

Comparison of Keenan Claim 1 Against Keenan Provisional

Claim 1 of Keenan	Disclosure in Keenan Provisional
1. A computer-implemented graphical user interface and integrated object environment system, comprising:	Keenan Provisional discloses this preamble. For example, Keenan Provisional discloses: Summary of Invention In previous business analysis systems, complex, analytically intensive generation of numerical output for business decisions have not also conveyed to the benefactors of those business decisions the costs, risks and accountability inherent in the information assets employed to generate such output. According to embodiments of the present invention, an Integrated Object Environment (IOE) running in a graph database environment is used. Diagrams of the production chain of information assets are created. These diagrams depict data flows linking together these information assets, for which the final product is calculated output ("calcs") used for a specific business purpose. These diagrams are loaded into a repository that stores, reveals and allows for maintenance of the topology. Embodiments include a standard of diagramming for classifying the objects and defining the format of the diagram. Design of object classifications is suited to capture the properties of the information assets in a way that associates the risk, cost, quality, accountability of the assets to the product chain that produces calcs which the business utilizes to set premiums and reserves, manage investment risk and produce financial reports and compliance reports. The same end-to-end production chain of information assets that produces calcs is represented in the topological repository and connected with its respective properties using the IOE. These properties (e.g. costs, risks and accountability) are then compiled and aggregated via their objects in the topology along the production chain which produced the calc and as such can then be associated with the business outcome that the calc enabled. p. 1.
a graph database storing graphical representations of topologies of physical infrastructure components for an enterprise as objects and relationships among the objects, the graphical representations comprising characteristics for the components, wherein data flow between the objects is defined as an object class stored in the graph database;	Keenan Provisional discloses this limitation. For example, Keenan Provisional discloses: According to embodiments of the present invention, an Integrated Object Environment (IOE) running in a graph database environment is used. Diagrams of the production chain of information assets are created. These diagrams depict data flows linking together these information assets, for which the final product is calculated output ("calcs") used for a specific business purpose. These diagrams are loaded into a repository that stores, reveals and allows for maintenance of the topology. p. 1. I. Introduction A. Overview An Integrated Object Environment (IOE) running in a graph database environment. Uses include storing, revealing and maintaining value and risk of information assets, such as the topology of an analytical infrastructure in query-driven, graph database. The system stores, reveals and links characteristics (Structural Metadata) of each object in the topology either within the IOE or in a linked repository, such that accountability, benefits and consequences of using those assets can easily and repeatedly be measured in the cost and profitability of the outcomes those assets enable. The system provides a wide-variety of visual presentation options of the topology allowing the user to view, using GUI, a subset of the topology by selecting use, object function class, functional specificity, relationship to specific object and other criteria. It provides a medium for applying user-provided logic to the topology directly, so that any component or aspect of it can be isolated, retrieved, and assessed. p. 1.
an integrated object environment (IOE) system, coupled to the graph database	Keenan Provisional discloses this limitation. For example, Keenan Provisional discloses:

and further coupled to a communications network;

Summary of Invention

In previous business analysis systems, complex, analytically intensive generation of numerical output for business decisions have not also conveyed to the benefactors of those business decisions the costs, risks and accountability inherent in the information assets employed to generate such output.

According to embodiments of the present invention, an Integrated Object Environment (IOE) running in a graph database environment is used. Diagrams of the production chain of information assets are created. These diagrams depict data flows linking together these information assets, for which the final product is calculated output ("calcs") used for a specific business purpose. These diagrams are loaded into a repository that stores, reveals and allows for maintenance of the topology.

Embodiments include a standard of diagramming for classifying the objects and defining the format of the diagram. Design of object classifications is suited to capture the properties of the information assets in a way that associates the risk, cost, quality, accountability of the assets to the product chain that produces calcs which the business utilizes to set premiums and reserves, manage investment risk and produce financial reports and compliance reports. The same end-to-end production chain of information assets that produces calcs is represented in the topological repository and connected with its respective properties using the IOE. These properties (e.g. costs, risks and accountability) are then compiled and aggregated via their objects in the topology along the production chain which produced the calc and as such can then be associated with the business outcome that the calc enabled.

p. 1.

Figure 2.1

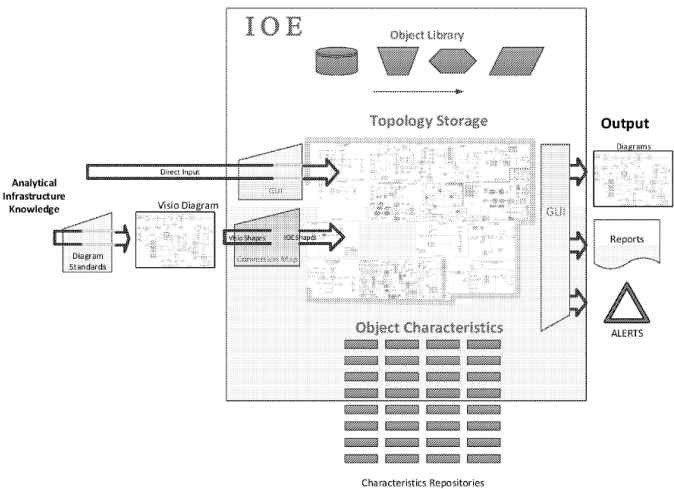


Figure 2.1.

the IOE system being configured to stack statements by applying sequences of the statements that alternate using internal functions of the IOE and referencing the topologies of the physical infrastructure components;

Keenan Provisional discloses this limitation. For example, Keenan Provisional discloses:

II. Core Technology

The IOE is intended to be effectively a dynamic map of the firm's analytic infrastructure including all physical assets that contribute to analysis, analytic reporting, and the information created by processing data. The major components of the IOE appear in Figure 2.1

Figure 2.1

Each real world component is defined as an object with associated characteristics such as state, and its relationship to other objects. Rendering software (GUI) converts the graph database + associated characteristics into Views, which are exportable as diagrams, as well as reports and alerts.

As described in Fig.1, the IOE is intended to reside in a graph database utilizing, object-oriented storage technology, where each physical infrastructure component is represented by an object class from a parsimoniously designed object class set. The relationships among objects are represented by a parsimonious set of pre-defined links. The data flow between the objects is also defined as an object class. The object classes and links are designed to have features that allow for its characterization including specification its state. The relationships characterizing the link s between objects are preferably fixed (non-modifiable), to accommodate a restriction on data flow objects feeding into other data flow objects. This environment allows for storage of the objects that represent the analytic infrastructure, and the structural metadata. A scripting language is preferably adopted and an interface designed that allows users to build and store/run complex programming statements that apply logic directly to the topology or any subset thereof.

p. 3.

D. Scripting, Function Nesting, and Stacking

For the language to support the twin goals of;

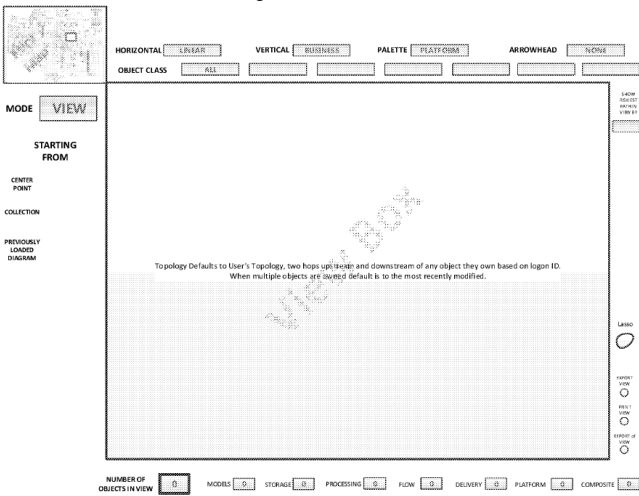
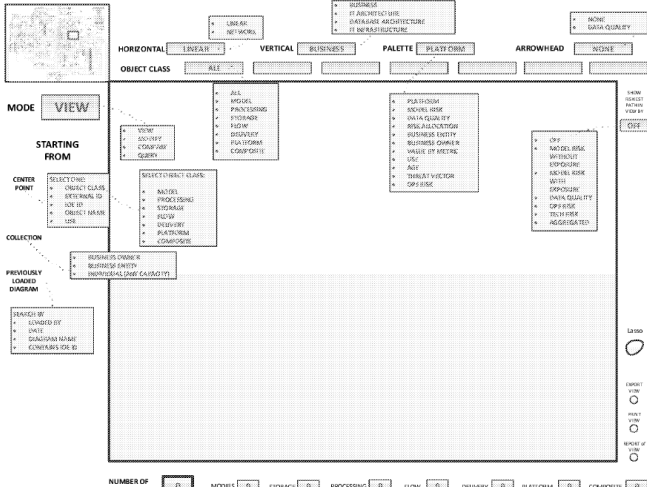
- i) Being able to apply logic to the topology to make transparent any part of it and/or critical characteristics of any part of it, and
- ii) To make the IOE a structural link among data and metadata repositories tied to the objects but not objectified in the IOE

It preferably supports both nested internal functions, and nested SQL statements. We use two programming environments:

- i) An interactive window in which commands can be issued directly against the IOE and the results screened or directed to stored objects or other output file formats, and
- ii) A scripting environment in which multi-line programs can be written, tested, stored, and executed either manually or by chron activation

	1. Nesting Functions Direct nesting of internal functions would work as in the following example: <pre>dplot(topology(supports(CCAR.engine)))</pre> In the above statement, <code>dplot()</code> is a function that generates a visual diagram for an infrastructure subset, <code>topology()</code> is, as described above, is a function that creates an annotated list of identified objects, where that annotation includes the required information for graphical display and/or other types of analyses, and <code>supports()</code>, as described above identifies all of the objects supporting an end-use reporting tool. So, given that the 'CCAR.engine' is a tool that creates official CCAR reports, this one-line statement should draw a diagram for the entire analytic infrastructure that supports CCAR reporting. We propose to use the object oriented programming convention of applying conditioning through qualifiers expressed in square brackets and working from the innermost condition to the outermost condition. For example: <pre>mods<-model\$extid{odd{model{type=="Pricing"}}</pre> would return a vector of external model IDs for which all of the models would be pricing models and all have odd ID values Other Statement Examples: <pre>list<-nests(platform[33])[models]</pre> should return a list of only models nested in the platform whose ID is 33. 2. Stacking of Statements By stacking we are thinking of applying sequences of statements that may alternate using the internal functions of the IOE and referencing the topology directly, and using SQL or a similar language to recover information from sources external to the IOE based on IDs returned/filtered/processed internally. For example, using the example presented immediately above, <pre>list<-nests(platform[33])[models] SELECT ModelOwner FROM CHARACTERISTIC.MODELINVENTORY WHERE MODELID IN list</pre> should return a list of model owners corresponding only to models nested in the platform whose ID is 33, and leaving list as an object in temporary memory. Alternatively, and avoiding the preservation of an intermediate object in temporary memory, the condensed statement, <pre>SELECT ModelOwner FROM CHARACTERISTIC.MODELINVENTORY WHERE MODEL IN (nests(platform[33])[models])</pre> pp. 10-11.
the IOE system being further configured to stack statements by using SQL to recover information from sources external to the IOE system based on identifiers that are at least one of returned, filtered or processed internally;	Keenan Provisional discloses this limitation. For example, Keenan Provisional discloses: 2. Stacking of Statements By stacking we are thinking of applying sequences of statements that may alternate using the internal functions of the IOE and referencing the topology directly, and using SQL or a similar language to recover information from sources external to the IOE based on IDs returned/filtered/processed internally. For example, using the example presented immediately above, <pre>list<-nests(platform[33])[models] SELECT ModelOwner FROM CHARACTERISTIC.MODELINVENTORY WHERE MODELID IN list</pre> should return a list of model owners corresponding only to models nested in the platform whose ID is 33, and leaving list as an object in temporary memory. Alternatively, and avoiding the preservation of an intermediate object in temporary memory, the condensed statement, <pre>SELECT ModelOwner FROM CHARACTERISTIC.MODELINVENTORY WHERE MODEL IN (nests(platform[33])[models])</pre> p. 11.

the IOE system being further configured to update external reference data, the objects and topological relationships;	Keenan Provisional discloses this limitation. For example, Keenan Provisional discloses: E. Updates & Maintenance 1. Updates of external reference data 2. Updates to objects and topological relationships p. 12.
a computing device comprising a display unit, the computing device coupled to the IOE system through the communications network and operable to:	Keenan Provisional discloses this limitation. For example, Keenan Provisional discloses: 5. Object Class View The View defaults to displaying all Object Classes. However the user can restrict this display by selecting only one or a subset of Object Classes to the View. Below is an example of a View with limited Object Classes. p. 21. VI. Advanced User Tools A. Comparing Diagrams: User selects two diagrams: a diagram previously imported into Visio and a diagram to be imported. One will be the "BASE" and the other will be the "CHANGED". Comparisons can be done from any View Option combination. There are two methods to comparing; Melding where all objects from both diagrams are displayed and color coded to indicate which are Common, which are New and which were on the Base diagram only. The other method is Twin Screen, where both the diagrams are displayed with the same color code representing the change in Object from Base to Changed. p. 28.
generate a display for the display unit, the display for representing a first topology from the graph database and comprising:	Keenan Provisional discloses this limitation. For example, Keenan Provisional discloses: 5. Object Class View The View defaults to displaying all Object Classes. However the user can restrict this display by selecting only one or a subset of Object Classes to the View. Below is an example of a View with limited Object Classes. p. 21. VI. Advanced User Tools A. Comparing Diagrams: User selects two diagrams: a diagram previously imported into Visio and a diagram to be imported. One will be the "BASE" and the other will be the "CHANGED". Comparisons can be done from any View Option combination. There are two methods to comparing; Melding where all objects from both diagrams are displayed and color coded to indicate which are Common, which are New and which were on the Base diagram only. The other method is Twin Screen, where both the diagrams are displayed with the same color code representing the change in Object from Base to Changed. p. 28.

a viewing region for presenting the first topology in one of a plurality of pre-determined views;	Keenan Provisional discloses this limitation. For example, Keenan Provisional discloses: Figure 3.1 View model  Figure 3.1. Figure 3.2 View mode showing drop down values for user-selectable boxes  Figure 3.2.
a mini map region for presenting the context of the first topology within a representation of a larger topology for the enterprise; and	Keenan Provisional discloses this limitation. For example, Keenan Provisional discloses:

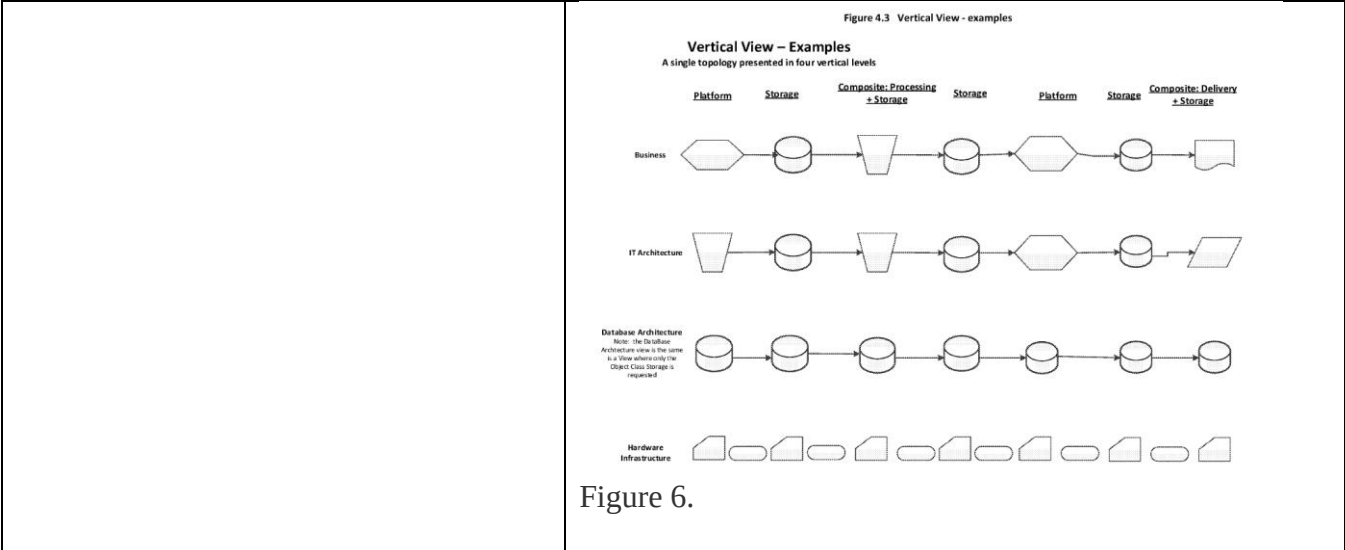


Figure 6.

one or more data regions for presenting calculated data based on the characteristics of the components in the first topology.

Keenan Provisional discloses this limitation. For example, Keenan Provisional discloses:

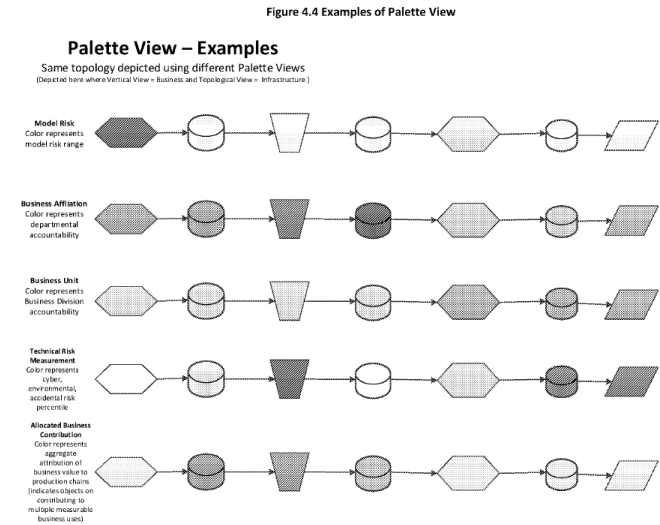


Figure 7.