



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NUMBER	FILING or 371(c) DATE	GRP ART UNIT	FIL FEE REC'D	ATTY. DOCKET NO.	TOT CLAIMS	IND CLAIMS
62/487,370	04/19/2017		260			

23460
LEYDIG VOIT & MAYER, LTD
TWO PRUDENTIAL PLAZA, SUITE 4900
180 NORTH STETSON AVENUE
CHICAGO, IL 60601-6731

CONFIRMATION NO. 8430
FILING RECEIPT



Date Mailed: 04/27/2017

Receipt is acknowledged of this provisional patent application. It will not be examined for patentability and will become abandoned not later than twelve months after its filing date. Any correspondence concerning the application must include the following identification information: the U.S. APPLICATION NUMBER, FILING DATE, NAME OF APPLICANT, and TITLE OF INVENTION. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this receipt. **If an error is noted on this Filing Receipt, please submit a written request for a Filing Receipt Correction. Please provide a copy of this Filing Receipt with the changes noted thereon. If you received a "Notice to File Missing Parts" for this application, please submit any corrections to this Filing Receipt with your reply to the Notice. When the USPTO processes the reply to the Notice, the USPTO will generate another Filing Receipt incorporating the requested corrections**

Inventor(s)

Sean C. Keenan, Reading, CT;
James G. Barrett, New York, NY;
Gordon Cooper, Spring, TX;

Applicant(s)

Sean C. Keenan, Reading, CT;
James G. Barrett, New York, NY;
Gordon Cooper, Spring, TX;

Power of Attorney:

Aaron Feigelson--59022

Permission to Access Application via Priority Document Exchange: No

Permission to Access Search Results: No

Applicant may provide or rescind an authorization for access using Form PTO/SB/39 or Form PTO/SB/69 as appropriate.

If Required, Foreign Filing License Granted: 04/25/2017

The country code and number of your priority application, to be used for filing abroad under the Paris Convention, is **US 62/487,370**

Projected Publication Date: None, application is not eligible for pre-grant publication

Non-Publication Request: No

Early Publication Request: No

Title

INTEGRATED OBJECT ENVIRONMENT

Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications: No

PROTECTING YOUR INVENTION OUTSIDE THE UNITED STATES

Since the rights granted by a U.S. patent extend only throughout the territory of the United States and have no effect in a foreign country, an inventor who wishes patent protection in another country must apply for a patent in a specific country or in regional patent offices. Applicants may wish to consider the filing of an international application under the Patent Cooperation Treaty (PCT). An international (PCT) application generally has the same effect as a regular national patent application in each PCT-member country. The PCT process **simplifies** the filing of patent applications on the same invention in member countries, but **does not result** in a grant of "an international patent" and does not eliminate the need of applicants to file additional documents and fees in countries where patent protection is desired.

Almost every country has its own patent law, and a person desiring a patent in a particular country must make an application for patent in that country in accordance with its particular laws. Since the laws of many countries differ in various respects from the patent law of the United States, applicants are advised to seek guidance from specific foreign countries to ensure that patent rights are not lost prematurely.

Applicants also are advised that in the case of inventions made in the United States, the Director of the USPTO must issue a license before applicants can apply for a patent in a foreign country. The filing of a U.S. patent application serves as a request for a foreign filing license. The application's filing receipt contains further information and guidance as to the status of applicant's license for foreign filing.

Applicants may wish to consult the USPTO booklet, "General Information Concerning Patents" (specifically, the section entitled "Treaties and Foreign Patents") for more information on timeframes and deadlines for filing foreign patent applications. The guide is available either by contacting the USPTO Contact Center at 800-786-9199, or it can be viewed on the USPTO website at <http://www.uspto.gov/web/offices/pac/doc/general/index.html>.

For information on preventing theft of your intellectual property (patents, trademarks and copyrights), you may wish to consult the U.S. Government website, <http://www.stopfakes.gov>. Part of a Department of Commerce initiative, this website includes self-help "toolkits" giving innovators guidance on how to protect intellectual property in specific countries such as China, Korea and Mexico. For questions regarding patent enforcement issues, applicants may call the U.S. Government hotline at 1-866-999-HALT (1-866-999-4258).

LICENSE FOR FOREIGN FILING UNDER

Title 35, United States Code, Section 184

Title 37, Code of Federal Regulations, 5.11 & 5.15

GRANTED

The applicant has been granted a license under 35 U.S.C. 184, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" followed by a date appears on this form. Such licenses are issued in all applications where the conditions for issuance of a license have been met, regardless of whether or not a license may be required as set forth in 37 CFR 5.15. The scope and limitations of this license are set forth in 37 CFR 5.15(a) unless an earlier

license has been issued under 37 CFR 5.15(b). The license is subject to revocation upon written notification. The date indicated is the effective date of the license, unless an earlier license of similar scope has been granted under 37 CFR 5.13 or 5.14.

This license is to be retained by the licensee and may be used at any time on or after the effective date thereof unless it is revoked. This license is automatically transferred to any related applications(s) filed under 37 CFR 1.53(d). This license is not retroactive.

The grant of a license does not in any way lessen the responsibility of a licensee for the security of the subject matter as imposed by any Government contract or the provisions of existing laws relating to espionage and the national security or the export of technical data. Licensees should apprise themselves of current regulations especially with respect to certain countries, of other agencies, particularly the Office of Defense Trade Controls, Department of State (with respect to Arms, Munitions and Implements of War (22 CFR 121-128)); the Bureau of Industry and Security, Department of Commerce (15 CFR parts 730-774); the Office of Foreign Assets Control, Department of Treasury (31 CFR Parts 500+) and the Department of Energy.

NOT GRANTED

No license under 35 U.S.C. 184 has been granted at this time, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" DOES NOT appear on this form. Applicant may still petition for a license under 37 CFR 5.12, if a license is desired before the expiration of 6 months from the filing date of the application. If 6 months has lapsed from the filing date of this application and the licensee has not received any indication of a secrecy order under 35 U.S.C. 181, the licensee may foreign file the application pursuant to 37 CFR 5.15(b).

SelectUSA

The United States represents the largest, most dynamic marketplace in the world and is an unparalleled location for business investment, innovation, and commercialization of new technologies. The U.S. offers tremendous resources and advantages for those who invest and manufacture goods here. Through SelectUSA, our nation works to promote and facilitate business investment. SelectUSA provides information assistance to the international investor community; serves as an ombudsman for existing and potential investors; advocates on behalf of U.S. cities, states, and regions competing for global investment; and counsels U.S. economic development organizations on investment attraction best practices. To learn more about why the United States is the best country in the world to develop technology, manufacture products, deliver services, and grow your business, visit <http://www.SelectUSA.gov> or call +1-202-482-6800.

SCORE Placeholder Sheet for IFW Content

Application Number: 62487370

Document Date: 04/19/2017

The presence of this form in the IFW record indicates that the following document type was received in electronic format on the date identified above. This content is stored in the SCORE database.

Since this was an electronic submission, there is no physical artifact folder, no artifact folder is recorded in PALM, and no paper documents or physical media exist. The TIFF images in the IFW record were created from the original documents that are stored in SCORE.

- Drawing

At the time of document entry (noted above):

- USPTO employees may access SCORE content via eDAN using the Supplemental Content tab, or via the SCORE web page.
- External customers may access SCORE content via PAIR using the Supplemental Content tab.

PROVISIONAL APPLICATION FOR PATENT COVER SHEET

This is a request for filing a PROVISIONAL APPLICATION under 37 CFR 1.53(c).

INVENTOR(S)							
Given Name (first and middle [if any])		Family Name or Surname		Residence (City and either State or Foreign Country)			
Sean C. James G. Gordon		Keenan Barrett Cooper		Reading, CT New York, NY Spring, TX			
☐ Additional inventors are being named on the <i>separately numbered sheets attached hereto.</i>							
TITLE OF THE INVENTION (500 characters max)							
INTEGRATED OBJECT ENVIRONMENT							
CORRESPONDENCE ADDRESS							
☒ Customer No. 23460							
ACCOMPANYING APPLICATION PARTS (check all that apply)							
☒ Specification <i>(including any claims and abstract)</i>		Number of Pages: 35		☐ Power of Attorney			
☒ Drawings		Number of Sheets: 17		☐ Assignment			
☐ Application Data Sheet. See 37 CFR 1.76				☐ CD(s), Number			
				☐ Other (specify)			
APPLICATION FEES							
☐ Applicant claims small entity status. See 37 CFR 1.27.							
				SMALL ENTITY		OTHER THAN A SMALL ENTITY	
PROVISIONAL FILING FEE				\$130		\$260	
☐ Application Size Fee – If the application and drawings exceed 100 sheets of paper, enter number of sheets here: - If application is filed in paper form, enter this number in Total Sheets, below. - If application is filed via EFS-Web, multiply this number by 0.75 and enter result in Total Sheets, below.							
Total Sheets =	- 100 =		+ 50 =	(round up to a whole number)	x 200 =	x 400 =	
TOTAL AMOUNT TO BE CHARGED					TOTAL	TOTAL	\$260
FEES FOR THIS PROVISIONAL APPLICATION FOR PATENT							
☒ The Commissioner is hereby authorized to charge the total filing fee(s) shown above, or credit any overpayment, to Deposit Account Number 12-1216.							
☒ The Commissioner is hereby authorized to charge any deficiencies in filing fees to Deposit Account Number 12-1216.							
The invention was made by an agency of the United States Government or under a contract with an agency of the United States Government:							
☒ No.							
☐ Yes, the name of the U.S. Government agency and the Government contract number are:							

LEYDIG

/Aaron R. Feigelson/

Aaron R. Feigelson, Reg. No. 59,022

Telephone: 312-616-5600

Facsimile: 312-616-5700

Date: April 19, 2017

INTEGRATED OBJECT ENVIRONMENT

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.

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.

The system preferably operates according to the following principles:

- Every object class is unique and designed to allow for specific features including states.
- A library of object classes created is parsimonious to allow for a wide range of uses and allows for a high degree of scalability and modularization.
- The IOE stores objects and their relationships as well as the object's structural metadata ("Characteristics").
- Diagrams are rendered by a front end/GUI of the IOE and are both input and output of the IOE. Diagrams are not stored in the IOE; instead the diagram serves as a communication medium to and from users and the IOE.
- There is a one to one relationship between an object class and a diagram shape.
- The diagramming standards are pre-specified sets of object shapes and formats and other visual elements.
- Diagrams created in the IOE will automatically synchronize with same intelligence stored in structural metadata associated with the object
- Objects are related to other objects through "links" that obey pre-set rules, thus establishing the logic of the topological structure and enforcing integrity within the system.
- The IOE preferably differentiates between 'hard data' (i.e. observations and measurements, and arithmetic combinations thereof) and 'calculations' which represent modeled combinations of inputs. We refer to the former only as 'data' and the latter as 'calcs'.
- Read and write access is for multiple levels of engagement:
 - A. Access and permissions are incorporated based on user function, structural metadata (Characteristics) describing business organization structure and business use cases.
 - B. All user levels have access to a simple, fast, and feature-rich application set.
- At least one object in a Task Sequence topology exists in an Infrastructure topology.

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.

A. Object Classes

Each object class and sub-class is itself represented as a list of fixed characteristics – that fixed set of characteristics define the object class, and establish the response mode for the logical functions (described below). The object class may also have a state variable as part of the characteristic set. Each class may also permit extension to include user-defined characteristics, however from a core technology perspective such user-defined characteristics should not affect the behavior of the objects.

Objects that contain all of the information for the fixed characteristic set that defines them are considered “fully identified”. Objects that do not contain all of the information for the fixed

characteristic set that defines them are considered “under-identified”. Under-identified objects may exist with the IOE but may not be supported by the full complement of internal functions.

Each object should have two identifiers: one internal to the IOE, and one that ties the object to an authoritative external repository. For example, all models in a company inventory (current, historical, and to some extent prospective) may be entries in one external data model characteristics repository system, along with a wide variety of descriptive data and metadata. Therefore, all “model” objects in the IOE need to carry their repository IDs as well as an ID internal to the IOE. By extension then, all objects in the IOE, with the exception of data flow objects must be associated with an authoritative external information repository, and linked directly to that repository by an ID unique to that repository. Objects without such a linkage are under-identified. The external identifier is preferably flexible to accommodate any potential diversity.

Every object in the IOE should carry with it, in addition to its defining object-specific and topology-related information:

- i) An object class identifier
- ii) A creation date
- iii) A creation source code
- iv) An internal IOE ID
- v) An external repository ID

Objects independent of the infrastructure can be stored in the IOE. These relationships are referred to as Task Sequence relationships as opposed to Infrastructure relationships.

The analytical infrastructure is a network of analytical components such as models, EUTs, databases, reports and dataflow information assets. There is immense diversity in the ecosystem of the analytical infrastructure. However, the IOE framework and approach is to place structure within this diversity and group the real world components into object classes. Objects classes can be further grouped into sub-classes. Objects should be as parsimoniously constructed as possible to enhance control, reduce clutter and maintenance costs. For each object class, characteristics are grouped into three categories:

- i) Required (mandatory, failure to include results in an under-identified object)
- ii) Pre-specified (information normally expected to, that supports general querying), and
- iii) User-defined (allowable extensions to the data/metadata embedded directly in the object)

The following are Object Classes:

1. Storage Objects

Data Storage object class includes databases, text files, spreadsheets wherein the purpose is to store data – even if temporarily. Effectively, if data, destined for inclusion in analysis or reporting, is held in a location a data storage object must exist. While we view data as “physical” and part of the overall analytic infrastructure, we do not seek to objectify production data at rest and in bulk. . For data

storage objects, certain boundary cases can be identified: models and modeling platforms usually require input files, which may remain resident as objects serving only within those environments. As these tend to be single-use objects and not formatted for secondary retrieval, we would exclude them from the topology. Alternatively, certain modeling platforms require the creation of data files that do contemplate re-use or multipurpose use, and explicitly facilitate their use for data storage. SAS data files/databases, for example, fall into this category and should be included in the topology as data storage objects.

Such a broad definition of data storage objects suggests that a great many may not be inventoried in any external repository, and key characteristics and metadata may not be available. However we think the trade-off is beneficial, and prioritize the clarity of the object classification over a reduction in under-identified objects. Very likely, under-identified data storage objects would fit primarily into a few subclasses (e.g. non-model EUTs) and small improvements in the control environment could reduce their number significantly.

2. Flow Objects

These represent the passing of data¹ from one object to the other. As such, each object should include:

- i) Field names
- ii) Frequency code and time/date stamp
- iii) Technology code (e.g. ETL, SELECT statement, Read() statement, etc.)
- iv) Data Quality (DQ) grade

While data flow objects have direction, with the source and destination information represented in the topology direction is implicitly included and so an explicit direction feature does not need to be embedded in the objects themselves. Data element level information should identify whether it is “datum” or is “calc”. Data flow objects link to downstream objects using a “passes to” concept in the data flow object, as opposed to characterizing the linkage as “receives from” in the downstream object.

3. Processing Objects

These objects perform non-model operations on data. This includes the application of arithmetic functions as well as aggregation (or potentially dis-aggregation) of granular data, the combining of data from different sources without applying any modeling, and the reformatting of data.

4. Model Objects

These are objects that receive data and/or outputs from other models, and generate outputs restricted to:

- i) Forecasts/projection
- ii) Estimates of unknown/unobservable values
- iii) Probability measures including probability distributions and values derived therefrom, and

¹ We distinguish between quantitative historical observations, or true “data”, and model outputs which are not observations but expectations or estimates which we call “calcs”. But in order to reduce clutter in the document we often use the common convention of applying the term “data” to cover both. Where it matters, we make the distinction explicitly.

iv) Quantitative hypothesis test values

It is the generation of any of these outputs that defines a model. Models have two modes that must be separately/simultaneously captured within the topology: development mode, and deployment mode. The reason for the distinction is that data and other inputs used to develop the model (including data storage and data flow objects) may differ from those used to run the model once deployed. Moreover, the development platform may differ from the deployment platform. For some models, one mode may not exist or not be identifiable. This could include cases where a model exists for extended periods of time in development mode without being deployed, as well as vendor models where the development mode information is limited or not provided. However, the 'modal' nature of models must be an inherent characteristic of the object class.

5. Platform Objects

These include physical and/or software systems that house data storage objects, models, or delivery tools, or that support the production of data flow objects. For example, a database, in combination with automated or user-interfaced query tools, may drive models that generate analytical reports, and such systems as a whole may reside on a server, a non-server memory device, or cloud service. Such systems would fall into the "platform" object class. Similarly, a modeling environment used to develop and potentially house models, e.g. Matlab, would fall into the platform class. By convention, we refer to objects housed within a platform as "nested". Platform object must be able to accommodate many-to-many relationships within its nested objects since, for example, a database may house multiple sub-databases, while a very large database may be distributed across multiple servers.

6. Delivery Tool Objects

Delivery tools represent objects that supply information to users. Effectively, these are the *final* objects in any purpose-defined topological subset. They are the *most downstream* of objects in any such subset. These would include such things as financial reporting engines, web-based pricing applications, and control-environment diagnostic applications. It should be noted that reports, such as tables, data files, charts or graphs that are generated by reporting systems, which may also store the supporting data and the generated reports, would be considered composite objects, as described below.

7. Composite Objects

Composite objects are composites of multiple objects designed to accommodate multiple class characteristics. For example, a spreadsheet may behave as both a data storage object and as a model. Essentially, the composite object acts as a wrapper, providing the necessary information to keep the underlying objects fully and independently identified, for each object class it is a member of, and providing an association between class characteristics and the components of the composite object. That is, to re-use the multi-function spreadsheet example above, the composite object would be a dual list of defining characteristics for 'models' and 'data source' objects, and provide a method by which data flow objects could be separately linked to the components. More specifically, some data may flow into the object only to be stored and later fed into some downstream objects and that data flow object needs to exist independently and be fully identified. At the same time, another data flow object may

flow into the composite purely to feed the model never to be re-directed to any other objects, and that data flow object needs to exist independently and be fully identified.

8. Unclassified Objects

These exist as placeholders for objects required to complete (fill in) a local topology that cannot be properly classified due to insufficient information.

B. Object Linkages

Linkages among objects carry the meaning of the activities supported by those objects, and so must themselves be specialized objects that can store critical relationship/topological information. To maximize integrity, there must be a strict rule set governing permissible linkages including permissible nesting relationships, object-to-object restrictions, origination and terminal node restrictions, etc. Table 1 below presents a partial list of possible linkage rules.

Table 1 - Object Class Function table and Linkage Rules

Object Class	Sub Class	Function	Object Linkage Rules: Object can be linked to:
Model	Development	Perform model operations on information	Flow or Task Sequence Object
	Deployment		Flow or Task Sequence Object
Platform		Host environment for other Model, Data Storage, Delivery tools or Data Processing objects	Flow or Task Sequence Object
Processing		Perform non-model operations on information	Flow or Task Sequence Object
Storage		Store information	Flow Object
Flow		Move information	Any except Flow Object or Task Sequence Object
Delivery Tool		Report information	Flow or Task Sequence Object
Composite		Data Processing or Modeling and Data Storage	Flow or Task Sequence Object
Unclassified		Various	Flow or Task Sequence Object

C. Functions

The required capability is to have a set of pre-specified functions that can be executed in a shell to return lists of objects and object relationship information. The pre-specified functions should have capabilities to allow for conditions and filters, some of which may be internal to the functions, some of which may be applied outside the functions in the form of a “where” statement. A standard script-driven and batch executable language may be adopted.

1. Object Representation

Some straightforward object identification and filtering syntax needs to be provided. We suggest consistency with common object oriented programming, using the closest set of square brackets to indicate identity as follows:

```
datastorageobject[33, ioeid=F], or equivalently, datastorageobject[33]
```

returns the data storage object whose external ID is 33. Here ioeid holds the default value F. Hence

```
datastorageobject[100, ioeid=T]
```

returns the data storage object whose internal ID is 100.

Because the objects will be themselves lists, a component extraction syntax is used. We suggest consistency with common object oriented programming, using the dollar sign \$ to indicate an object component. For example, if 'model.name' is an included characteristic of model objects, and the model with an external ID of 23456543 is named "European Windstorm Model", then

```
model[23456543]$model.name
```

should return "European Windstorm Model". Similarly, if 'location' and 'owner' are both characteristics of a 'server' sub-class of platform, then the statement,

```
output<-platform[platform.type=="server", c($ioeid, $location, $owner)]
```

should return a list of all servers identified by ID, location, and owner.

2. Basic Functions

Some examples of required functions are presented below.

a) `topology(x)`

This function extracts from the overall topology, a sub-topology based on an object list *x*, or other subsetting criterion. The requirement here is that this function return a list-object with the appropriate object-level information and formatting to support the generation of visual diagrams and other similar analyses. As with many of required functions, the list should include a time/date stamp since the topology will be changing in real time.

b) `linksto(n, m, nested=Y)`

This function returns a list of objects linked to object *n*, where the proximity is *m* steps. That is, if *m*=1, the list included only those object directly linked to *n* along with the value *m* for each object returns. If *m*=2, objects two steps removed from *n* are included in the list, etc.. The default value nested=Y

indicates that "nesting" is a form of "linking" which would cause the function to return platform objects that appear within the given proximity range.

c) `inheritsfrom(n, m)`

This function applies to data storage objects, models, and delivery tools, and generates a list of data flow objects feeding object *n* from *m* levels of directness, along with the value *m* for each object returns. That is, it generates a data lineage list for any object using data.

d) `isnestedin(n)`

This function returns the platform within which object *n* is nested. If object *n* is not nested within any platform, it should return a NULL.

e) `nests(n)`

This function applies only to platform objects and returns a list of objects nested within platform *n*.

f) `supports(a, use="ALL")`

This function applies only to delivery tool objects, and works backwards to create a list of all objects topologically upstream from reporting tool *a*. The "use" modifier would allow the user to narrow the topology to apply only to specific uses of *a* where the object *a* contains enough information to subset the topology.

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:

```
dplot(topology(supports(CCAR.engine)))
```

In the above statement, `dplot( )` is a function that generates a visual diagram for an infrastructure subset, `topology( )` 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 `supports( )`, 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:

```
mods<-model$extid[odd(model[type=="Pricing"])]
```

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:

```
list<-nests(platform[33])[models]
```

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,

```
list<-nests(platform[33])[models]
SELECT ModelOwner
      FROM CHARACTERISTIC.MODELINVENTORY
      WHERE MODELID IN list
```

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,

```
SELECT ModelOwner
      FROM CHARACTERISTIC.MODELINVENTORY
      WHERE MODEL IN (nests(platform[33])[models])
```

should accomplish the same thing.

E. Updates & Maintenance

1. Updates of external reference data
2. Updates to objects and topological relationships

F. Vectorized Object Representation

Given that each object is defined by a fixed set of characteristics, we require a coding capability wherein a vector or character string is designed with a fixed format that applies to all objects in the IOE. Specifically, this string must contain all of the fixed characteristic data for each object so that each segment of the string corresponds to a specific fixed characteristic including the class and subclass information. For example, given that we are not expecting to have more than 99 high-level object classes, the first two digits of the string could contain a code (1-99) indicating what class of object the class is from. Letting the maximum length of the internal object ID be n , then the next n digits could contain the internal ID, and so on. With an appropriately long and properly segmented string, every object could be represented and fully characterized by its string.

G. Topology Diagnostics

The IOE preferably has certain built-in diagnostic capabilities targeted primarily toward automated integrity checking and maintenance. Among these is the need to capture and retain the history of the complete topology. Assuming a syntax where “all” is a permissible subsetting criterion, this could be as simple as running the statement,

Topology(all)

III. Graphical User Interface

There are four modes for the user to interact with the IOE via a Graphical User Interface (GUI).

View: View any section of the topology and Characteristics by Object

Modify: Add, change and delete Objects or Characteristics

Compare: Visualize the difference between two diagrams

Query: Interrogate the topology and Characteristics using standard queries and user – generated queries (script writing). Query results can be tabular (reports) or diagrammatic results in which case the diagram will appear in the View Box.

Each mode brings up a slight different screen format. We refer to the portion of the GUI that the user sees as the “screen” as opposed to the logic which dictates what appears on the screen which would be the unseen parts of the GUI.

Figure 3.View model

Figure 3.2 View mode showing drop down values for user-selectable boxes

The screen contains the following basic components

View Box: This component uses most of the screen. Within the View Box the user sees the topology to view, modify or compare. It is view-only in the View and Compare modes and interactive in the Modify mode.

Mini Map: In the upper left corner is a depiction of the topology. The default view is the entire company. Of course objects and labels are far too small to be seen individually but the purpose of the mini map is to show the user where within the entire company topology the current View is located. This would be analogous to having a street map of Altoona, PA in the main part of the screen (the View Box) while the Mini Map would indicate where on a map of the US Altoona is.

The Mini Map defaults to the following View options

- Horizontal: LINEAR
- Vertical: BUSINESS
- Palette: PLATFORM
- Arrowhead: NONE
- Object Class: ALL

Locator Box: Within the Mini Map is the Locator Box. Regardless of how the user zooms in and out of the Mini Map, the Locator Box will encompass the same topology and so will expand or contract to remain in sync with the Mini Map display as the zoom function is exercised. When the cursor is not in the Mini Map the Locator Box will display the same boundaries represented by the View in the View Box. It does this by automatically shifting position, shrinking and expanding borders to encompass topology represented in the View Box.

Table 3.1 MINI MAP FUNCTIONALITY

Cursor position	On Mini Map outside Locator box	Inside Locator Box
Single left click	Demarcates the Mini Map based on Landmarks set up by the IOE System Administrator. This is a toggle on/off (a second left click removes Landmarks)	Activates locator box boundary allowing user to expand or contract any side or corner of the locator box
Double left click	opens the Mini Map to full screen overtop existing screen. Escape key or double left click shrinks Mini Map back to normal size. Mini Map and Locator Box retain full functionality in this full screen mode.	Transfers topology inside the Locator Box to replace what is in the View Box
Single right click	Brings up properties of the Mini Map. One property allows the user to change the default View options of the Mini Map	Allows user to open another View Box filled with the contents of the Locator Box. Up to six View Boxes can be open at the same time.
Mouse wheel forward	Zooms in the Mini map. The Locator Box boundaries remain fixed relative to objects rather than remaining fixed relative to size on the screen.	Zooms in the Mini map. The Locator Box boundaries remain fixed relative to objects rather than remaining fixed relative to size on the screen.
Mouse wheel back	Zooms out the Mini map. The Locator Box boundaries remain fixed relative to objects rather than remaining fixed relative to size on the screen.	Zooms out the Mini map. The Locator Box boundaries remain fixed relative to objects rather than remaining fixed relative to size on the screen.

IV. Visualizing the Topology

The intelligence embedded in the stored topology both in terms of the object to object relationships and the associated characteristics of objects can be viewed in a variety of ways. One way is by varying the appearance of the visual elements using visual dimensions. The other way is by selection of View options which includes Topology Type, Horizontal and Vertical Views.

A. Visual dimensions

In order to maximize the amount of information depicted in a sub-set of the topology via GUI (referred to as a “View”), most object shapes and lines can also depict the value, range of values of a specific characteristic or the sub-class of the object. The shapes themselves are dimensions as they differentiate object classes.

Aside from shapes, which represent the object class of the object, there are two other dimensions for depicting intelligence; Color and Line Weight

The two dimensions of Color and Line Weight are applied to visual elements of the Topology and is preset rather than being a user option except in the case of special Views. For instance, shape fill color can represent platform association in the default view but represent model risk in a specific Palette View (see below). The number of characteristic or sub-class values possible for each visual component is limited by the differences the normal human eye can detect on today’s screens.

Listed below are the two dimensions of Color and Line Weight applied to the visual elements and in parenthesis the number of possible values. Combining two visual element options can lead to many more possible key characteristic or sub-class possibilities. For instance the combination of Border Thickness and Border Line color can represent 2 x 4 or 8 different possible key characteristic values or sub-classes.

- Border thickness (2)
- Border line color (4)
- Shape fill color (20)
- Shape border color (6)
- Flow Line thickness (2)
- Flow Line color (4)
- Flow Arrowhead fill color (6)

Examples of using visual dimensions.

- The shape border thickness could depict the model object status.

- The shape fill color could depict model risk levels in model objects when selecting the model risk palette View.
- Flow line color could depict whether the flow is generated automatically or manually invoked.

B. Selection of Topology to view

There are four ways for the user to select a subset of the topology to view.

Mini Map:

The user selects the View by entering one or more of the following from a drop-down menu (the list below is not an exhaustive list). The scope of the topology presented includes all objects connected upstream and downstream within the selection criteria requested. The user can also opt to have the output appear in a table which can be exported as excel file. If the criteria entered is not currently stored in the IOE, user will be given the option to add new topology (see below).

Single Object Selection:

- Use
 - Object ID
 - Object Name
- Depicts all objects feeding into and all objects being fed by the Object selected

Multiple Object Selection:

- Business entity
- Diagram previously Imported
- Person's Name (will search all name fields – owner, steward, BUE, model officer, etc)
- Object Class or subclass

After the user makes a selection they can then refine the view by filtering for:

- Upstream or downstream paths only
- Entering a start point or end point or both in addition to the View criteria above
- Example 1: End point = Technical Pricing Model, showing all objects related to business unit = Commercial. The resulting diagram will be drawn showing all objects connected to any object which feeds ultimately to the Technical Pricing Model. Since there is no upstream limit the diagram will show all upstream relationships.
- Example 2: Start Point = Bloomberg, show all objects related to Polypaths application.

There are several options the user can select to view the topology. These options can be selected in combination with one another

Table 4.1 View Options - A summary of the View Options:

View Name	Options	Description
OBJECT CLASS	All Classes	All object classes are depicted
	Specify Object Classes (Flow objects will not be shown unless all object classes are selected)	User can select one or more classes to depict. Data Flow Lines will not be depicted unless all classes are selected. In lieu of data flow lines, objects will be joined by lines that look like data flow lines but the arrow heads will be a different shape.
HORIZONTAL	Linear	An instance of a particular Data Storage object appears for every extract and load downstream of the initial instance of the Data Storage object in the View
	Network	all Data Storage objects appear only once in the View
VERTICAL	Business	Highest level functionality
	IT Architecture	Application centric
	IT Infrastructure	Hardware centric
PALETTE	Platform	color fills the object shapes based on the platform that hosts the object. Objects not hosted by platforms will have white fill.
	Model Risk	color fills the object shapes based on the value of the risk characteristics for the models in the view
	Data Quality	color fills the Flow arrowhead shapes based on the value of the value of data quality characteristic for the data elements feeding the object. This palette can be selected
	Risk Capital Allocation	color fills the object shapes to indicate the absolute value of Risk capital allocated to the object.
	Business Entity	color fills the object shapes based on the Business Entity or Entities that use the output from a Path of objects
	Business Owner	color fills the object shapes based on the business entity that owns or co-owns the object
	Value by Business Metric	color fills the object shapes to differentiate the value of the business metric(s) associated with each production chain of objects in the view.
	Use	color fills the object shapes based on the Uses within the View
	Age	color fills the object shapes based on the value of the age characteristic for that object

	Technical Threat Vector	color fills the object shapes based on the value fo the technical risk ranking characteristic of that object
	Operational Risk	color fills the object shapes based on the value of operational risk characteristic of the object or its associated control
ARROWHEAD	None	Arrowheads depicted in solid black
	Data Quality	color fills the Flow arrowhead shapes based on the value of the value of data quality characteristic for the data elements feeding the object. This palette can be selected

1. Horizontal View

Linear View: This view depicts the production chain of information assets as the data flow sequentially progresses downstream through the chain of information assets. This View is created by following the path of data elements in the data flow objects as they are passed from object to object. Since objects are stored in the IOE as a single instance, the multiple loads and extracts to the same database in a View will be depicted as unclassified objects in the topological space allowing depiction of the proper sequence of the production chain. These Unclassified Objects are created dynamically whenever a Linear View is requested. The Characteristics of these dynamically created Unclassified Objects equals the Characteristics of the Storage Object for which it represents an instance of.

For example, if model 20 produces calc “A” and calc “A” is subsequently loaded to database STS and model 30 extracts calc “A” from STS then by inference, model 20 and model 30 are on the same production chain and 20 is upstream of 30. On the Linear View Model 20 will be connected to Model 30 with a Data Flow object to a Unclassified Object representing the storage in STS followed by a Data Flow object connecting the Unclassified Object to Model 30. And if STS is loaded and extracted from multiple locations along the production chain a proxy shape will be depicted at each load and extract point to maintain production chain continuity.

Network view: This view does not use the data element lineage to infer the path of the production chain but rather depicts only a single instance of an object in the View. Thus all loads and extracts to a common object (e.g. data store object) are shown as immediate upstream and downstream objects. Production chain lineage is not maintained.

Figure 4.1 Horizontal View – Comparison of options

2. Vertical View

This View option allows the user to tune the Functional Specificity of the objects depicted. This is useful when Platform and Composite objects exist within the View. In the Vertical View Composite and Platform objects are considered to be stacked as if in a functional support silo. The purpose of the Vertical View is to simplify the View by isolating only those components of Composite and Platform objects that are essential to the needs of the viewer.

Figure 4.2 Cross-Section Schematic of Vertical View options

User selects View Level (e.g. Business) , the IOE looks for the Object class IDs (first two digits of the Internal ID) based on the chart above and depicts those objects on the View.

The user can also by-pass the standard View Levels and create their own Vertical View by specifying which object classes to be depicted.

Figure 4.3 Vertical View - examples

3. Palette View

Palette View options allow the user to highlight a particular set of Characteristics that are preset such as risk, business ownership, age, etc. A Palette is a View that embeds the intelligent of a different set of key characteristics other than the default View. This is accomplished by switching the mapping of the Visual Elements to characteristics different than what is in the default View. Below are examples of Palette Views:

1. Model Risk : color fills the object shapes based on the value of the risk characteristics for the models in the view.
2. Business Owner: color fills the object shapes based on the business entity that owns or co-owns the object
- 3.
4. Business Entity: color fills the object shapes based on the Business Entity or Entities that use the output from a Path of objects.
5. Use: color fills the object shapes based on the Uses within the View
6. Age: color fills the object shapes based on the value of the age characteristic for that object
7. Technical Threat Vector: color fills the object shapes based on the value of the technical risk ranking characteristic of that object
8. Data Quality: color fills the Flow arrowhead shapes based on the value of the value of data quality characteristic for the data elements feeding the object
9. Operational Risk: color fills the object shapes based on the value of operational risk characteristic of the object or its associated control
10. Value by Business Metric: color fills the object shapes to differentiate the value of the business metric(s) associated with each production chain of objects in the view.
11. Risk Capital Allocation: color fills the object shapes to indicate the absolute value of Risk capital allocated to the object.

A special mode of the Palette View is invoked with in the Compare Mode. In this mode the Palette View is locked on the Compare option which utilizes the color coding set forth in the Compare Mode.

Examples of Palette View

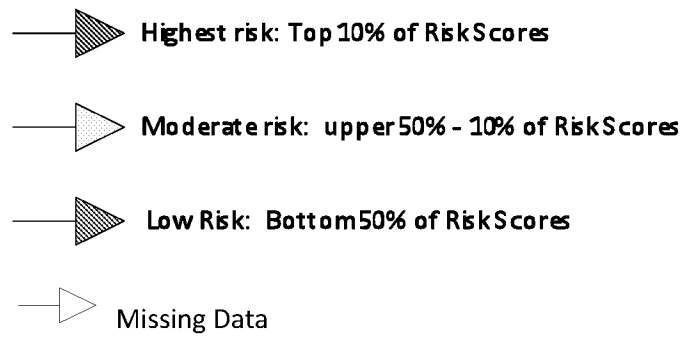
Figure 4.4 Examples of Palette View

4. Arrowhead View

Arrowhead View is intended to depict Data Quality measurement. The method is by color coding the arrowhead fill space. The default Arrowhead color is black.

- Flow Arrowhead Fill Color

Arrowheads will fill with color based on the risk measurement of the upstream calc and data at that point. For example three relative gradations of risk could be represented. Values to determine Data Quality score will be set by Contributors. For example the colors and ranges might represent:



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.

Figure 4.5 Object Class View - example

6. Granularity Options

Objects can be decomposed once given enough characteristics. We anticipate this to happen first with models. A View Option can be selected to depict the internal modules of a model inside a model shape. Similar capability exists for other objects such as Platforms and Composite Objects.

7. Interactive GUI Functions

- **Path Highlighting:** The user can click on any object in the topology on the screen. The View will then highlight the upstream and downstream flows into and out of the object selected. The highlighting is accomplished by using the thicker line weighting on the Flow Objects in different color than normally used for lines. Clicking again on the object toggles this feature off. Paths upstream and downstream from up to six objects can be toggled on at the same time. A different highlighting color may be used for each.

Figure 4.6 Path Highlight for a single Object (Model 1)

Figure 4.7 Path Highlight for two Objects (Model 1 and Model 2)

The Path Highlight can be converted into a Report by clicking a report icon on the screen. Clicking this icon prompts a box to appear asking the user if they would like the Upstream Paths, Downstream Paths or both. Following this selection, the user will be asked to select which object classes they wish to include in the report. Select all is the default. Following this selection the user will be asked which Characteristics of the Object Classes selected would they like on the report. The default is all Characteristics.

- List Characteristics: A full list of the characteristics of an object can be brought to the screen by double (left) clicking the object. Further, double-clicking on a Characteristic brings up the company's Business Glossary entry for that term. The list will appear with an "x" in the upper right corner to allow the user to close the list.

The Characteristic list can be converted into a Report by clicking a report icon on the screen.

- Navigating the View Box: The user can scroll or grab and drag in any direction to see the horizontal limits of the View.
- Riskiest Path: The user can display the Riskiest Path of the View by selecting one of the risk metrics. The IOE will compute based on pre-loaded formulae the path with the highest risk based on the metric selected by the user. The riskiest path will appear in red as seen in the screen below in Figure 4.3. A sample list of risk metrics:
 - Model Risk without exposure
 - Model Risk with exposure
 - Data Quality
 - Ops Risk
 - Tech Risk
 - Aggregated Risk

Figure 4.8 View with Riskiest Path

Search Functions: search diagrams loaded previously

1. Results returned in tabular form.
2. By External ID – return list of diagrams where model was loaded. Allow user to select a diagram to view
3. By Business Affiliation (e.g Business Unit)
4. By Name of Object - user first can select the type of object they are looking for, then enter the name. The IOE will return closest results if no exact match found.
5. Name of Object
6. Load date
7. Name of person who did the import

8. *Links to Documents*

An icon within the object indicates a document electronically stored outside of the IOE has been linked to the object. The icon will represent the document format. By clicking on this icon the application which supports the document will open and the document will appear on the screen.

9. Risk Reports For View Selected

The screen has a box in one corner which will allow the user a quick way to calculate the risk of the selected View. This creates both a visual depiction of the information requested as well as giving the user the option of printing the report. Four examples of those risk reports:

- **Aggregate Model Risk of the View:** Calculation of aggregate model risk for all models in the View.
- **Depict Riskiest Path:** This option depicts the Path of whose objects in aggregate measure riskiest vs all other Paths in the View. The formula for this calculation may be based on Risk value characteristics of the Objects.
- **Depict Lowest Data Quality Path:** This option depicts the Path of whose data elements in aggregate measure lowest in data quality vs all other Paths in the View. The formula for this calculation may be based on Data Quality value characteristics of the Data Elements.
- **Calculate Aggregate risk by View -** Risk values of the objects of the View are aggregated based on an aggregation formula.

10. Exporting

Any View created in the IOE is exportable to Visio by clicking on the Export button on the right side of the View Box. The map of Visio shapes to IOE shapes is the same as the import function.

V. Modifying the Topology

A basic principle of the IOE is that each Object has an owner. This permits assignment of accountability for the Flows feeding and being fed from the Object. Alerts and Acceptances (see following section) require knowing the owner of an Object.

Figure 5.1 Modify Mode

Figure 5.2 Modify mode showing drop down values for user-selectable boxes

A. Import Visio diagrams

A map is preset that maps Visio shapes and formats to IOE shapes and formats.

The user selects the import function in the IOE GUI, which first checks to see if object already exists in IOE based on external ID number first, then name match. If object already exists:

A message is posted to the screen "The following objects matched existing objects from The IOE, new relationships will be loaded to existing object.

The existing Objects will be updated with new relationships based on the imported visio diagram.

The import function also provides the following utilities

Provide object count by class and total count once diagram is imported.

The IOE will save the diagram objects so that it can be reproduced for audit purposes and to do diagram comparisons and allow users to select as a View in the future (see Selection of Topology to View above)

B. Adding objects directly to the IOE

Using a screen in the IOE GUI, the user selects method of adding object (this is not an exhaustive list):

External Object ID

Use

Business Affiliation

Name of owner

Name of object

If the IOE returns a topology with the object(s) already stored that match the criteria the user entered. The user can browse the list to see if the object they want to load is already in the IOE. If not the user can they add the object. Or, the user can modify the object(s) found. See “Modifying Objects” below.

User populates the canvas by dragging necessary shapes to the canvas and entering the minimum amount of information to initialize the object in the IOE. Those minimums are stated in the Core section for each Object Class.

C. Connecting Objects in the View box

Flow Objects connect to any point on the Object shape in the diagram. This differs from the way in which other diagramming products work which requires lines to connect only at certain points on the shape. The reason for expanding the capability of connecting is to allow for more clarity in the depiction of the topology.

D. Modifying object characteristics

Double click on the object to open characteristics table. Double click on the Characteristic and make necessary changes and save.

Capability to do en mass changes (e.g. property change for entire object class). Select object class, select class change option. Follow change drop down menu.

E. Saving Modifications

Unless saved as FINAL, this topology will not be saved in the production topology when the session ends. Instead the temporary topology, in whatever manner it was modified, is saved in a Temporary topology and the person modifying can name the temporary saved topology. Saving as temporary – any changes between existing and temporary adds are then highlighted in the same manner as compare.

F. Storing History

In order to preserve past states of the topology, the IOE preferably stores the history of the full IOE topology on a periodic basis.

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.

Figure 6.1 Compare mode Meld Method

Figure 6.2 Compare mode Twin Screen Method

B. Reports

Other than reports mentioned in the above sections in connection with Views or GUI interfaces, there are special reports which can be created using the GUI. Examples of those appear below.

1. Tree Pair File Report

Objective: The query is to identify pairs of objects of the same class upstream of the selected object. These pairs are referred to as "Tree Pairs".

Tree Pair is defined as a pair of objects, connected via a Data Flow object through any combinations of objects

Working upstream from the selected object, each object of the same class regardless of any object(s) between the selected model and itself is considered a part of the tree of the selected object's class. In succession, the query continues looking upstream from each object in the same object class in the Tree for the next object. This in turn creates a object Pair. The query continues upstream along each Path and Branch of the tree until the last object of the same class is found. The collection of Pairs for a the selected model are referred to as Tree Pairs of object class of "x" where "x" is a given selected model. . The collection of objects that feed the selected object may be connected linearly or in a forked path.

If there is a loop (a sequence in which an object appears more than once), then the sequence is stopped at the second instance of that object.

Parameters:

The user selects the one or more objects for the query to determine the Tree Pairs. These selected objects will be displayed in the first column of the report

2. Lasso Reports:

Reports which use as the select criteria the object set bounded inside an irregular shape drawn by a “freeform” tool (same as the Visio tool called “freeform”). This topology subset can be converted into a Report by clicking a report icon on the screen. Clicking this icon, the user will be asked to select which object classes they wish to include in the report. Select all will be the default. Following this selection the user will be asked which Characteristics of the Object Classes selected would they like on the report. The default is all Characteristics.

Figure 6.3 Lasso Report – example of selecting objects using the Lasso function

C. Alerts

Informing contributors and consumers of the Topology is a key benefit of the IOE. Since characteristics are linked to objects in the topology the changing of key characteristics affects not only the contributors and consumers of that object but also those contributors and consumers downstream. Alerts can be built that can take advantage of the community and accountability within a common topology to warn

others of changes that impact them by virtue of their location on the topology. There are two basic types of Alerts.

1. *Key Characteristic Changes:*

Conditions can be set for each object class for Alerts to be generated upon the change in a characteristic when a new value is entered that outside set parameters. These values can either be associated with the object in the binary object code or can be characteristics in the relevant repository of characteristics for that object class. A period delta check of the repository could look for new values outside the parameters for each object class.

The Alerts can be tailored based on the relationship with Upstream and Downstream objects. The Alerts will be sent via email or other means of electronic communication to advise the owner of the targeted object that a change to a characteristic with a related object has changed outside parameters. An example of this would be the change to model status. When a model validation expires this needs to be communicated to downstream objects of the model. An email would be generated to the owners of the model classes deemed to be impacted by the change to the model characteristic of validation status to inform them of this fact so that they may take action to mitigate the resulting risk.

Changes to characteristics can also include formulas that calculate risk parameters based on new raw values put into to characteristics of one object but flow downstream to other objects. Entire paths can be monitored for changes to aggregate characteristics values. For instance a production chain of four models might be set to alert downstream consumers of those four models if the aggregate model risk of the four models exceeded a certain amount.

2. *Object Capability Changes:*

A more robust set of Alerts can be used to notify upstream and downstream object owners that an object has been added/removed/replaced or otherwise undergone alteration in capabilities. This Alert would be signaled as higher importance than a Characteristic change.

An example would be where a model owner sources data from a new vendor. This addition to the IOE would then trigger an alert to all downstream object owners.

3. *New use of sourced data and generated calculations:*

When users draw flow objects from storage objects and add data elements to the Flow Object Characteristics the IOE will locate the source of the calculations (Model Object) and the source of Data Elements (Storage Object). Once identified, the IOE will send an Alert to the owner of the Model and Storage object advising them that their Data and Calculations are being sent to a new object.

4. *Accepting Changes*

The IOE system administrator can set Acceptance requirements for modifications to the topology. Setting Acceptance Criteria requires the IOE to send an email or other such electronic message to the designated responsible party within a set amount of time of Criteria being met. The designated party has a set amount of time to Accept the modification. Rejecting, or failing to Accept the modification

requires removal of the modification and a notice sent to the modifying party. Both these set times can be set by the IOE system administrator.

Examples of Acceptance Criteria:

Model Owners must accept the modification of use of their Model's output

Storage Owners must accept the modification of use of their Storage output

Process Owners must accept the modification of use of their Process output

Any Object owner must accept the addition of or subtraction of Flow Objects to the Object they own.

VII. Interfaces

A. Maintaining Characteristics (Structural Metadata)

Characteristics from Structural Metadata Repositories for the respective object in the IOE are preferably loaded into the IOE on a daily basis. History is preferably saved for a defined period.

B. Updating Characteristics

The IOE is preferably informed of new and updated topological links and objects using importation of visio diagrams or by diagram maintenance interface.

C. Queries from outside the IOE

The IOE preferably allows queries originating in Structural Metadata repositories to query the topology in the IOE.

VIII. Glossary

Analytical Infrastructure (topology): Depicts Information Assets and information in motion between information assets.

Branch: Term that is used only relative to a selected Path. It is a linear string of linked objects joining a Path at a fork.

Business Affiliation: Business entity associated with object.

Business Metric: Financial value resulting from the information produced by a production chain, tree, Path, Branch or any other collection of objects acting in concert.

EUT/EUC (End User Tool/End User Computing Tool): Software (e.g. Excel spreadsheet) that manipulates data and whose output is consumed by a model, another EUT/EUC or used in a final business report. Some models are EUTs.

Calcs: Output from a model. It remains in this state for its life and thus never reverts to being referred to as “data. If data is transposed or otherwise manipulated by a non-model tool (e.g. excel spreadsheet) the output from this tool is still referred to as “data”. External (to the company) model output is also considered “calcs”.

Canvas: Physical screen space in the IOE Modification layer where user builds or modifies topology

Characteristics: Structural Metadata that describe the properties of an object.

Diagram: Communication vehicle between user and the IOE and between the IOE and the user depicting static topological information in a format selected by, and consumable by users which can be set in documentation, stored as a file in other applications. This term contrasts with the term for the visual depiction of topology seen in the GUI of the IOE which is referred to as a “View”. Another way to understand a diagram is as a fixed portion of the topology. A diagram is topology packaged to be communicated.

Direct Model: Term specific to the CCAR Model Risk Buffer Calculation, model that is mapped to CCAR Scenario Loss.

Fork: Two or more Branches joining at a single object.

Hops: The number of data flow objects between two non-data flow objects.

Immediate (upstream or downstream): First non-data flow object upstream of a selected object (“immediate upstream”) also defines the first non-data flow object downstream from a selected object (“immediate downstream”).

Instance: As used in this document, an instance of an object in a diagram represents not the Objects absolute location in the topology but rather the location of a point in a Linear View where Flow Objects extract from or load to the Object. The reason why multiple instances are used is to depict the production chain of the analytical infrastructure when multiple extracts and loads to the same database are required. In the Network View, there is only one instance of an Object which in the case of multi-use Storage Object will show the linkages of Objects to a single Storage Object thus obfuscating the production chain linearity.

Junction: Multiple Paths join in at a single object and multiple Paths exit from this object.

Junction Problem: Situation where multiple Paths join at a Junction but no information from one or more upstream Paths is conveyed, in any form, by the downstream Paths. This creates a problem for tracking the data lineage and requires that data elements be characteristics of Data Flow objects in order to trace the flow of information through a Junction.

MLD (Model Level Diagram): Type of IOE diagram centered around a specific model. The purpose of this diagram is to depict feeds into the model from all sources (including other models) and the destination of the output (including other models). MLDs are joined together to form Stream Diagrams (see below).

Module: internal component of a model which produces output consumed within the model.

Object: (see Object Class section above)

Path: A sequential set of objects connected by data flow without a fork. It is defined by a single object on the upstream end and a single object on the downstream end. It is possible to have a multiple paths starting at the designated upstream object and ending at the designated downstream object although this would indicate a probable misconstruction of the diagram or process.

Production Chain: Collection of objects used in the creation of specific business useable output (delivery tool object contents)

Structural Metadata: For this document, the term “Characteristic” is used interchangeably. Examples of structural metadata for most objects: owner, risk measurement, age, business affiliation, users

System Administrator (IOE System Administrator): The individual with the authority and access rights for setting rules, operating parameters, Alerts and Acceptance Criteria of the IOE.

Topology: Representation of the relationships between objects in an analytical infrastructure. In the case where analytical infrastructure produces information, these relationships represent the storage, processing and flow of information. Communicating topology to and from the IOE is done with diagrams.

Tree: The collection of objects (or a specific class of objects; models for example “Model Tree”) that are on the paths feeding a selected object from upstream or the collection of objects on paths fed from a

specific object downstream. Phraseology would be “Upstream tree of object X”. Or “Upstream Model tree of Model x”.

Task Sequence (Topology): Depicts tasks in process order sequence including tasks that require use of information assets.

Ultimate (*upstream or downstream*): Object depicted as the first object upstream of a selected object (“ultimate upstream”) also the final object downstream from a selected object (“ultimate downstream”).

View: Depiction of topology in the IOE via GUI. A view that is exported or printed from the IOE is a “Diagram”.

View Box: that portion of the IOE GUI screen that contains the View.

Figure 2.1

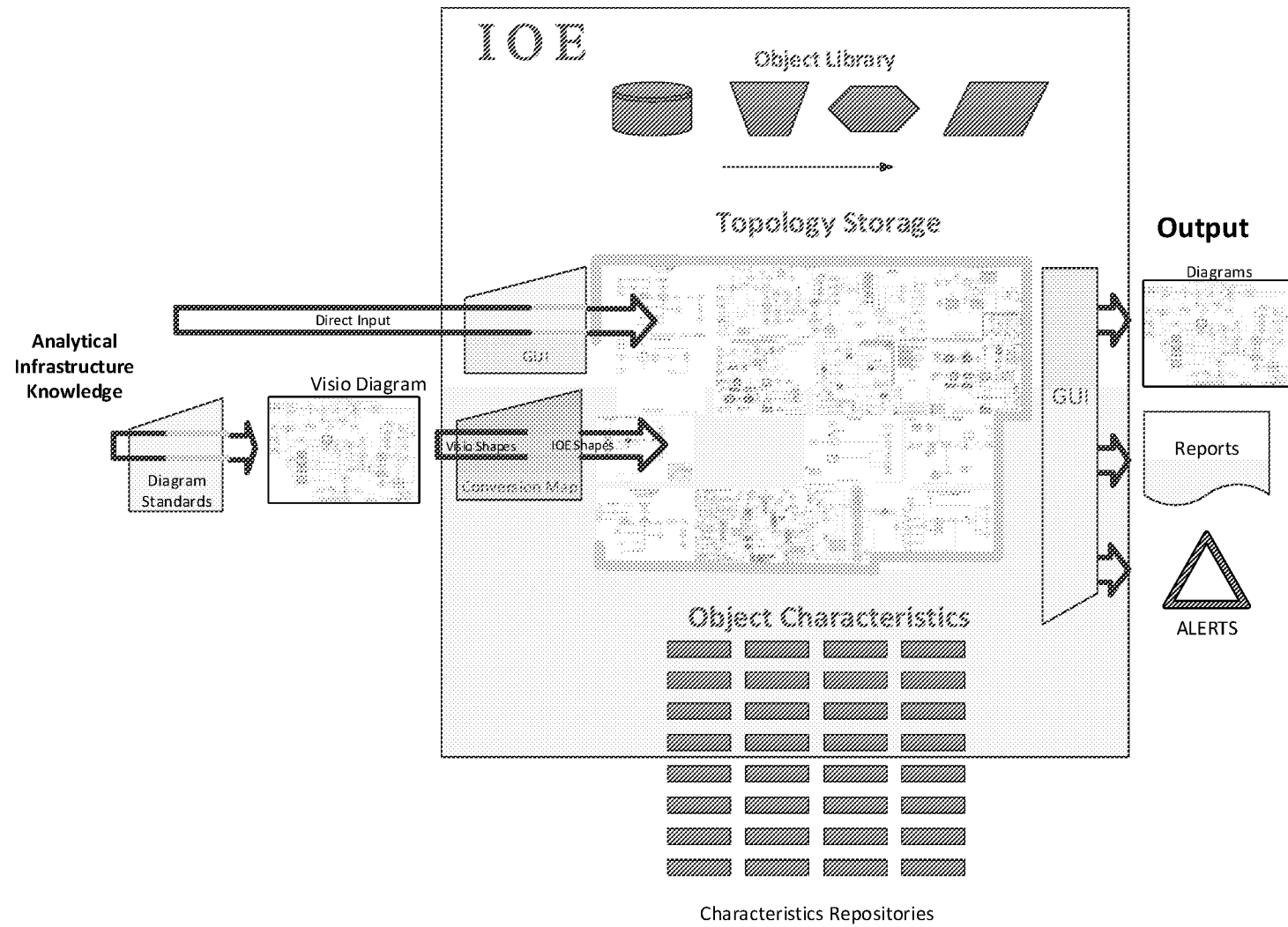
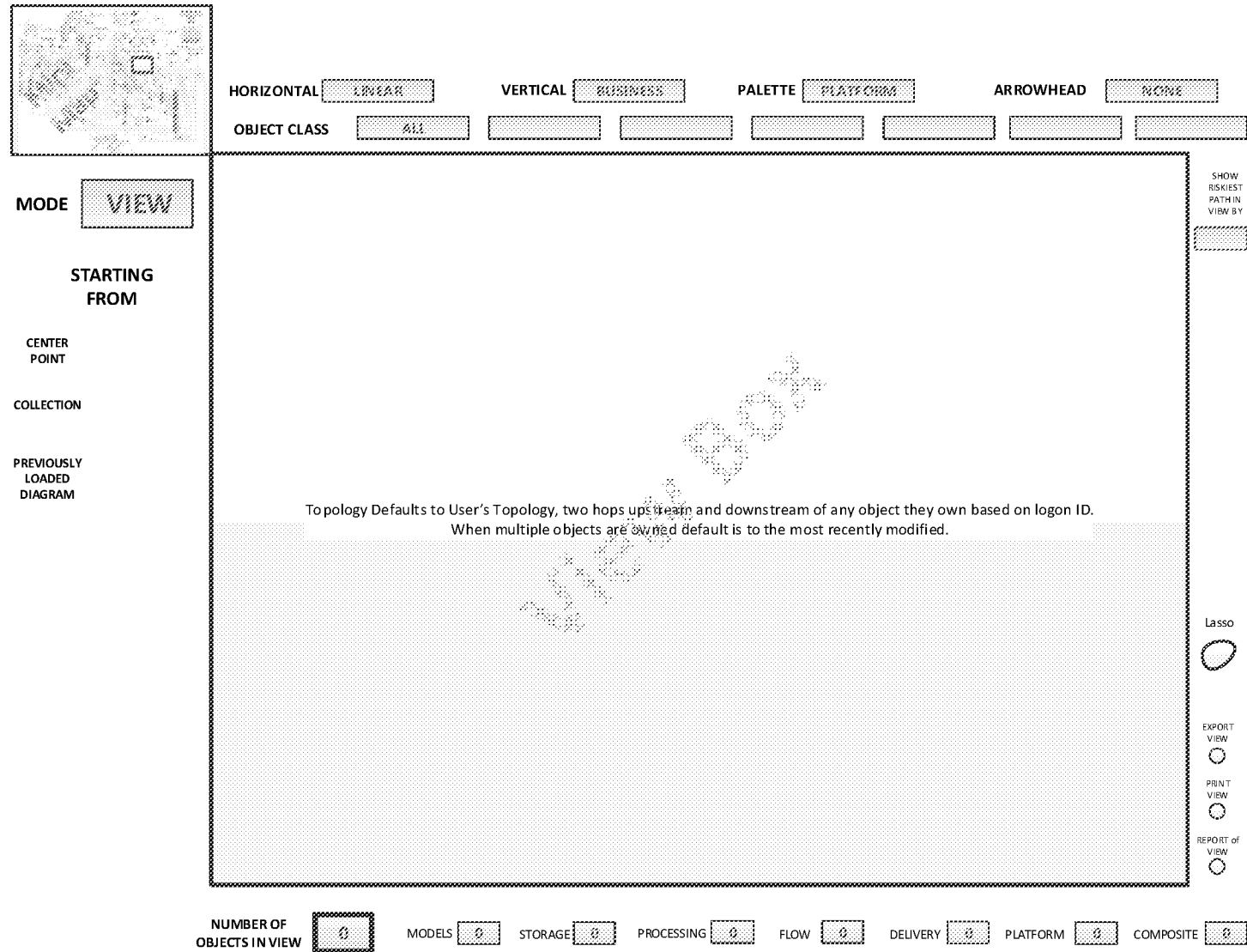


Figure 3.1 View model



Horizontal **Vertical** **Palette** **Arrowhead**

Object Class

MODE

STARTING FROM

CENTER POINT

COLLECTION

PREVIOUSLY LOADED DIAGRAM

Topology Defaults to User's Topology, two hops upstream and downstream of any object they own based on logon ID.
When multiple objects are owned default is to the most recently modified.

VIEW BOX

SHOW RISKIEST PATH IN VIEW BY

Lasso 

EXPORT VIEW ☐

PRINT VIEW ☐

REPORT of VIEW ☐

NUMBER OF OBJECTS IN VIEW **MODELS** **STORAGE** **PROCESSING** **FLOW** **DELIVERY** **PLATFORM** **COMPOSITE**

Figure 3.2 View mode showing drop down values for user-selectable boxes

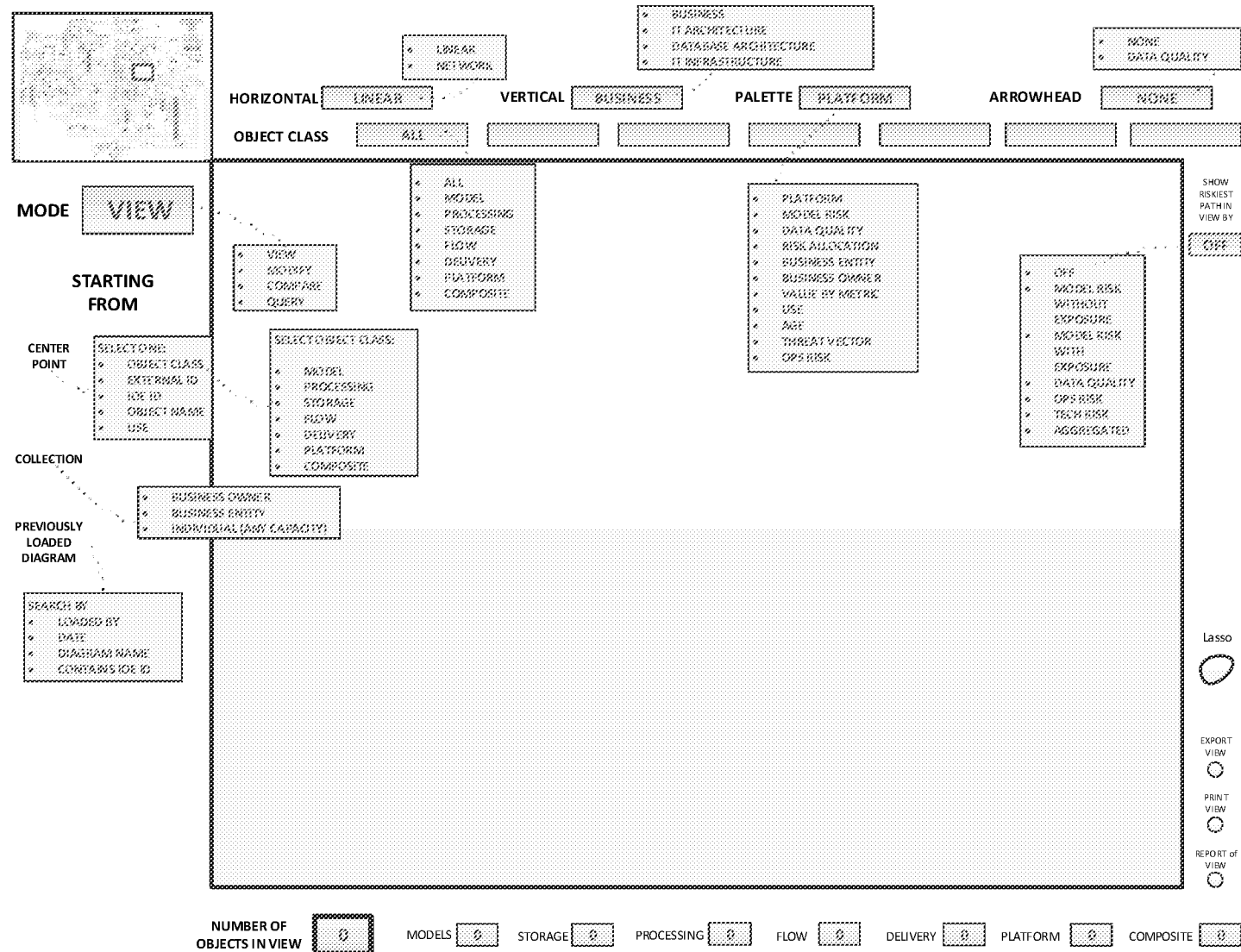


Figure 4.1 Horizontal View – Comparison of options

Horizontal View – Example

Same topology in Linear and Network View

(Depicted here where Vertical View = Business and Topological View = Infrastructure)

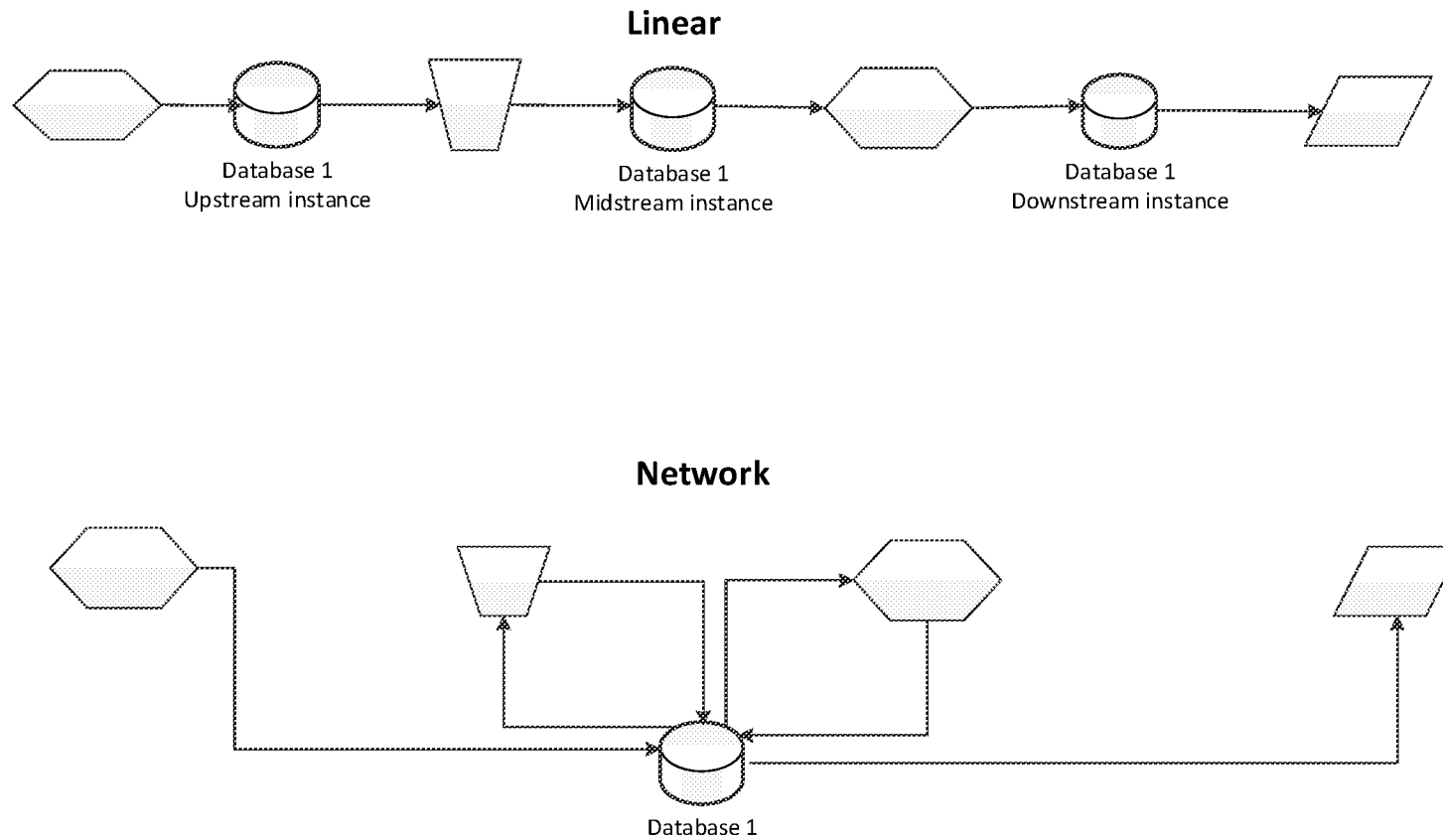


Figure 4.2 Cross-Section Schematic of Vertical View options

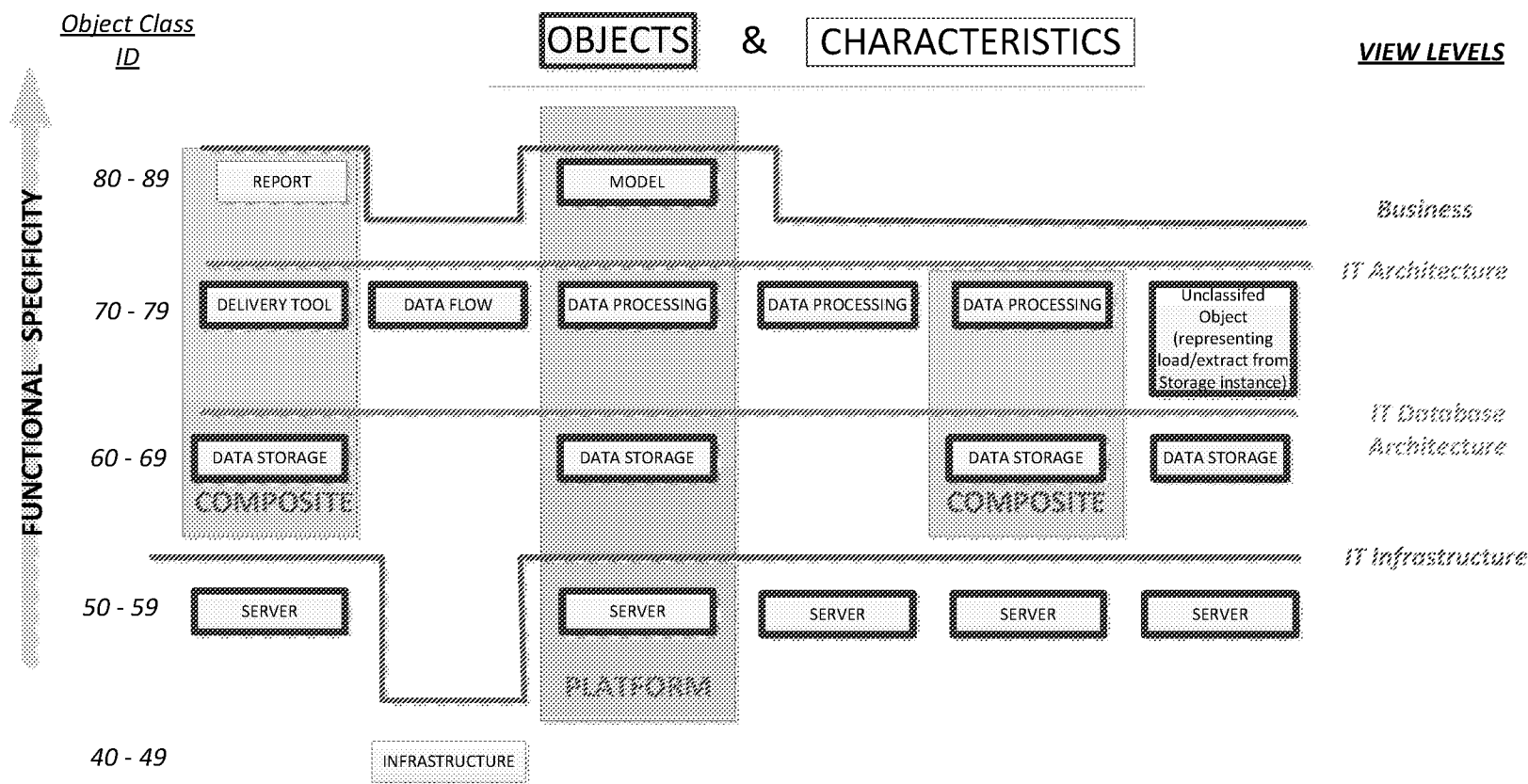


Figure 4.3 Vertical View - examples

Vertical View – Examples

A single topology presented in four vertical levels

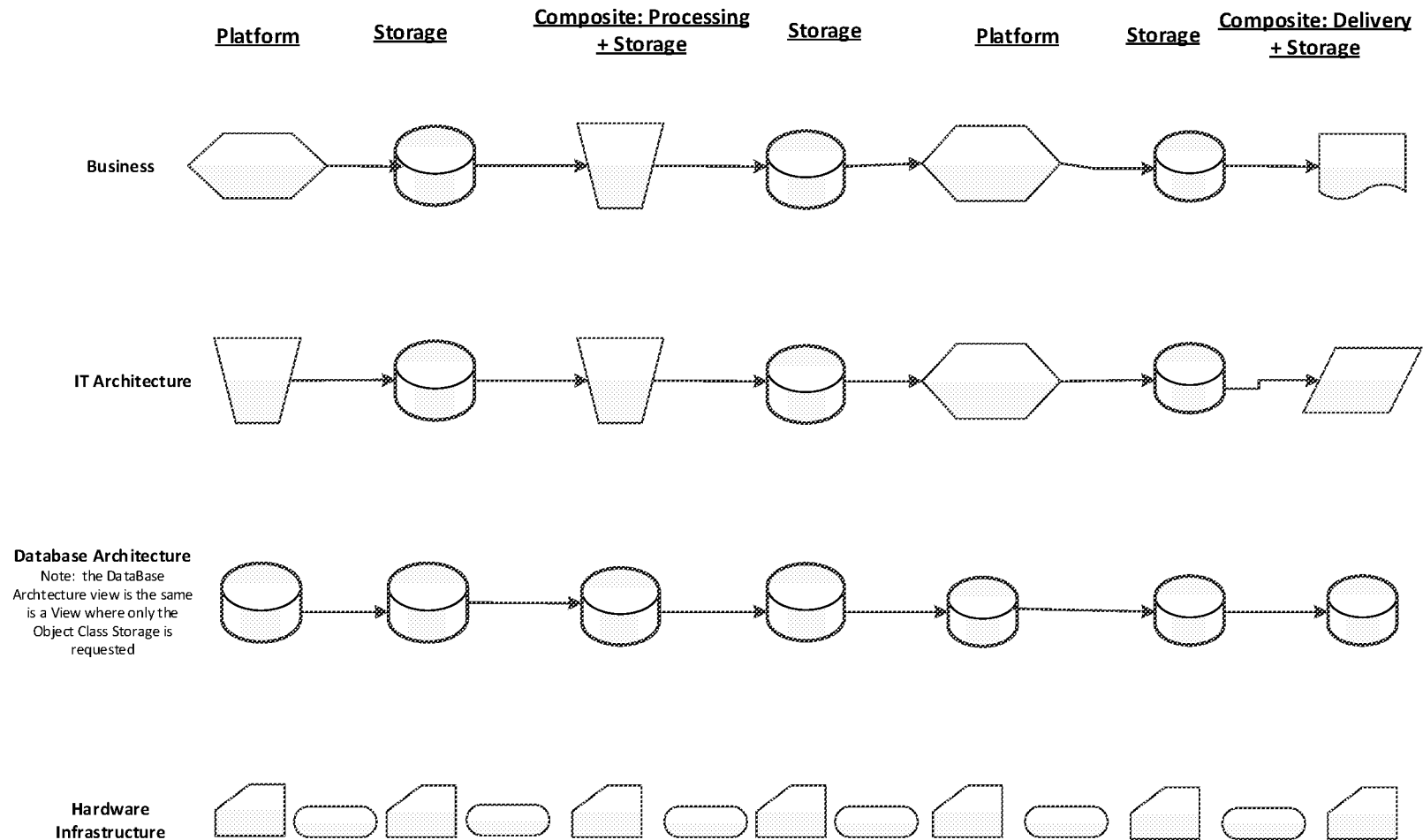


Figure 4.4 Examples of Palette View

Palette View – Examples

Same topology depicted using different Palette Views
(Depicted here where Vertical View = Business and Topological View = Infrastructure)

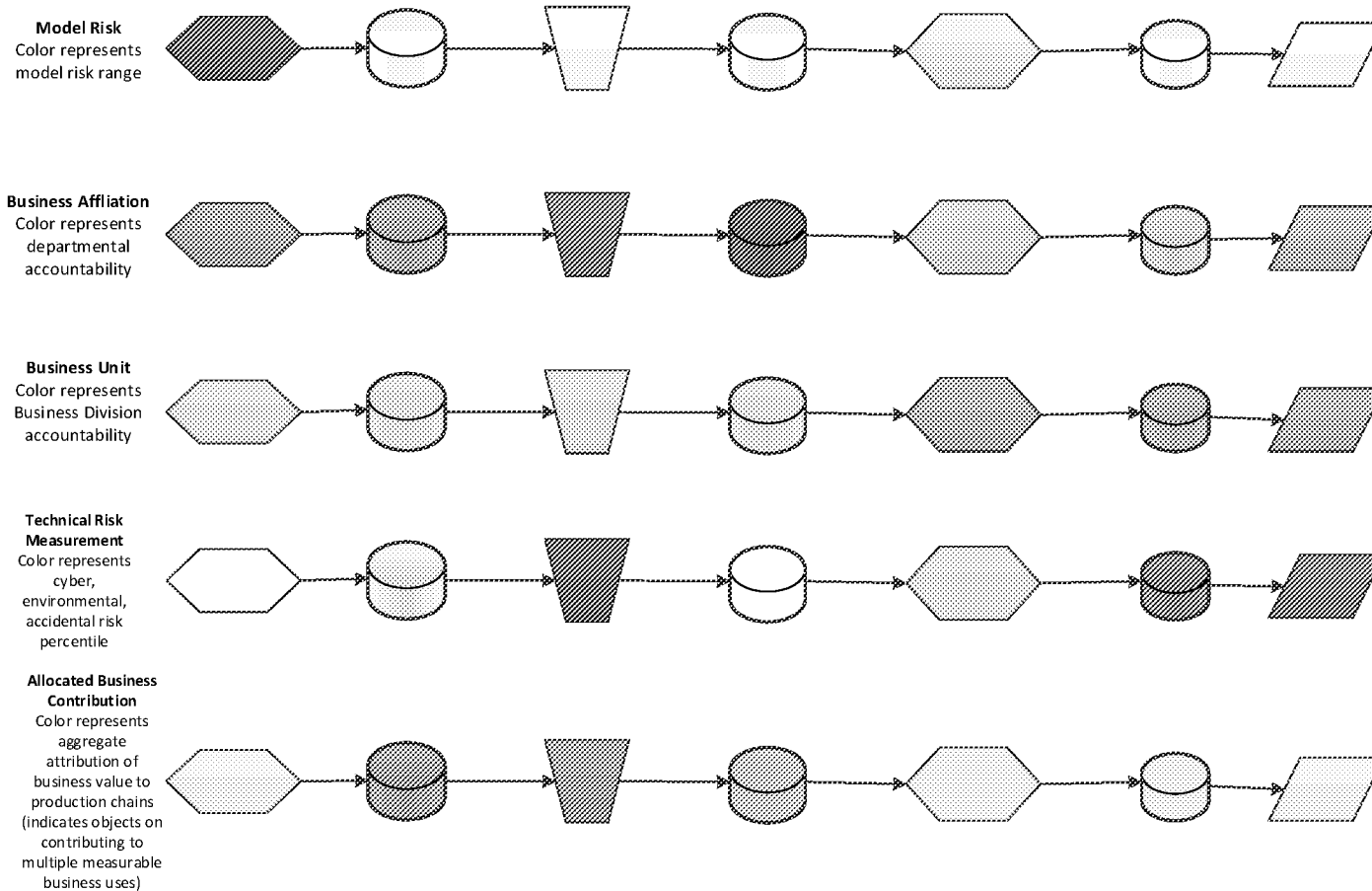


Figure 4.5 Object Class View - example

Object Class View – Example

Selected: Model and Delivery Tool Object Classes

(Depicted here where Vertical View = Business and Topological View = Infrastructure,
Palette view = Platform)

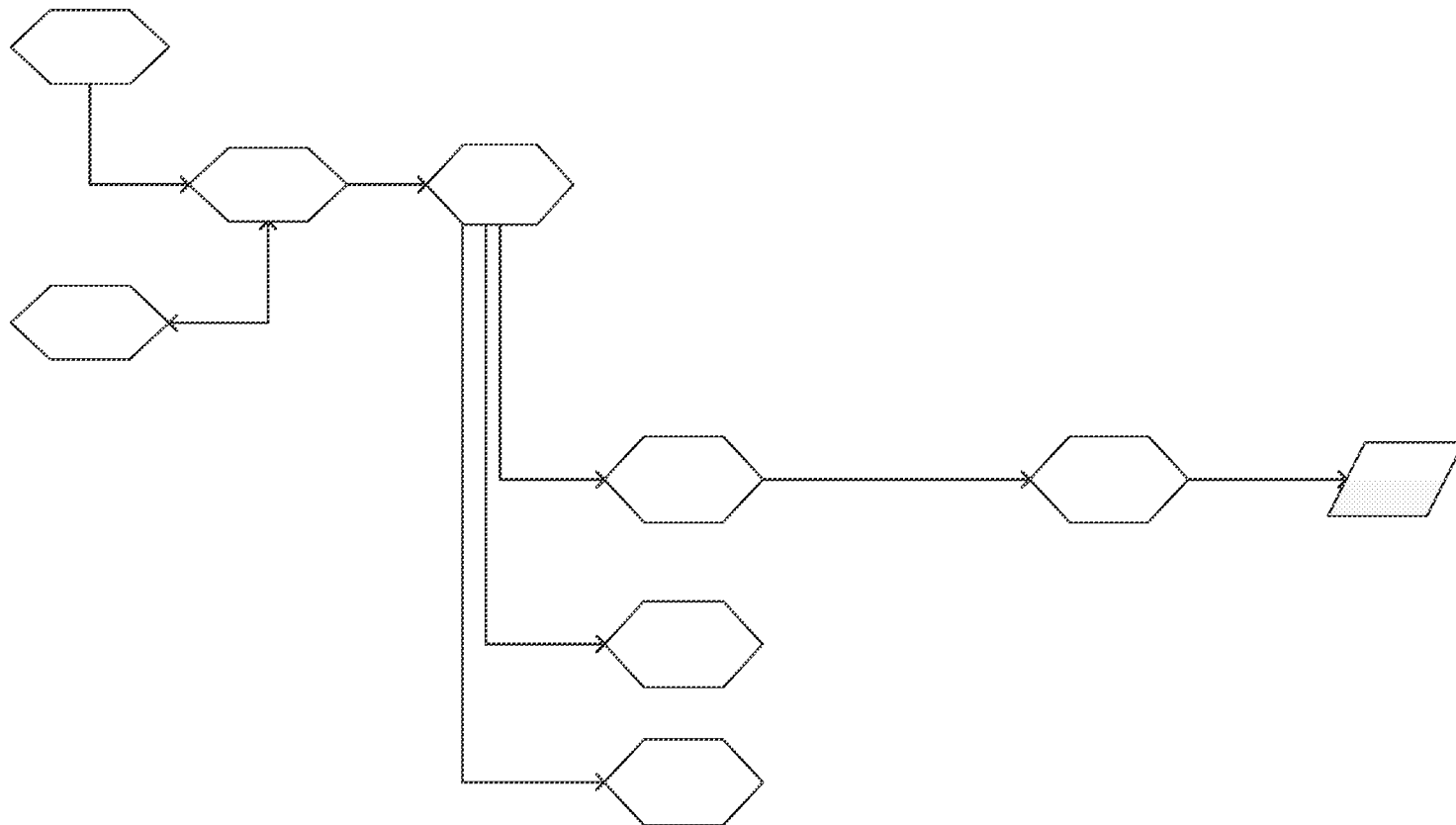


Figure 4.6 Path Highlight for a single Object (Model 1)

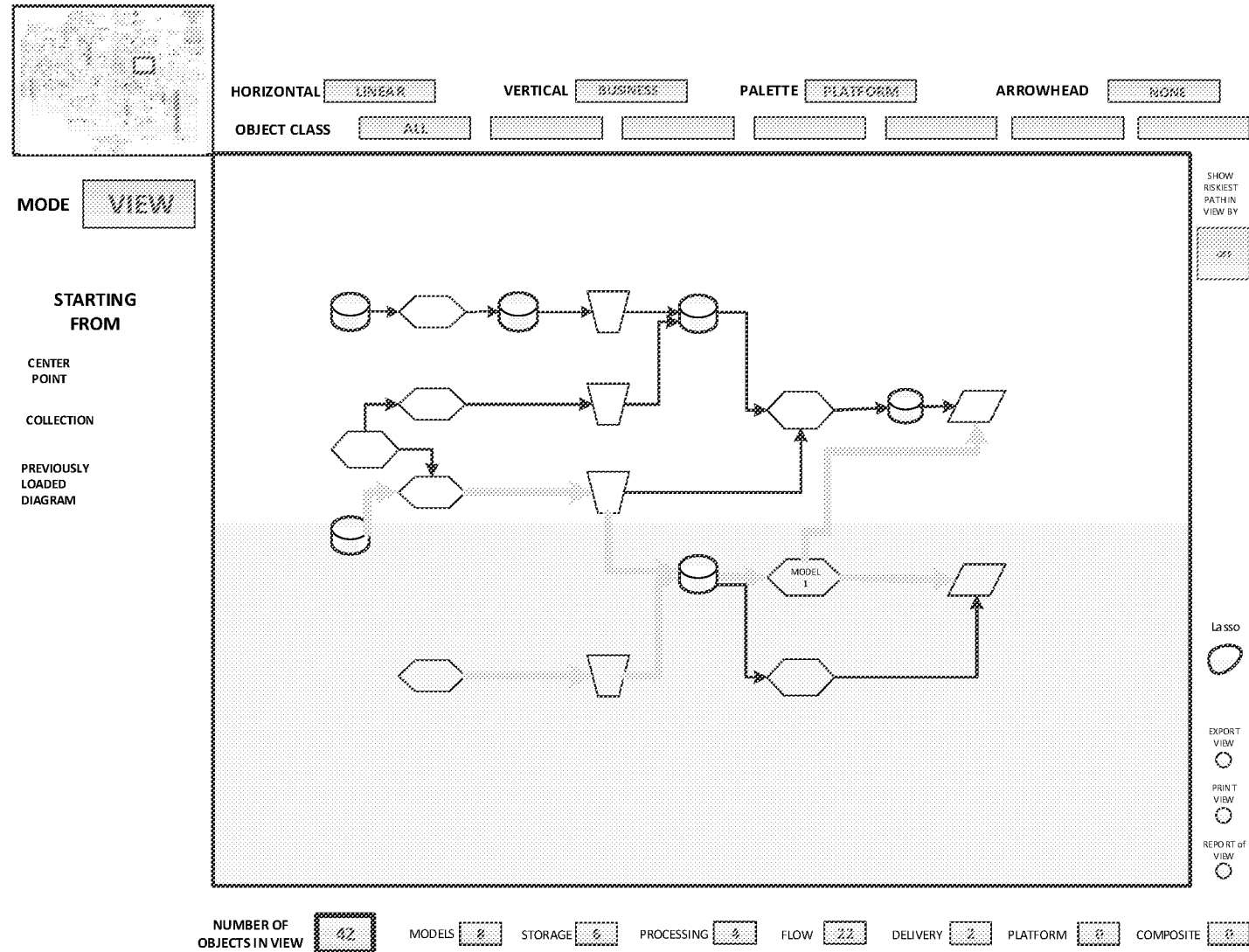


Figure 4.7 Screen Path Highlight for two Objects (Model 1 and Model 2)

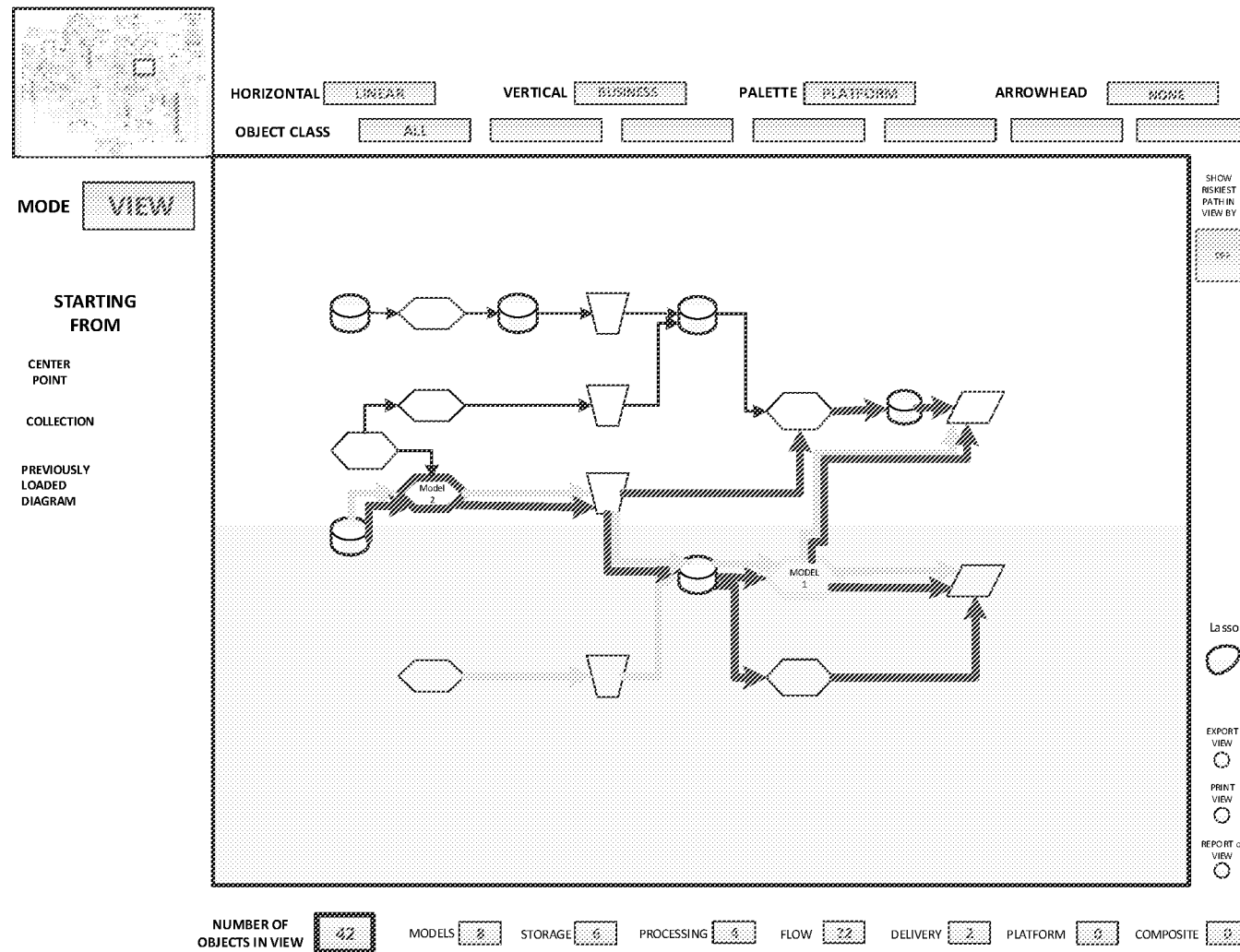


Figure 4.8 View with Riskiest Path

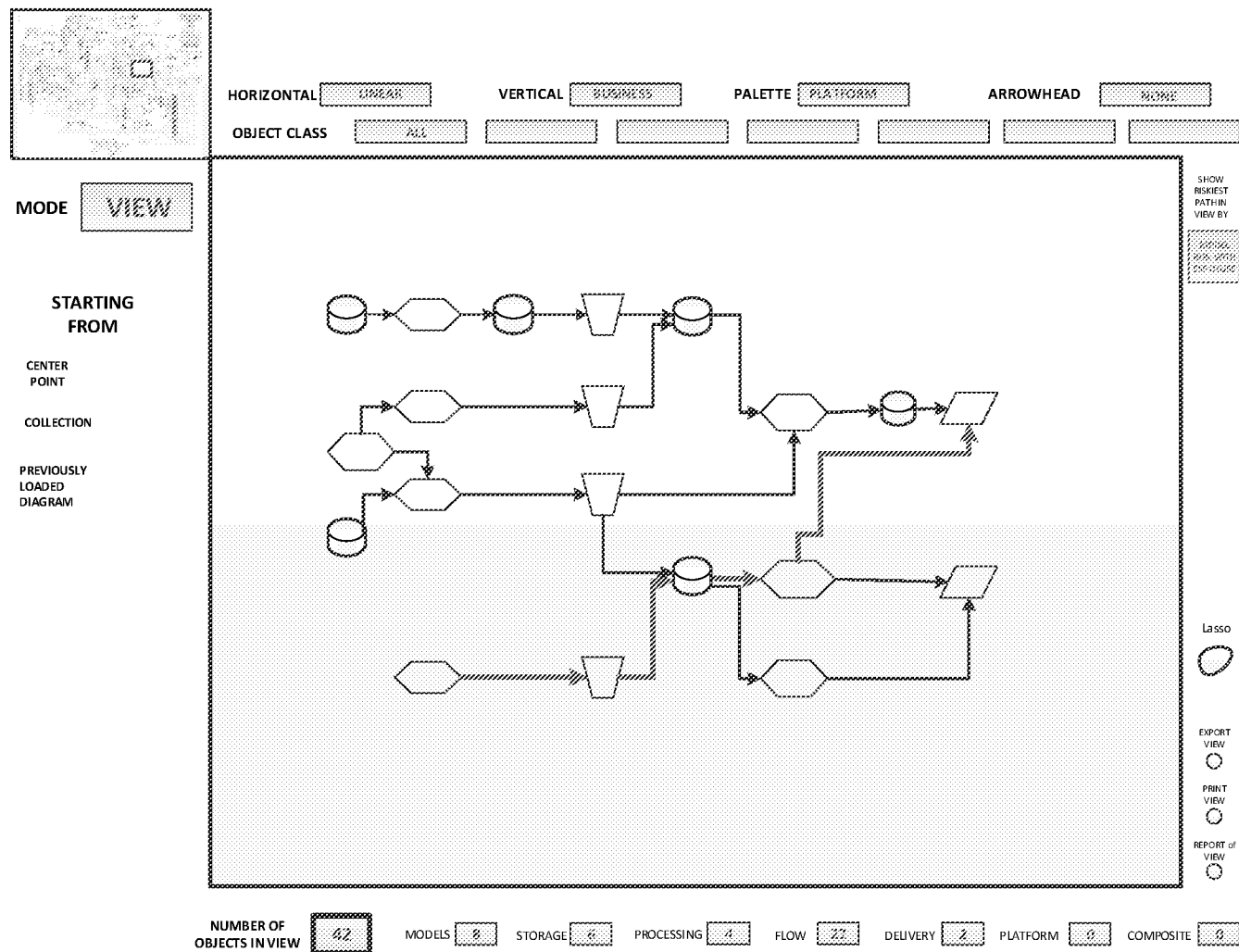


Figure 5.1 Modify Mode

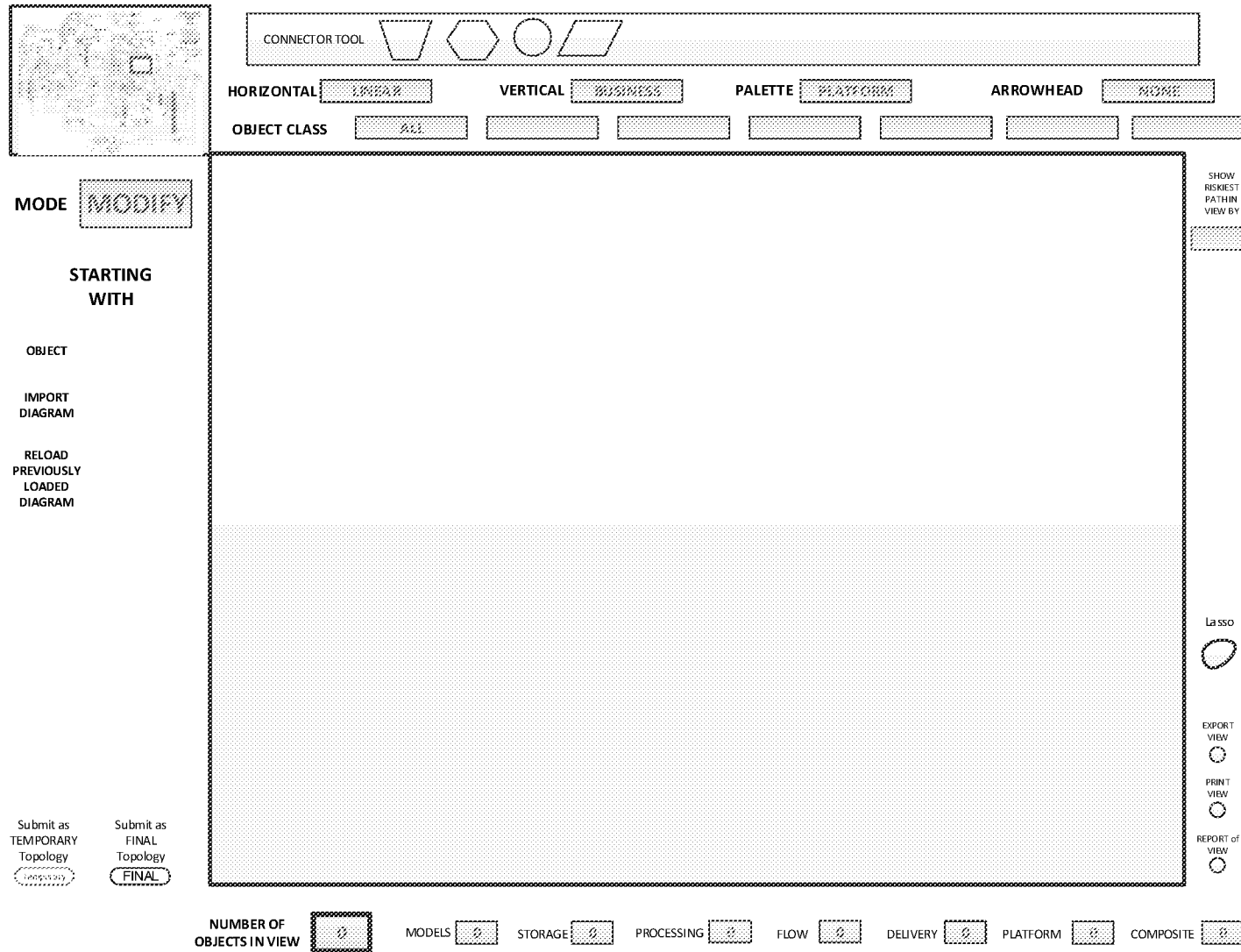


Figure 5.2 Modify mode showing drop down values for user-selectable boxes

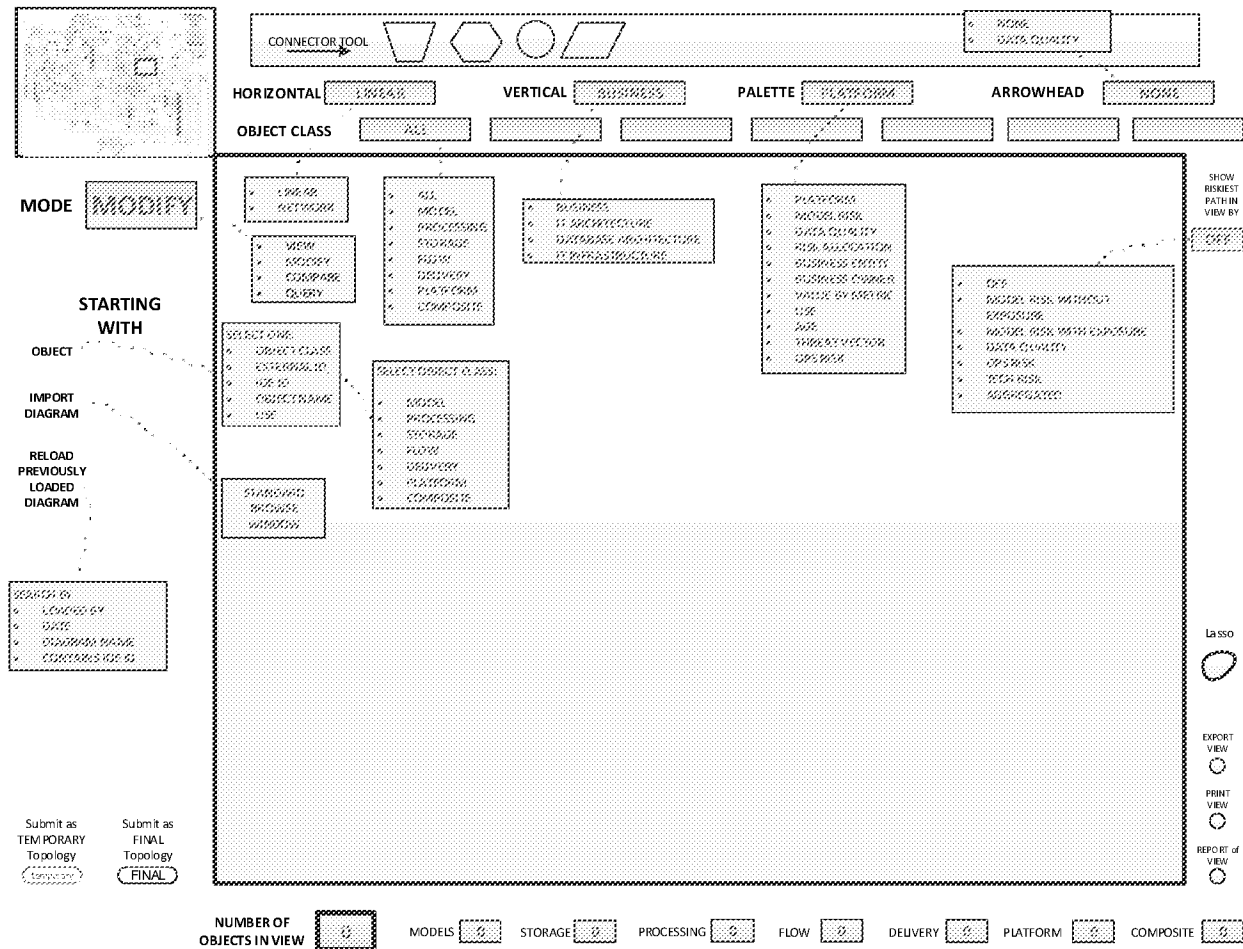


Figure 6.1 Compare mode Meld Method



Figure 6.2 Compare mode Twin Screen Method

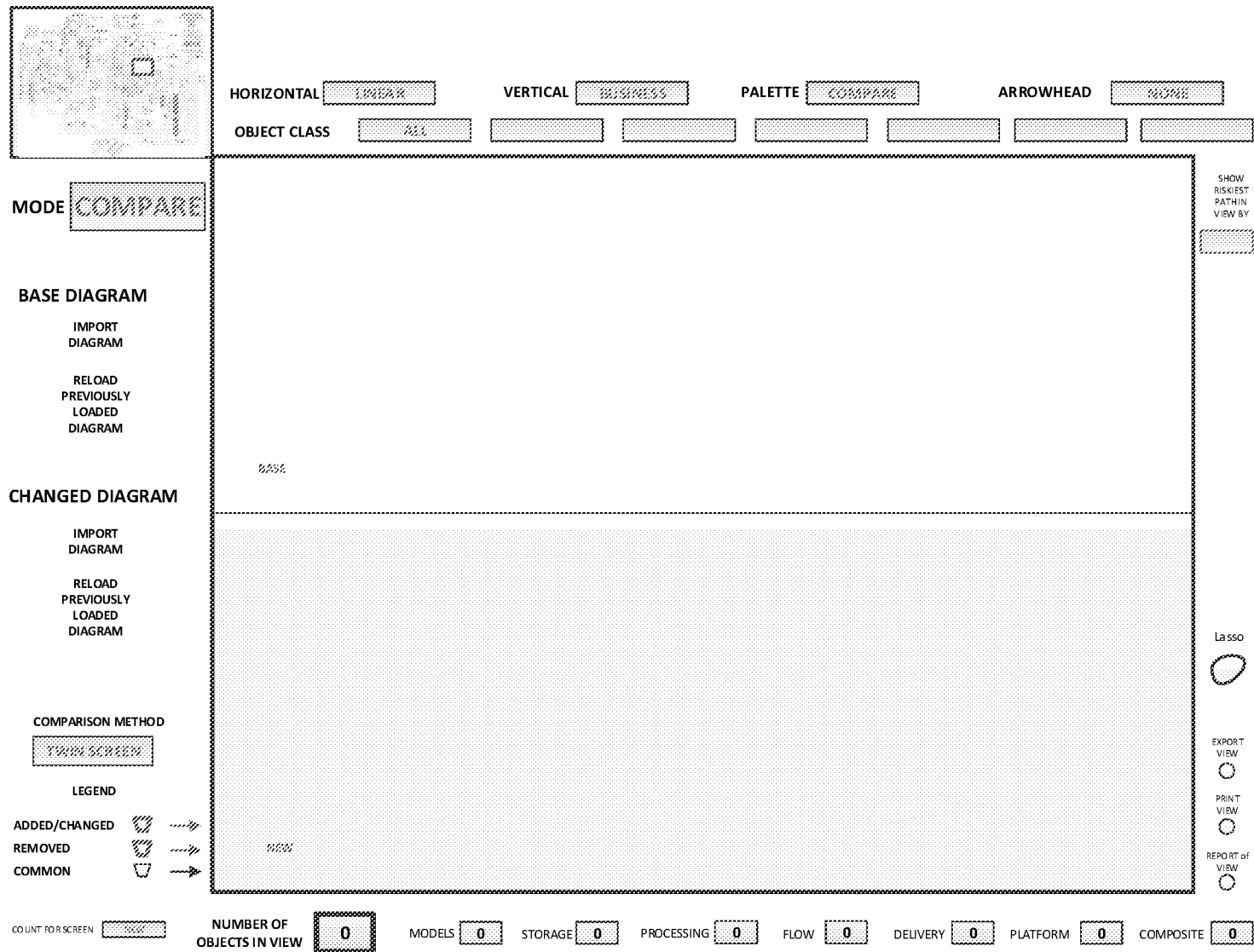
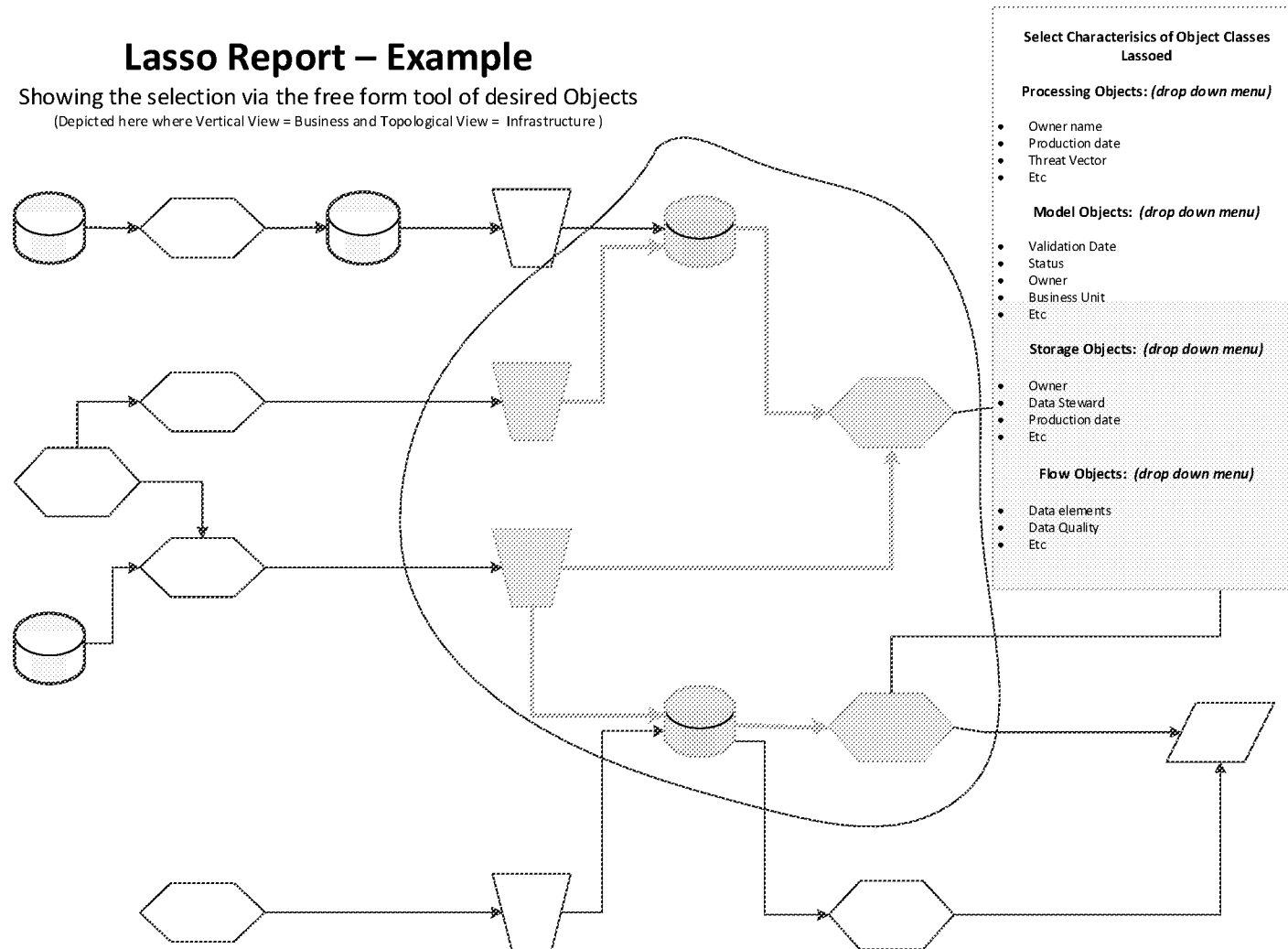


Figure 6.3 Lasso Report – example of selecting objects using the Lasso function



IOE: Representation of Infrastructure

MODEL CHARACTERISTICS

- Model Risk & Analytics
- Validation Status
- Exposure / Risk Score
- Purpose
- Ownership

PLATFORM CHARACTERISTICS

- Hardware Metadata
- Software Metadata
- Application Risk Metrics
- Development Timetables
- Ownership

DATA STORAGE CHARACTERISTICS

- Data Elements
- Data Quality
- Storage Metadata
- Ownership

CALC

Electronic Patent Application Fee Transmittal

Application Number:				
Filing Date:				
Title of Invention:	INTEGRATED OBJECT ENVIRONMENT			
First Named Inventor/Applicant Name:	Sean C. Keenan			
Filer:	Aaron Ross Feigelson			
Attorney Docket Number:				
Filed as Large Entity				
Filing Fees for Provisional				
Description	Fee Code	Quantity	Amount	Sub-Total in USD(\$)
Basic Filing:				
PROVISIONAL APPLICATION FILING	1005	1	260	260
Pages:				
Claims:				
Miscellaneous-Filing:				
Petition:				
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD(\$)
Extension-of-Time:				
Miscellaneous:				
Total in USD (\$)				260

Electronic Acknowledgement Receipt

EFS ID:	28973392
Application Number:	62487370
International Application Number:	
Confirmation Number:	8430
Title of Invention:	INTEGRATED OBJECT ENVIRONMENT
First Named Inventor/Applicant Name:	Sean C. Keenan
Customer Number:	23460
Filer:	Aaron Ross Feigelson
Filer Authorized By:	
Attorney Docket Number:	
Receipt Date:	19-APR-2017
Filing Date:	
Time Stamp:	18:35:11
Application Type:	Provisional

Payment information:

Submitted with Payment	yes
Payment Type	DA
Payment was successfully received in RAM	\$260
RAM confirmation Number	042017INTEFSW00005226121216
Deposit Account	121216
Authorized User	Aaron Feigelson

The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:

37 CFR 1.16 (National application filing, search, and examination fees)

37 CFR 1.17 (Patent application and reexamination processing fees)

IPR2023-00656 - IriusRisk

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Provisional Cover Sheet (SB16)	726658_Cover.pdf	86814	no	1
			89b6fd9d027273d87cc4a337ec021ebb6f9e ba2fd		
Warnings:					
This is not a USPTO supplied Provisional Cover Sheet SB16 form.					
Information:					
2	Specification	726658_ProvisionalApplication.pdf	383476	no	35
			d0ce04ad5f9365a8636e42b4d021d5c02ee fe86b		
Warnings:					
Information:					
3	Drawings-other than black and white line drawings	726658_FiguresB.pdf	925648	no	17
			69f3832326e03f8526d85655e425616e383 ab7c7		
Warnings:					
Information:					
4	Fee Worksheet (SB06)	fee-info.pdf	29198	no	2
			d306b5f936a653a42e7ed5530b7f831b111f 4793		
Warnings:					
Information:					
Total Files Size (in bytes):			1425136		

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for a filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application.

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.