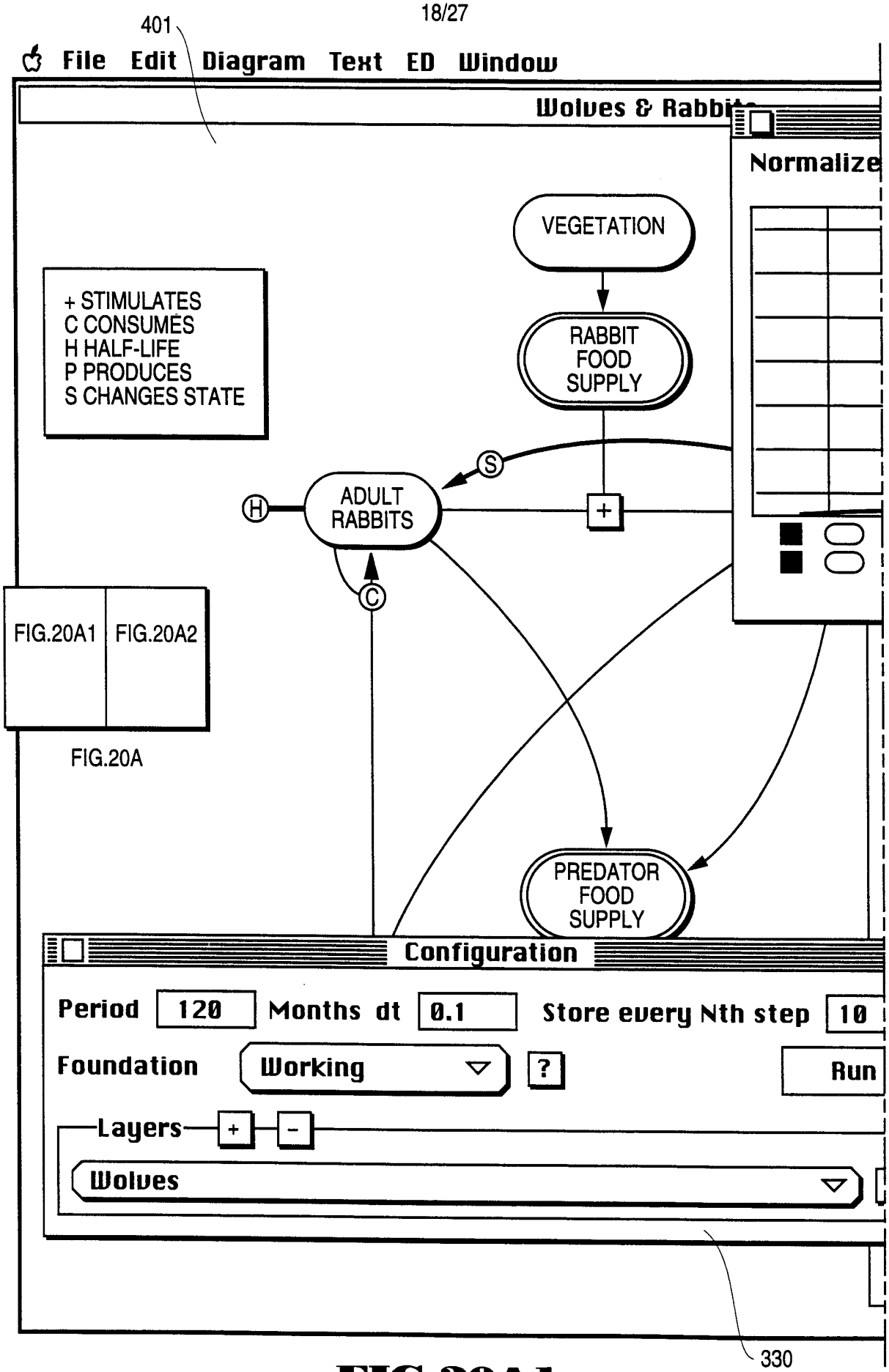

401
File Edit Diagram Text ED Window
Wolves & Rabbits
Normalize
VEGETATION
RABBIT FOOD SUPPLY
+ STIMULATES
C CONSUMES
H HALF-LIFE
P PRODUCES
S CHANGES STATE
S
H
ADULT RABBITS
+
C
FIG.20A1
FIG.20A2
FIG.20A
PREDATOR FOOD SUPPLY
Configuration
Period 120 Months dt 0.1 Store every Nth step 10
Foundation Working ?
Run
Layers + -
Wolves
330

FIG.20A1

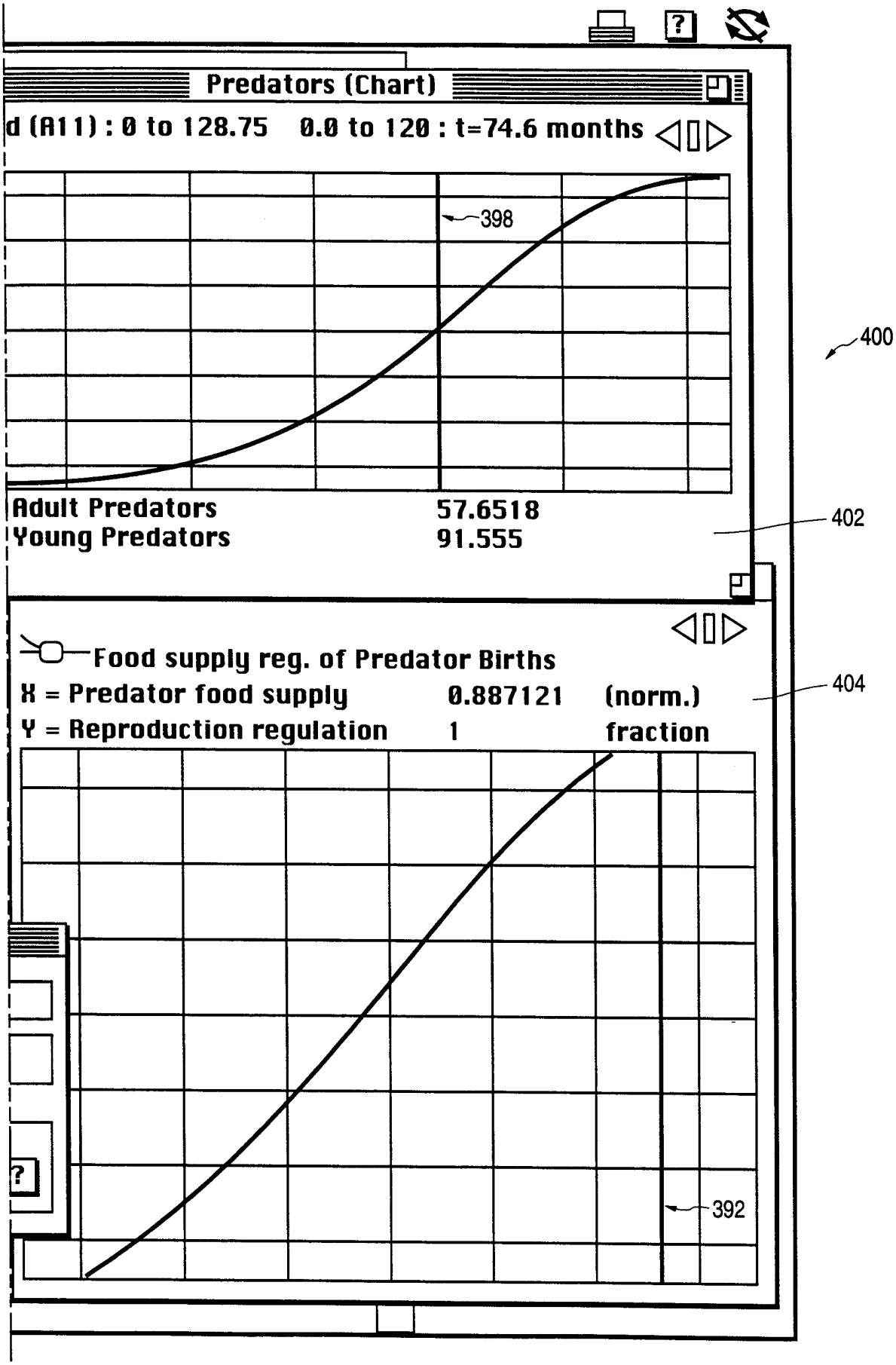

Predators (Chart)
d (A11) : 0 to 128.75 0.0 to 120 : t=74.6 months
398
400
Adult Predators 57.6518
Young Predators 91.555
402
Food supply reg. of Predator Births
X = Predator food supply 0.887121 (norm.)
Y = Reproduction regulation 1 fraction
404
392

FIG.20A2

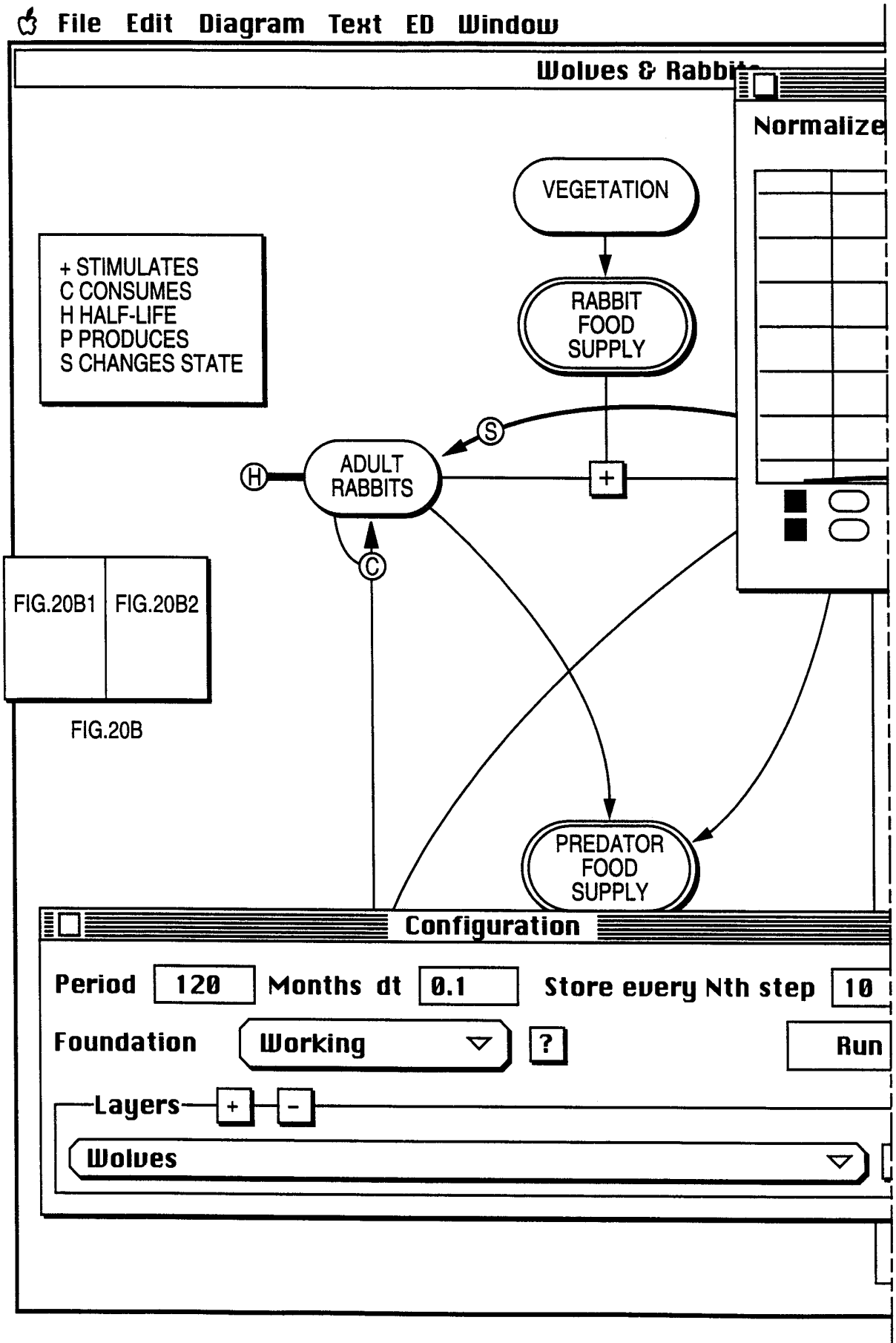

File Edit Diagram Text ED Window
Wolves & Rabbits
Normalize
VEGETATION
RABBIT FOOD SUPPLY
+ STIMULATES
C CONSUMES
H HALF-LIFE
P PRODUCES
S CHANGES STATE
S
H
ADULT RABBITS
+
C
FIG.20B1
FIG.20B2
FIG.20B
PREDATOR FOOD SUPPLY
Configuration
Period 120 Months dt 0.1 Store every Nth step 10
Foundation Working ?
Run
Layers + -
Wolves

FIG.20B1

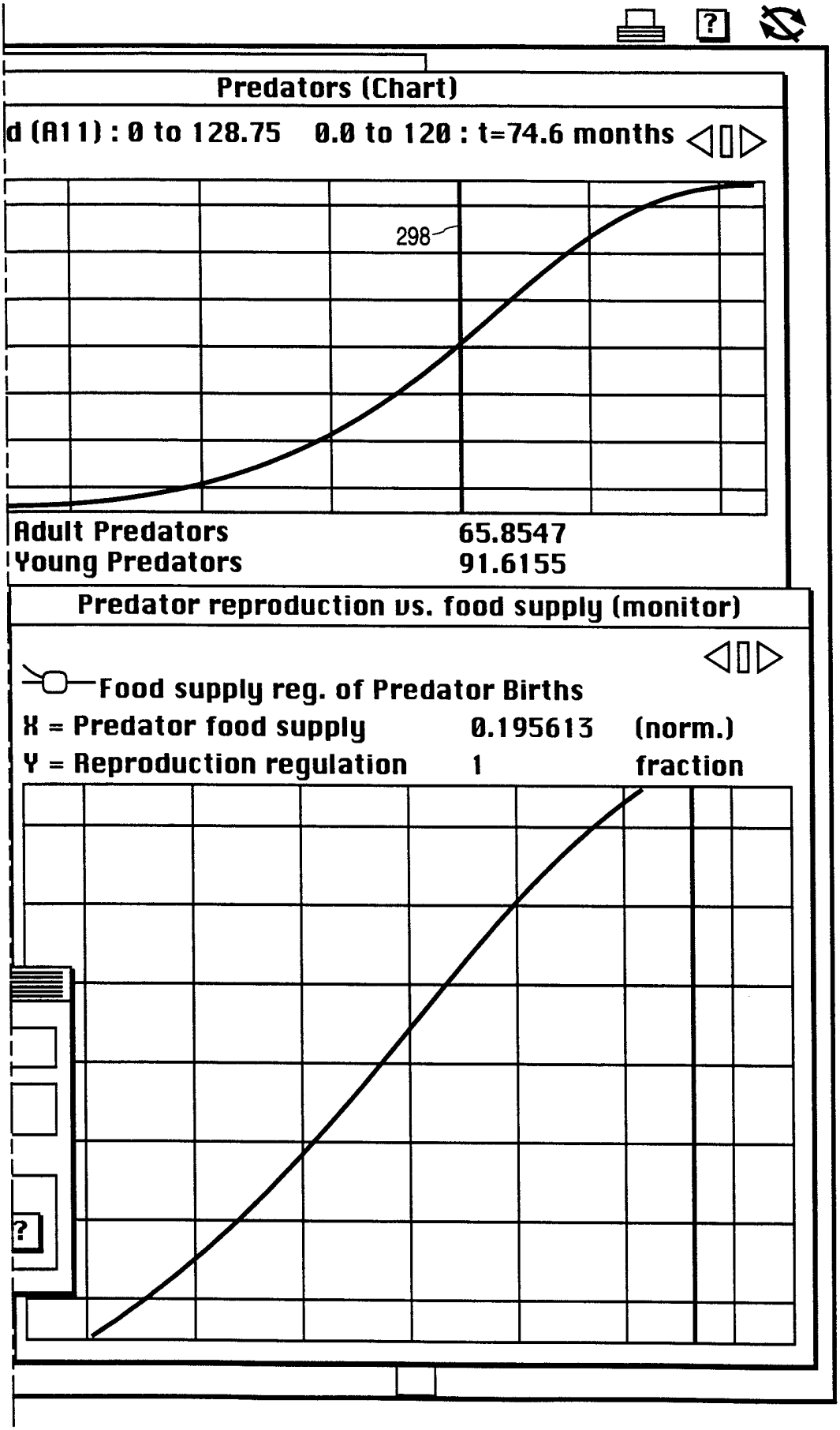

Predators (Chart)
d (A11) : 0 to 128.75 0.0 to 120 : t=74.6 months
298
Adult Predators 65.8547
Young Predators 91.6155
Predator reproduction vs. food supply (monitor)
Food supply reg. of Predator Births
X = Predator food supply 0.195613 (norm.)
Y = Reproduction regulation 1 fraction

FIG.20B2

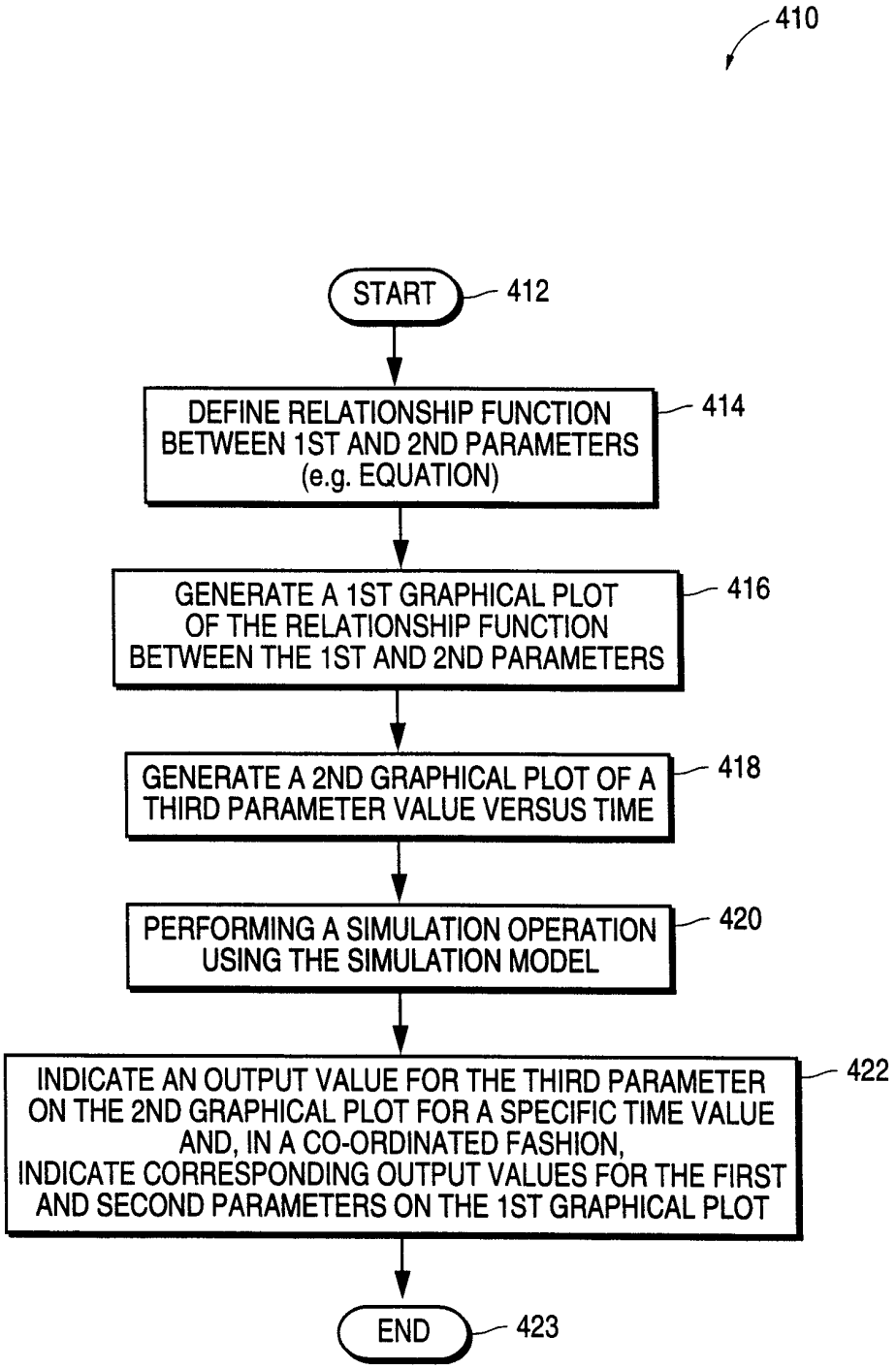

410
START
412
DEFINE RELATIONSHIP FUNCTION BETWEEN 1ST AND 2ND PARAMETERS (e.g. EQUATION)
414
GENERATE A 1ST GRAPHICAL PLOT OF THE RELATIONSHIP FUNCTION BETWEEN THE 1ST AND 2ND PARAMETERS
416
GENERATE A 2ND GRAPHICAL PLOT OF A THIRD PARAMETER VALUE VERSUS TIME
418
PERFORMING A SIMULATION OPERATION USING THE SIMULATION MODEL
420
INDICATE AN OUTPUT VALUE FOR THE THIRD PARAMETER ON THE 2ND GRAPHICAL PLOT FOR A SPECIFIC TIME VALUE AND, IN A CO-ORDINATED FASHION, INDICATE CORRESPONDING OUTPUT VALUES FOR THE FIRST AND SECOND PARAMETERS ON THE 1ST GRAPHICAL PLOT
422
END
423

FIG.21A

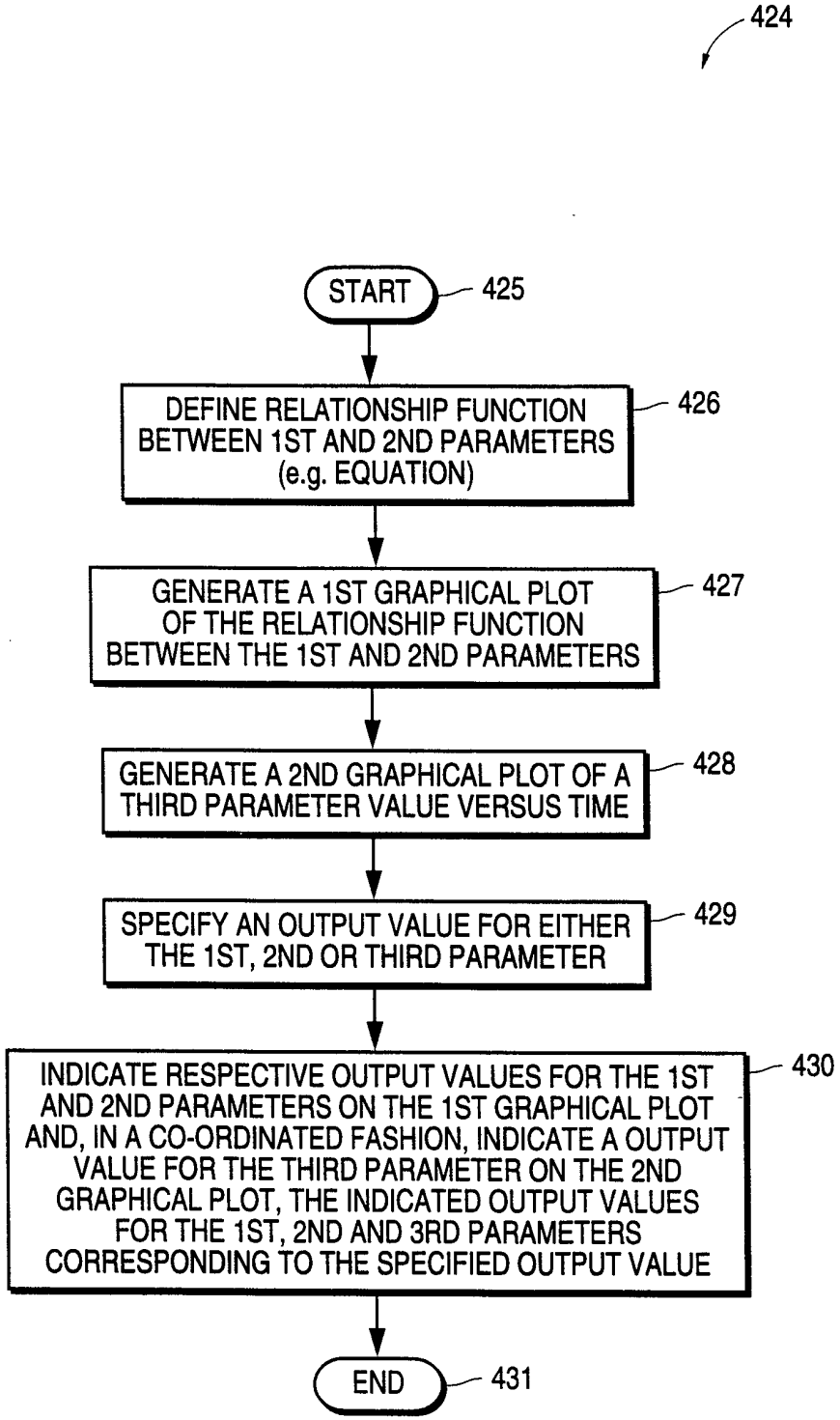

424
START
425
DEFINE RELATIONSHIP FUNCTION BETWEEN 1ST AND 2ND PARAMETERS (e.g. EQUATION)
426
GENERATE A 1ST GRAPHICAL PLOT OF THE RELATIONSHIP FUNCTION BETWEEN THE 1ST AND 2ND PARAMETERS
427
GENERATE A 2ND GRAPHICAL PLOT OF A THIRD PARAMETER VALUE VERSUS TIME
428
SPECIFY AN OUTPUT VALUE FOR EITHER THE 1ST, 2ND OR THIRD PARAMETER
429
INDICATE RESPECTIVE OUTPUT VALUES FOR THE 1ST AND 2ND PARAMETERS ON THE 1ST GRAPHICAL PLOT AND, IN A CO-ORDINATED FASHION, INDICATE A OUTPUT VALUE FOR THE THIRD PARAMETER ON THE 2ND GRAPHICAL PLOT, THE INDICATED OUTPUT VALUES FOR THE 1ST, 2ND AND 3RD PARAMETERS CORRESPONDING TO THE SPECIFIED OUTPUT VALUE
430
END
431

FIG.21B

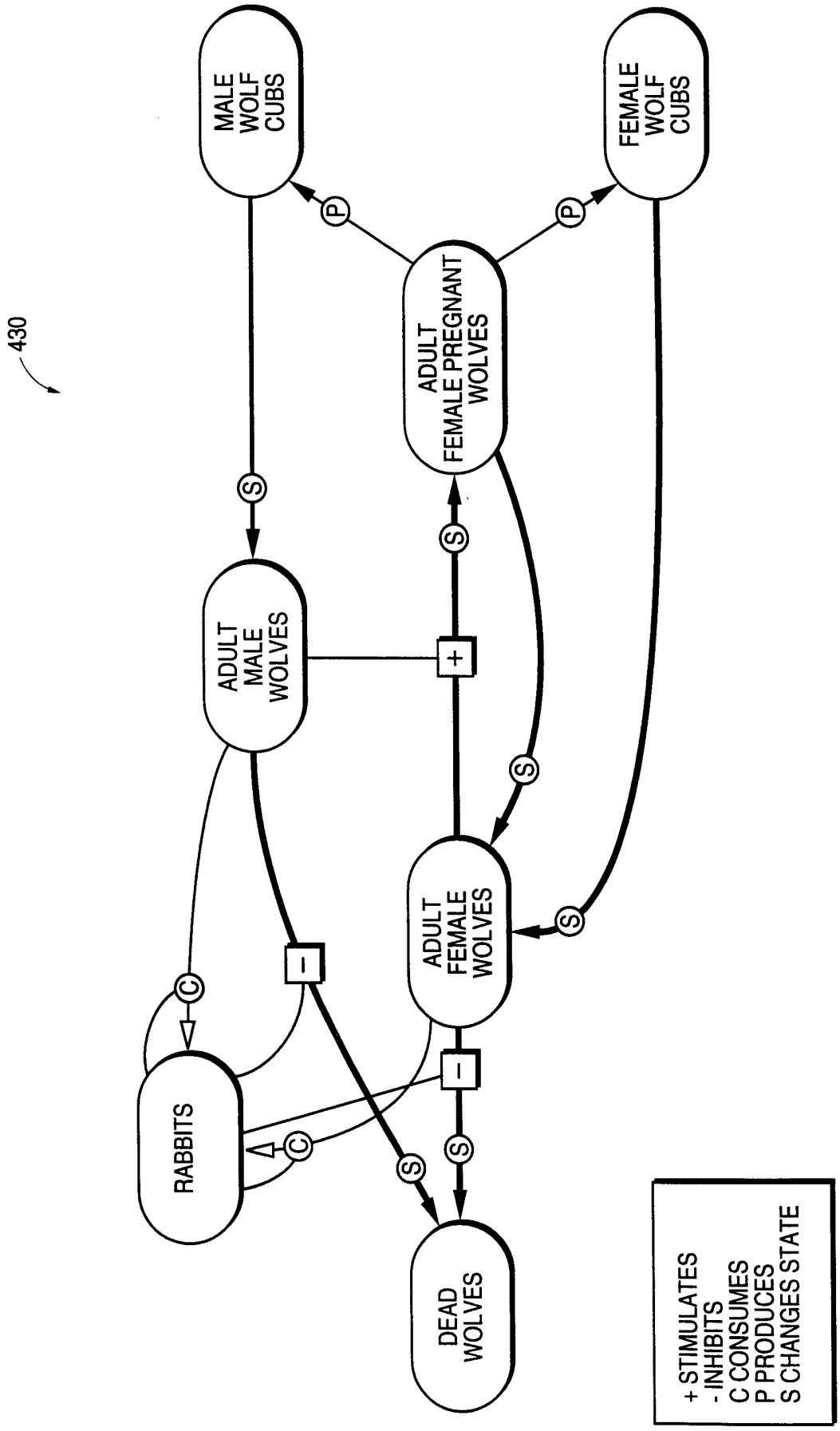

430
MALE WOLF CUBS
FEMALE WOLF CUBS
ADULT FEMALE PREGNANT WOLVES
ADULT MALE WOLVES
ADULT FEMALE WOLVES
RABBITS
DEAD WOLVES
P
S
C
+
-
+ STIMULATES
- INHIBITS
C CONSUMES
P PRODUCES
S CHANGES STATE

FIG.22

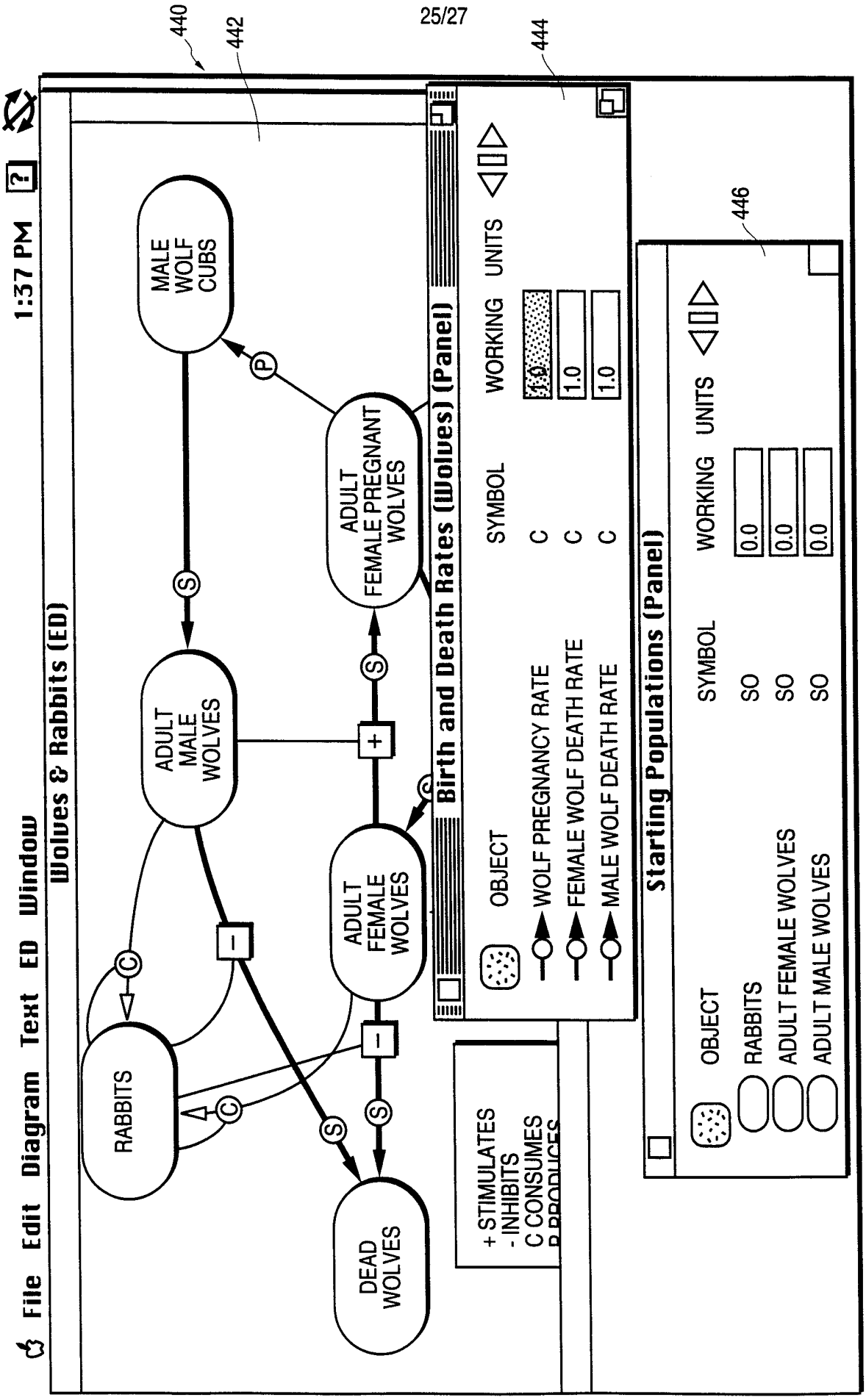

440
442
444
446
File Edit Diagram Text ED Window
1:37 PM
Wolves & Rabbits (ED)
MALE WOLF CUBS
ADULT MALE WOLVES
ADULT FEMALE PREGNANT WOLVES
ADULT FEMALE WOLVES
RABBITS
DEAD WOLVES
+ STIMULATES
- INHIBITS
C CONSUMES
P PRODUCES
Birth and Death Rates (Wolves) (Panel)
OBJECT
SYMBOL
WORKING UNITS
WOLF PREGNANCY RATE C 1.0
FEMALE WOLF DEATH RATE C 1.0
MALE WOLF DEATH RATE C 1.0
Starting Populations (Panel)
OBJECT
SYMBOL
WORKING UNITS
RABBITS SO 0.0
ADULT FEMALE WOLVES SO 0.0
ADULT MALE WOLVES SO 0.0

FIG.23

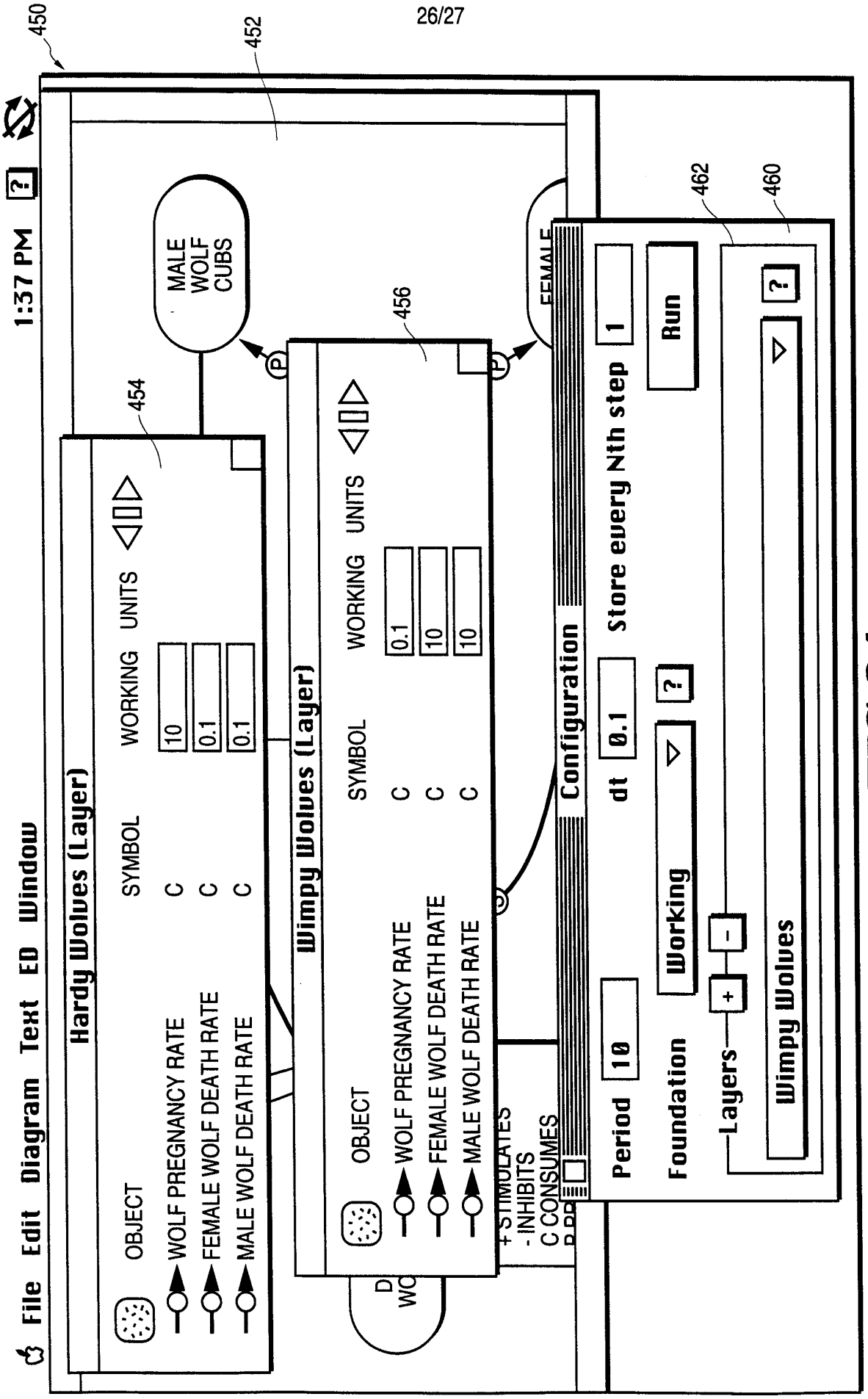

450
452
454
456
460
462
File Edit Diagram Text ED Window
1:37 PM
MALE WOLF CUBS
FEMALE
Hardy Wolves (Layer)
OBJECT
SYMBOL
WORKING UNITS
WOLF PREGNANCY RATE C 10
FEMALE WOLF DEATH RATE C 0.1
MALE WOLF DEATH RATE C 0.1
Wimpy Wolves (Layer)
OBJECT
SYMBOL
WORKING UNITS
WOLF PREGNANCY RATE C 0.1
FEMALE WOLF DEATH RATE C 10
MALE WOLF DEATH RATE C 10
+ STIMULATES
- INHIBITS
C CONSUMES
Configuration
Period 10
dt 0.1
Store every Nth step 1
Run
Foundation
Working
Layers
Wimpy Wolves

FIG.24

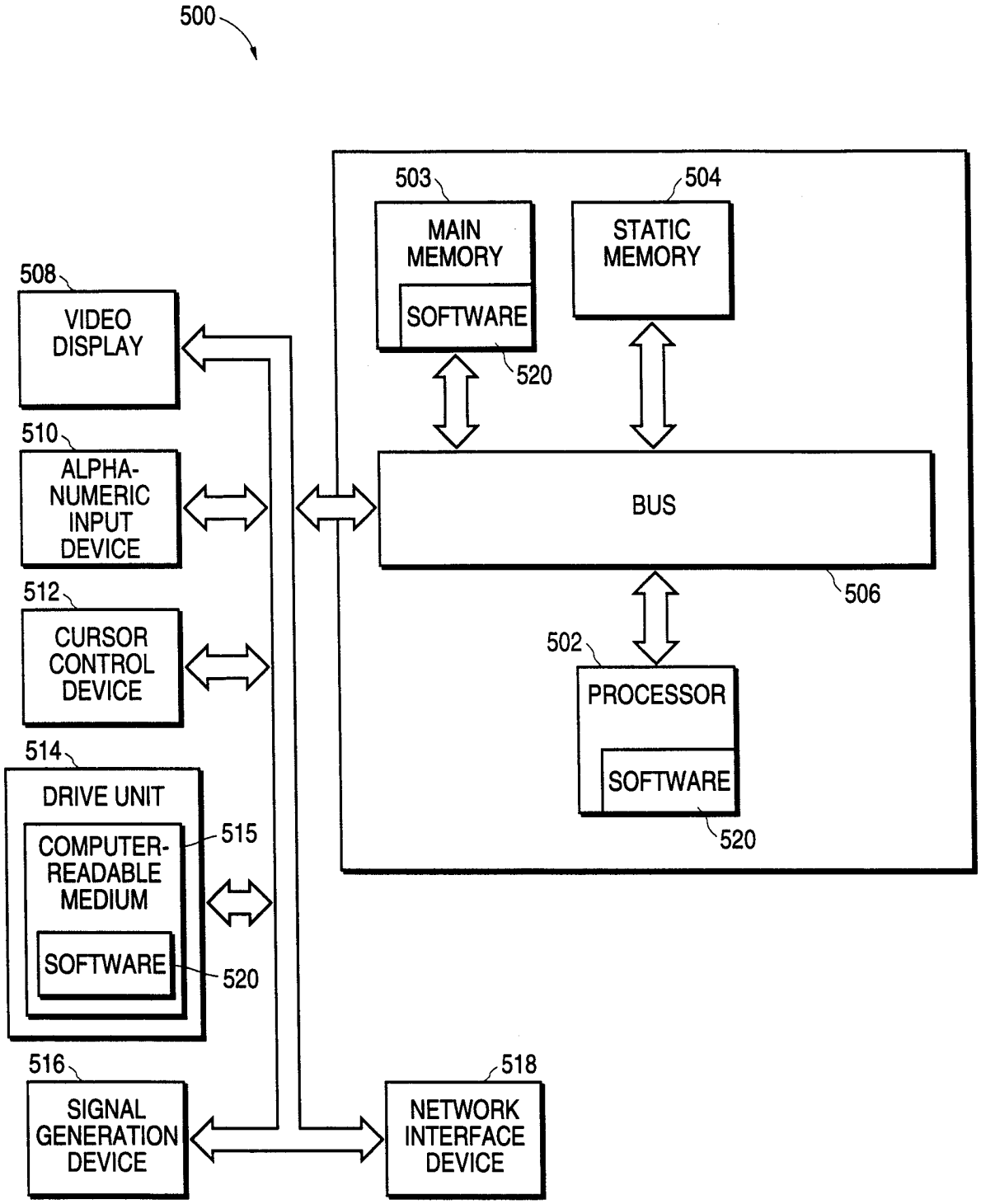

500
503
MAIN MEMORY
SOFTWARE
520
504
STATIC MEMORY
508
VIDEO DISPLAY
510
ALPHA-NUMERIC INPUT DEVICE
512
CURSOR CONTROL DEVICE
514
DRIVE UNIT
COMPUTER-READABLE MEDIUM
515
SOFTWARE
520
BUS
506
502
PROCESSOR
SOFTWARE
520
516
SIGNAL GENERATION DEVICE
518
NETWORK INTERFACE DEVICE

FIG.25

INTERNATIONAL SEARCH REPORT

Inte onal Application No
PCT/US 98/22592

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F9/44

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

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G06F

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

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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	LIN Y -B ET AL: "A FLEXIBLE GRAPHICAL USER INTERFACE FOR PERFORMANCE MODELING" SOFTWARE PRACTICE & EXPERIENCE, vol. 25, no. 2, February 1995, pages 193-216, XP000655499 see page 196, line 24 - page 202, line 23	1-16
A	EP 0 483 040 A (IBM) 29 April 1992 see page 2, line 19 - line 37 see figure 1 see figure 2	1-16
A	EP 0 587 519 A (IBM) 16 March 1994 see column 1, line 25 - line 47 see column 2, line 10 - line 23 see figure 2	1-16

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

☒ Patent family members are listed in annex.

° Special categories of cited documents :

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

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

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

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

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

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

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

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

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
8 March 1999	15/03/1999

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Brandt, J

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

.formation on patent family members

Inter ›nal Application No
PCT/US 98/22592

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0483040 A	29-04-1992	JP 6019655 A	28-01-1994
EP 0587519 A	16-03-1994	JP 2527904 B	28-08-1996
		JP 6110593 A	22-04-1994