# Proyecto Taller V

# Sistemas de Data Warehousing con Componente Geográfica

1998

*Inti Rebelese*

***Tutor: Raúl Ruggia***

# Indice

# 1 Introducción

Existen hoy en día diversos sistemas para el análisis de grandes volúmenes de información, utilizados fundamentalmente para la toma de decisones.
Estos son algunos de ellos:

Los **Sistemas de Data Warehousing** permiten resolver requerimientos de acceso a información de calidad para el análisis gerencial y la toma de decisiones. Para esto extraen datos de Base de Datos fuentes, los transforman y los presentan al usuario a través de diferentes herramientas Front-End que hace posible su análisis.

Los **Sistemas de Información Geográfica (GIS)** consisten en un conjunto herramientas para almacenar, consultar, y analizar datos espaciales. Tales sistemas no solo manejan datos de mapas e imágenes, pero también manejan registros y atributos de cualquier cosa que pueda estar relacionado a una ubicación geográfica. La tecnología es aplicable a diversas aplicaciones abarcando desde manejo de recursos a respuestas de emergencia y recuperación de desastres, desde estudios políticos a estudios forestales y marítimos, desde marketing masivo a manejo de infraestructura urbana, y desde estudios locales a través de análisis regional a investigaciones de cambios globales.

**Data Mining** es una tecnología diseñada para "escrudiñar" en grandes volúmenes de información, por información oculta, permitiendo que pueda usarse para incrementar los beneficios de una empresa. Las áreas de aplicación pueden ser muy diversas, marketing, negocios, sciencia, ...

El área de Data Warehousing es una de las más activas actualmente tanto a nivel académico como industrial. Una de las características más importantes que presenta es que se basa en la integración de soluciones técnicas más que en productos completos. A pesar del desarrollo del área, un punto poco explorado es el de la integración de herramientas GIS. Esto reviste de gran interés industrial, ya que permitiría resolver en forma más adecuada una importante gama de problemas, así como académico ya que concierne la resolución de problemas no triviales de integración de modelos de datos y herramientas.

# 2 Tecnologías involucradas

Las herramientas que serán estudiadas para su integración son:

## *2.1 Herramientas OLAP*

Las herramientas *OLAPs (On-Line Analytic Processing)* permiten representar los datos del problema en términos de *dimensiones*. Por ejemplo, si se trata de ventas de productos en diferentes zonas, una dimensión del problema son las zonas, otra los productos y otra el tiempo. De esta manera, las consultas de análisis de datos de una dimensión en función de la otra se realizan en forma inmediata. Los OLAPs pueden ser *servidores*, si almacenan los datos en este modelo dimensional; o *clientes* si se conectan a un servidor de base de datos (por ejemplo Relacional) y transforman los requerimientos dimensionales en términos relacionales. La utilización de unos u otros depende del tipo de problema a resolver, no existiendo un modelo que sirva para el 100% de los problemas.

### 2.1.1 PowerPlay (COGNOS)

#### 2.1.1.1 Qué es ?

En este proyecto la herramienta OLAP de interés, es PowerPlay de la empresa COGNOS.

Esta herramienta tiene como característica importante para nuestro interés, la posibilidad de funcionar como servidor OLE, permitiendo así la comunicación con otras aplicaciones a través de objetos OLE (Ver Apéndice).

#### 2.1.1.2 Funcionalidades de PowerPlay

##### *2.1.1.2.1 Explorar los indicadores de performance:*

Los indicadores de performance son las medidas que maneja cada reporte de PowerPlay.

##### *2.1.1.2.2 Ver los impactos en cada aspecto respecto a los resultados generales:*

Esto se debe a que cada aspecto importante está organizado en dimensiones. Juntas proveen una visión multidimensional de la realidad. Cada dimensión esta estructurada en niveles, recorriéndolos es que se puede analizar con exactitud distintos aspectos de la información.

*2.1.1.2.3 Cambio de visión:*

Otra posibilidad más de cambiar la perspectiva de la realidad, es el cambio de los aspectos que se contraponen en el reporte. Esto se logra cambiando las dimensiones que aparecen en las columnas o filas, por cualquier otra de las dimensiones.

*2.1.1.2.4 Ordenamiento de la información:*

La información que se presenta en un reporte puede ser ordenada tanto por columnas como por filas. El ordenamiento va a depender de que medida en particular se esté tomando en el momento de ordenar.

*2.1.1.2.5 Resaltado de la información de acuerdo a criterios deseados:*

Se pueden definir filtros para valores que están en ciertos rangos, permitiendo así resaltar dichos valores del resto.

## *2.2 Herramientas GIS*

Las herramientas GIS permiten almacenar y visualizar información geográfica, así como almacenar información asociada a la misma.
Estas herramientas ofrecen un alto nivel de análisis sobre los aspectos geográficos, pero también sobre la información que está asociada a estos. Por ejemplo: Mostrar aquellas regiones de un mapa donde el crecimiento de la población superó la media nacional, y que están en la zona costera del país.

### 2.2.1 MapObjects (ESRI):

#### 2.2.1.1 Qué es ?

En este proyecto la herramienta GIS de interés, es MapObjects de la empresa ESRI.

MapObjects es un conjunto de componentes de software, ActiveX, que permiten agregar mapas a las aplicaciones y poder realizar distintas operaciones sobre los mismos.

#### 2.2.1.2 Objetivos de MapObjects

- Ofrecer ciertas funcionalidades de una poderosa herramienta GIS como es ArcView (ESRI), en un conjunto de componentes. Esto permite a los desarrolladores disponer de las herramientas para combinar los componentes GIS, en la forma adecuada para construir una aplicación específica.
- Costo inferior en adquirir el producto MapObjects que el paquete ArcView.

#### 2.2.1.3 Funcionalidades básicas

Se pueden implementar estas y otras funciones en programas construidos con MapObjects:

- Desplegar un mapa con múltiples layers, tales como carreteras, ríos y fronteras.
- Zoom sobre un mapa.
- Dibujar gráficos tales como puntos, líneas, círculos y polígonos.
- Escribir texto descriptivo.
- Identificar características sobre un mapa apuntando sobre el mismo.
- Seleccionar características a través de líneas y dentro de cajas, áreas, polígonos y círculos.
- Seleccionar características a una distancia específica de otras.
- Seleccionar características con una expresión SQL.
- Calcular estadísticas básicas sobre características seleccionadas.

- Consultar y modificar datos asociados con características seleccionadas.
- Desplegar imágenes de fotografía aérea o imágenes satelitales.
- Dinámicamente desplegar datos en tiempo real o serializados en el tiempo.
- Dada una dirección encontrar un punto en el mapa.

#### 2.2.1.4 Funcionalidades no ofrecidas

- Despliegue de cartografía de alta calidad.
- Proyecciones de sistemas de coordenadas.
- Sofisticados análisis espaciales, tales como modelado superficial o edición topológica.

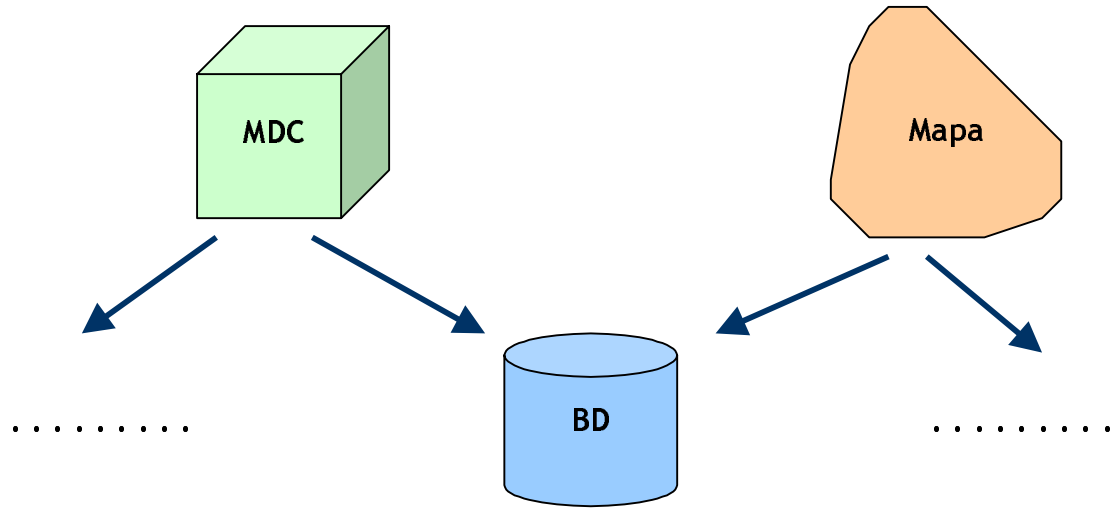
# 3 Formas de integración

## *3.1 Posibles niveles de integración*

Existen distintos niveles de integración entre las herramientas GIS y las herramientas OLAP, de acuerdo a sí solo comparten información en común, o si comparten información y funcionalidades.

## *3.2 Integración a nivel de los datos*

La integración solo a nivel de los datos, implica que parte o toda la información que está en un cubo multidimensional, y aquella de origen geográfico y la relacionada con ella, es la misma. Pero las herramientas que operan en cada caso, no tienen ningún tipo de integración, no pudiendo sacar provecho de la combinación de sus funcionalidades. Cada una de las herramientas trabaja aisladamente sobre un conjunto de datos en común.



### 3.2.1 Como se manejan los datos en un cubo multidimensional

Un cubo multidimensional puede ser generado a partir de diversas fuentes de datos, dependiendo de la herramienta que se utilice para ello.

Se debe definir:

- Origen de los datos (BD, o consultas definidas sobre una BD).
- Definición de las dimensiones que formarán el cubo.
- Definición de los niveles de cada una de las dimensiones.
- Definición de las medidas del cubo.

| Características |
|---|
| Repositorio de datos no modificable. |
| Alto nivel de acceso a los datos. |
| Regeneración del cubo para refrescar datos. |

### 3.2.2 Como se manejan los datos geográficos

Así mismo las herramientas de manejo de bases de datos geográficas, además de manejar mapas cartográficos, permiten relacionar información distinta a aquella de origen puramente geográfico.

Un sistema GIS está formado por:

- Layers, o capas, a partir de los cuales se arman los mapas.
- Información asociada a los aspectos geográficos. Normalmente se almacenan en una base de datos asociada.

| Características |
|---|
| Operaciones específicas sobre la información asociada a los aspectos geográficos.<br><br>***Por ejemplo:*** Ubicación de ciertos puntos de un mapa que cumplan ciertas condiciones de acuerdo a ciertos valores asociados. |
| Operaciones sobre los aspectos geográficos.<br><br>***Por ejemplo:*** Ubicación de ciertos puntos de un mapa que están a cierta distancia de una recta. |

**En MapObjects:**

- La información asociada con la geografía se almacena en una base de datos de tipo DBase (.dbf).
- Existe una tabla asociada por cada uno de los layers.
- Las operaciones se realizan sobre los layers , por lo cual no permite realizar operaciones con información que no se encuentre en las tablas asociadas.
- Existen operaciones para agregar datos a un layer, agregandola a la tabla asociada.

### 3.2.3 Niveles de integración de los datos

Se pueden definir distintos niveles de integración de los datos que se encuentran en cubo multidimensional, y los que se encuentran en una base de datos geográfica:

1. Integración de parte o de todas las dimensiones de un cubo a la base de datos geográfica.
2. Integración de los datos geográficos al cubo multidimensional.

#### 3.2.3.1 Integración de dimensiones a la base de datos geográfica

Como hemos visto, cada uno de los layers que forman parte de un mapa tienen una única tabla asociada a ellos. La integración de dimensiones a la base de datos geográfica implicaría:

- Integrar las mismas en cada uno de los layers, lo cual llevaría a integrarlas con cada una de las tablas asociadas a los layers.
- Representar la estructura jerárquica de una dimensión en una única tabla.

| Desventajas |
|---|
| No se puede representar la estructura jerárquica de una dimensión en una única tabla.<br>Se podrían manejar más tablas, lo cual implica que se deba implementar una aplicación para acceder a dicha jerarquía. |
| **Principalmente**: No es conveniente pasar la información almacenada en un data warehouse, a un modelo relacional, ya que se pierde las ventajas de la existencia de un data warehouse. |

**Conclusión** ⇒ Estas desventajas dan la pauta de la necesidad de soluciones a nivel de aplicación para acceder a datos ubicados en un data warehouse desde herramientas GIS, pudiendo así usar dicha información para su tratamiento a nivel geográfico.

### 3.2.3.2 Integración de los datos geográficos al cubo multidimensional.

Otra posibilidad de integración de los datos es definir en el cubo mutlidimensional una o más dimensiones que representen la geografía. Estas dimensiones deben estar relacionadas con el resto de las dimensiones del cubo.

**Objetivo:** La definición de dimensiones en un data warehouse permitirían tener como aspecto importante para el análisis de la información asociada, la ubicación geográfica.

**Ejemplo:** Si tenemos un data warehouse que contiene dimensiones que representan productos vendidos, vendedores, clientes, épocas del año, etc, podría ser de gran interés agregar una dimensión que represente zonas de venta, agregando un aspecto más para el análisis de la información.

La herramienta GIS con la que estamos trabajando maneja tres tipos de aspectos geográficos que pueden ser representados en un mapa:
(ver Tecnologías involucradas)

- **Polígonos:** Permiten representar regiones, lagos, ...
- **Líneas:** Permiten representar rutas, líneas férreas, ríos, ...
- **Puntos:** Permiten representar ciudades, ...

Cada uno de estas aspectos tiene definidos en las bases de datos geográficas un identificador geográfico usado internamente por la herramienta GIS, por lo cual puede ser necesario que se definan identificadores asociados a la realidad para ser usadas como categorías en las dimensiones a definir.

*3.2.3.2.1 Características geográficas de cada uno de los aspectos en MapObjects:*

| Aspecto Geográfico | Características Gegográficas |
|---|---|
| **Polígono** | Pueden contener cualquiera de los demás aspectos geográficos.<br>Pueden contener a otros polígonos. |
| **Recta** | Pueden haber puntos que pertenezcan a las mismas.<br>No contienen a otras rectas. |
| **Punto** | No contiene a ningún otro aspecto geográfico. |

*3.2.3.2.2 Características geográficas en el data warehouse:*

Cada una de estas características geográficas deben ser tomadas en cuenta en el momento de definir las dimensiones que representen la geografía:

| Aspecto Geográfico | Características en la dimensión |
|---|---|
| **Polígono** | Pueden haber más de un nivel con categorías asociadas a este tipo de aspecto geográfico.<br>Los niveles se ubicarán siempre en los niveles inferiores de la dimensión. |
| **Recta** | Puede existir solo un nivel con categorías asociadas a este tipo de aspecto geográfico.<br>Puede ser un nivel inferior, o superior a niveles asociados a polígonos. |
| **Punto** | Puede existir solo un nivel con categorías asociadas a este tipo de aspecto geográfico.<br>Puede ser un nivel inferior. O superior a niveles asociados a polígonos, o a niveles asociados a rectas. |

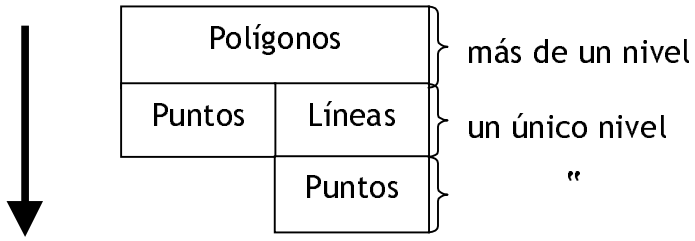
*3.2.3.2.3 Posibles definiciones de dimensiones geográficas:*

Existen distintas posibilidades de definir las dimensiones geográficas:

1) Una única dimensión que represente todas los aspectos geográficos de interés.
2) Una dimensión para cada una de los aspectos geográficos de un mapa.
3) Algunas de los aspectos geográficos podrían formar parte de los niveles superiores de una dimensión y otras representar una dimensión a parte.

1) Se define una única dimensión la cual, dependiendo de las formas geográficas presentes, representa toda la geografía.

Las categorías asociadas a las formas geográficas se podrían ordenar en la dimensión, de los niveles inferiores a superiores, de la siguiente manera:




**Características:**

- Los polígonos se ubicarían en los niveles inferiores ya que las demás formas geográficas podrían estar contenidas en ellos, o sea, serían categorías de los niveles superiores de la dimensión.
- Los polígonos podrían estar representados en más de un nivel, en el caso de que se tengan polígonos contenidos en otros.
- En el caso de las líneas y los puntos, solo se representarían en un único nivel ya que no se contienen a sí mismos.
- Se podría dar que en la geografía se tengan puntos que estén sobre algunas líneas, lo cual en la dimensión se representaría como un nivel superior a las líneas, donde se tienen aquellas categorías que se corresponden con dichos puntos.

**Ventajas**

Tomar en cuenta como más prioritarias las categorías asociadas a aspectos geográficos del tipo polígono, y luego las categorías asociadas a rectas, y puntos. Ya que Este enfoque es también usado en las herramientas GIS, en el momento de la visualización.

***Por ejemplo:*** En vez de visualizarse todas las regiones, rutas y ciudades de un mapa que podría llevar a que la visualización sea confusa, solo se muestran las rutas y ciudades cuando se hace un acercamiento a una región específica. Lo mismo se podría reflejar en la dimensión geográfica con respecto a las categorías.

2) Se define una dimensión por cada uno de los aspectos geográficas.

**Características:**

- La dimensión que representa los polígonos, podría tener más de un nivel.
- La de los apectos geográficos de tipo líneas y puntos tendrá un único nivel.

| Polígonos | Líneas | Puntos |
|---|---|---|
| más de un nivel | un único nivel | un único nivel |

**Ventajas**

Permite que se puedan seleccionar para analizar, cada uno de los aspectos representados en forma separada.

**Desventajas**

Seleccionar las categorías incorrectas, en cualquiera de las dimensiones geográficas, que no se correspondan con la realidad, produce que no se obtengan valores por no haber intersección.

***Por ejemplo:*** Seleccionar una ciudad que no pertenece a la región seleccionada.
Las dimensiones relaciondas a los aspectos geográficos de tipo punto y recta, al tener un único nivel, deben tener categorías únicas.
El análisis queda restringido al aspecto geográfico de menor "jerarquía".

***Por ejemplo:*** Si existe una dimensión para ciudades y otra para regiones, al tener siempre una categoría seleccionada en una dimensión, los valores que se pueden analizar siempre están relacionados a una ciudad. No pudiendo analizar las regiones en su totalidad.

3) Combinación de las dos definiciones anteriores.

**Caso 1:**

- Definir una dimensión que tenga en los niveles inferiores las categorías asociadas a los polígonos. En los niveles superiores las categorías asociadas a los puntos o las líneas.
- Definir otra dimensión que se corresponda con las líneas o puntos, dependiendo de cómo se haya definido la otra dimensión.

| | | | |
|---|---|---|---|
| Polígonos | más de un nivel | Puntos | un único nivel |
| Líneas | un único nivel | | |

**Caso 2:**

- Definir una única dimensión para los polígonos.
- Una dimensión que tenga en los niveles inferiores las categorías asociadas a las líneas y en los niveles superiores aquellas asociadas a los puntos.
- En la segunda dimensión, pueden existir categorías asociadas a los puntos en los niveles inferiores si no tienen ningún tipo de relación con las categorías asociadas con las líneas.
  Esta dimensión tendrá dos niveles como máximo, el primero para las categorías asociadas a las líneas y puntos, y el segundo para las categorías asociadas a los puntos que tienen relación con aquellas asociadas a las líneas.

| | | | | |
|---|---|---|---|---|
| Polígonos | más de un nivel | Puntos | Líneas | un único nivel |
| | | | Puntos | " |

| **Ventajas** |
|---|
| Combina las ventajas de las dos definiciones anterioes. |
| **Desventajas** |
| Las mismas que la definición anterior. |

Son posibles más casos de combinación de acuerdo a como se combinen los aspectos geográficos.

## *3.3 Integración a nivel de aplicación*

### 3.3.1 Objetivo

Integrar y combinar las funcionalidades que ofrecen las herramientas OLAP con las que ofrecen las herramientas GIS, y viceversa.

**Características básicas de las herramientas para la integración:**

**Herramientas GIS:**

- Interfase geográfica, que es una interfase apropiada para ser usada para filtrar las categorías de un data warehouse relacionadas a aspectos geográficos.
- Análisis espacial que puede ser complementario al análisis de información que se obtiene a través de las herramientas OLAP.

**Herramientas OLAP:**

- Acceso a información almacenada en un data warehouse para ser mostrada, de disintas formas, en una interfase geográfica. Permitiendo acceder, a una herramienta GIS, a un repositorio de información de mayor nivel de análisis que los repositorios usados por estas herramientas.
- Permitir que el análisis a través de las operaciones de las herramientas OLAP, sea usado a nivel geográfico.

⇒ Se logra el desarrollo de aplicaciones más potentes al ofrecer nuevas funcionalidades que no son posibles con las herramientas por separado.

### 3.3.2 Nuestro caso de estudio

La integración de las herramientas que se tratará es aquella que hace uso de componentes que ofrecen PowerPlay y MapObjects para integrar en un aplicativo desarrollado en determinado lenguaje:

| Herramienta | Características |
|---|---|
| **PowerPlay** | Servidor OLE. |
| **MapObjects** | Componentes ActiveX. |

**Observación:** Ninguno de estos componentes ofrecen el total de las funcionalidades posibles que tienen cada una de estas herramientas.

### 3.3.3 Condiciones para la integración

Para la integración de las funcionalidades de ambas herramientas, es necesario un mínimo de integración a nivel de los datos:

1. Definir las dimensiones asociadas con los aspectos geográficos en el data warehouse de acuerdo al análisis que se quiera realizar. (Ver integración a nivel de datos)
2. Definir un layer por cada uno de los niveles definidos en las dimensiones geográficas.
3. Relacionar los identificadores usados en la herramienta GIS con las categorías usadas en la herramienta OLAP.
4. Relacionar los layers definidos y sus aspectos geográficos, de acuerdo a la estructura de las dimensiones geográficas.

**En el data warehouse tendremos:**

- Dimensiones que representan los aspectos geográficos, donde cada categoría es el nombre asignado a cada uno de estos aspectos en la realidad.

- La cantidad de dimensiones existentes sobre los aspectos geográficos depende del modelo de data warehouse que se quiera construir, y esto no debería afectar el relacionamiento con los componentes GIS.

- La forma en que estén estructuradas las dimensiones geográficas debe ser tomada en cuenta, ya que afecta las operaciones que deberán realizarse sobre el data warehouse para obtener los valores necesarios en los componentes geográficos, y viceversa sobre los componentes GIS.

**En las bases de datos geográficas tendremos:**

- *En las tablas asociadas a los layers el mapeo del identificador interno con las categorías del data warehouse.*
  En el caso que estos identificadores difieran con los nombres de las categorias, se deberá tener un campo más en la tabla que contenga los nombres de dichas categorías.

- *Los layers deben corresponderse a los niveles en las dimensiones geográficas.*
  Como hemos mencionado anteriormente (ver integración a nivel de datos), pueden existir más de un nivel para los aspectos geográficos de tipo polígono, en una dimensión geográfica. De acuerdo a la cantidad de niveles existentes para las zonas deberá existir un layer correspondiente. Además cada uno de los layers que representan a los niveles superiores deben ser iguales al que representa el primer nivel, pero cada uno con mayor nivel de detalle de acuerdo al nivel que representan.

- *Relcionar los layers, de acuerdo al nivel que representan en las dimensiones geográficas.*

  Esto implica que deba haber en las tablas asociadas a dichos layers, un campo que tenga el identificador geográfico del aspecto geográfico al que pertenecen en el layer del nivel superior. Para el layer correspondiente al primer nivel no será necesario que exista dicha información.

Este es el mínimo de información necesario que hay que agregar en los layers, ya que el resto de la información se encuentra en el data warehouse, y es a través de las operaciones de los componentes OLAP que se accederán a ellos.

### 3.3.4 Interacción entre los componentes

Hay dos sentidos posibles de interacción entre los componentes OLAP y GIS:

1. Operaciones desde los componentes OLAP hacia los GIS.
2. Operaciones desde los componentes GIS hacia los OLAP.

#### 3.3.4.1 Interacción OLAP $\Rightarrow$ GIS

Las modificaciones al estado de las dimensiones geográficas en el reporte, deben sincronizarse con la visualización geográfica relacionada.

$\Rightarrow$ Debe ser implementado sobre los componentes GIS, operaciones que permitan modificar la vista geográfica actual, a partir de las nuevas categorías en las dimensiones geográficas.

**Que tenemos en la base de datos geográfica:**

- El mapeo entre los identificadores de los aspectos geográficos con las categorías de las dimensiones correspondientes.
- Cada aspecto geográfico tiene la información sobre cual es el aspecto geográfico al que pertenece.

A partir de dicha información se podrá modificar la visualización geográfica actual dada una categoría del data warehouse.

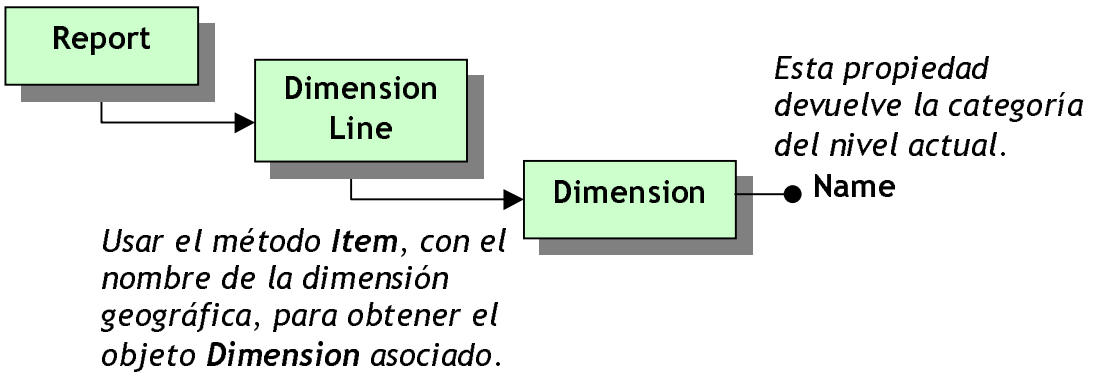
**Modificaciones al estado de una de las dimensiones geográficas:**

- Pasar a un nivel superior, seleccionando una categoría del nivel actual.
- Pasar al nivel inferior.

**Pasos para la interacción, al pasar a un nivel superior:**

1. Obtener la nueva categoría, resultante de la modificación del estado de una de las dimensiones geográficas.

   *Como obtener la nueva categoría con los componentes de PowerPlay:*
   (Ver Apéndice)

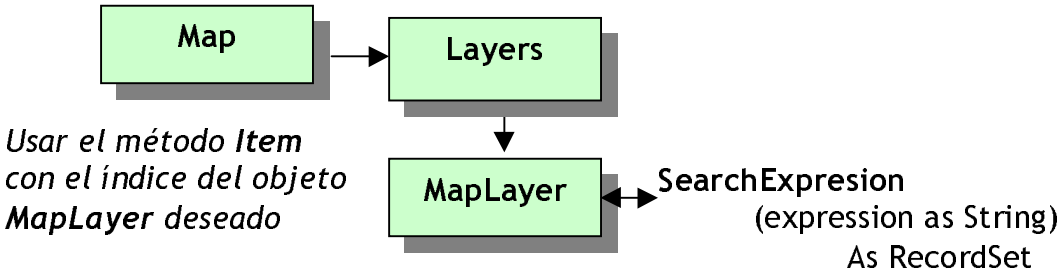



2. Identificar en el layer actual, que corresponde al nivel anterior a la modificación en la dimensión geográfica correspondiente, el aspecto geográfico asociado a la nueva categoría.
   *Como identificar el aspecto geográfico con los componentes de MapObjects:*

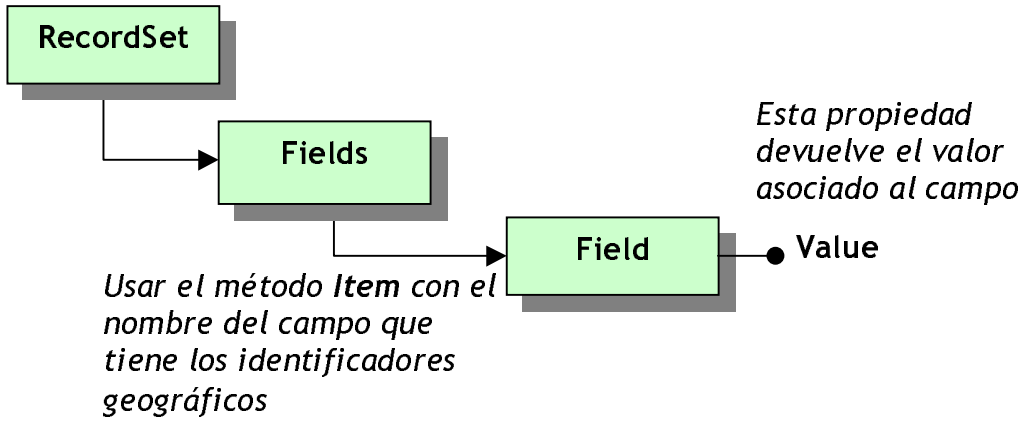
   Sobre el objeto **MapLayer** correspondiente al layer actual, usar el método **SearchExpresion** con la expresión SQL que busque al aspecto geográfico correspondiente a la nueva categoría.

   Debido a que un layer representa al nivel completo de una dimensión geográfica, para obtener el aspecto geográfico asociado a la nueva categoría se debe:

   1. Seleccionar los aspectos geográficos que pertenecen a la categoría anterior, usando el campo definido en cada layer para dicho propósito, con el identificador correspondiente a la categoría anterior.
      **Obs:** Esto no es necesario si el layer es el primero en el orden de niveles.
   2. Además seleccionar aquel aspecto que tiene como categoría asociada la nueva categoría, usando el campo que tiene el mapeo entre identificadores geográficos y categorías.



Con el objeto **RecordSet** obtenido de la búsqueda, se puede obtener el identificador, ya que dicho objeto tiene un solo registro resultante de la búsqueda anterior:



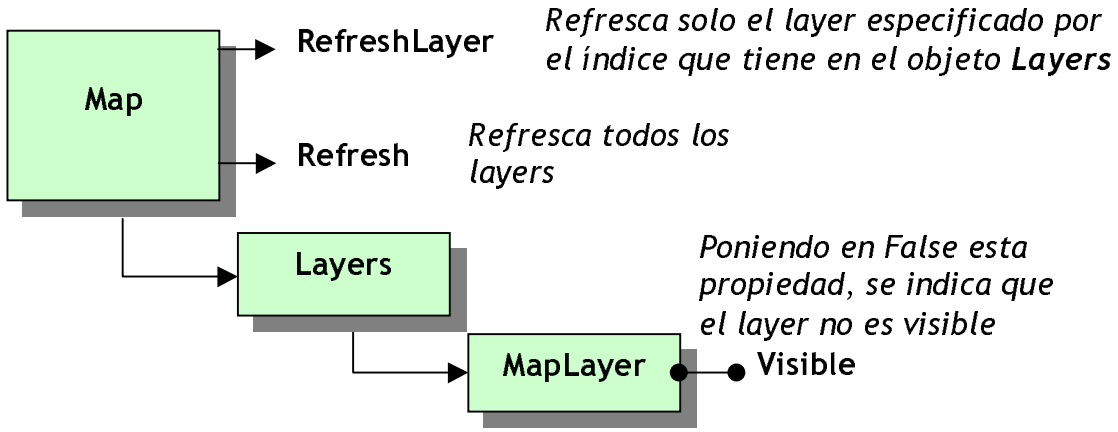

3. Para poder cambiar el enfoque sobre los aspectos geográficos que pertenecen al identificador que se obtuvo en el punto anterior, se debe obtener el objeto geométrico que representa al aspecto asociado a dicho identificador.

   *Como obtener el objeto geométrico con los componentes de MapObjects:*

   Con el mismo objeto **RecordSet** que se obtuvo de la búsqueda en el punto anterior, se puede obtener dicho objeto llamado **Shape**. Existe un campo definido en todos los layers, de nombre "shape", cuyo valor es el objeto en cuestión.

4. Ocultar los layers que no se correspondan al nuevo nivel.

   *Como ocultar los layers con los componentes de MapObjects:*



Luego de marcar cada objeto **MapLayer** como no visible, se debe refrescar al objeto **Map** para que no muestre los layers no visibles. Esto se puede hacer refrescando cada layer o refrescando todos los layers. El refresco completo puede ser ineficiente si solo se ocultan pocos layers de la totalidad.

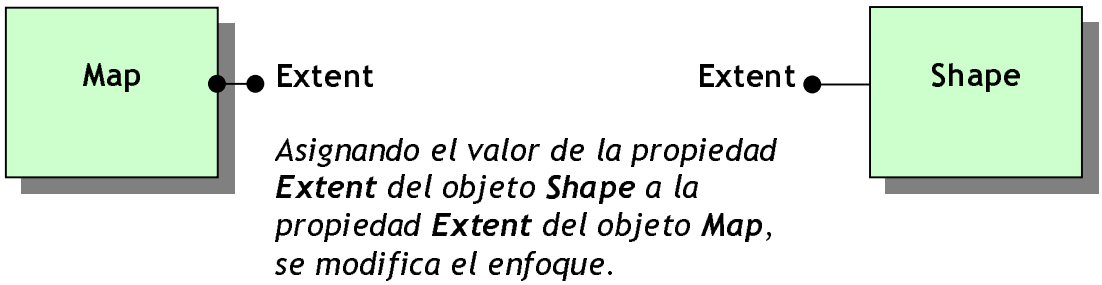
5. Hacer visible los layers correspondientes al nuevo nivel, enfocando la visualización en la interfase geográfica sobre los aspectos geográficos que pertenecen al aspecto seleccionado.

   *Como hacer visibles los layers con los componentes de MapObjects:*

   Repetir el punto anterior pero marcando los layers deseados como visibles.

   *Como cambiar el enfoque sobre los aspectos con los componentes de MapObjects:*

   En el punto 3 se obtuvo el objeto **Shape** asociado al aspecto geográfico correspondiente a la nueva categoría. Es a partir de dicho objeto que se modificará el enfoque sobre los aspectos geográficos.




   La propiedad **Extent** proporciona la extensión de un objeto geométrico, y asignándola a la interfase geográfica, a través del objeto **Map**, se modifica la visualización sobre dicha extensíon.

**Pasos para la interacción, al pasar a un nivel inferior:**

1. Ocultar los layers que no se correspondan al nuevo nivel.
2. Hacer visibles los layers que se correspondan al nuevo nivel, estableciendo el enfoque sobre los aspectos geográficos que pertenecen al aspecto asociado con la nueva categoría.

### 3.3.4.2 Interacción GIS ⇒ OLAP

Las modificaciones a la interfase geográfica deben reflejarse en las dimensiones geográficas asociadas en el reporte.

**Modificaciones en la interfase geográfica:**

- Pasar al layer siguiente, seleccionando un aspecto geográfico.
- Pasar al layer anterior.

**Pasos para la interacción, al pasar al layer siguiente:**

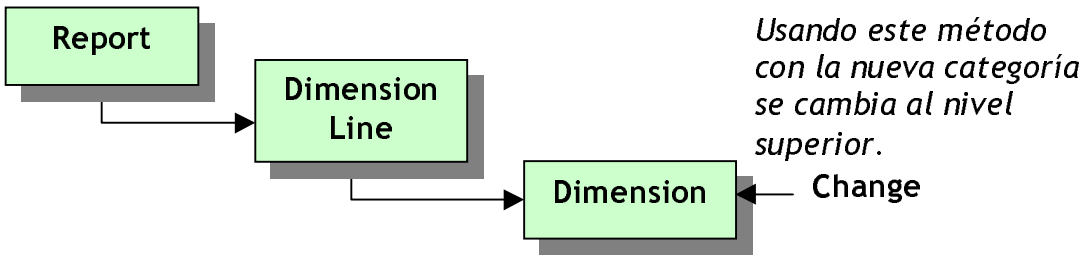
1. Al seleccionar un aspecto geográfico debe obtenerse la categoría asociada a dicho aspecto.

   *Como obtener la categoria asociada con los componentes de MapObjects:*

   Al seleccionar un aspecto geográfico, podemos obtener la categoría asociada a dicho aspecto ya que tenemos definido un campo en la tabla asociada al layer para dicho fin.

2. Con la categoría obtenida se debe cambiar al nivel superior en la dimensión correspondiente. Salvo que la dimensión tenga un solo nivel como hemos visto en la integración a nivel de datos.

   *Como cambiar al nivel superior con los componentes de PowerPlay:*

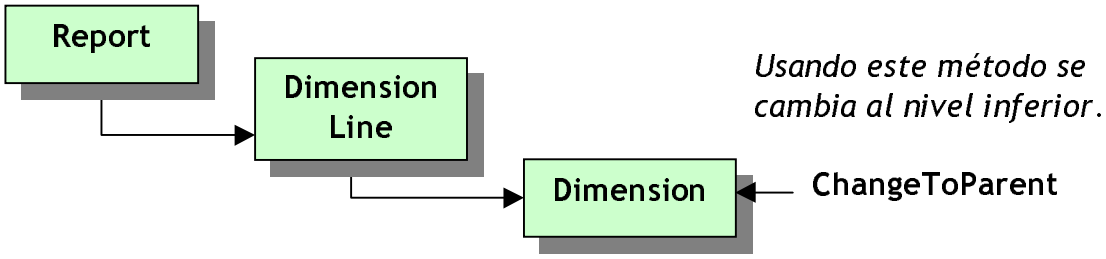



3. Ocultar los layers que no se correspondan al nuevo nivel.
4. Hacer visibles los layers que no se correspondan al nuevo nivel, estableciendo el enfoque sobre los aspectos geográficos pertencientes al aspecto asociado a la nueva categoría.

**Pasos para la interacción, al pasar al layer anterior:**

1. Pasar al nivel inferior al actual en la dimensión geográfica asociada.

   *Como cambiar al nivel inferior con los componentes de PowerPlay:*




2. Ocultar los layer que no corresponden al nuevo nivel.
3. Hacer visibles los layers que corresponden al nuevo nivel, cambiando el enfoque sobre los aspectos geográficos que pertenecen al aspecto asociado a la nueva categoría.

### 3.3.5 Integración de un reporte a una aplicación

**Objetivo:** Poder integrar un reporte de PowerPlay con una interfase geográfica en una misma interfase.
De esta forma se pueden sincronizar en una única interfase las operaciones que se realizan sobre los mismos.

| Problemas |
|---|
| 1. Los componentes de PowerPlay no ofrece una interfase para integrar en una aplicación.<br>2. Se debe usar un contenedor OLE para visualizar un reporte, pero no ofrece los controles para poder operar sobre el reporte.<br>3. Los controles de manejo del reporte deben ser implementados en la aplicación, accediendo al reporte mediante los componentes OLE de PowerPlay.<br>4. A través de los objetos OLE de PowerPlay es muy difícil poder obtener las categorías para seleccionar de las dimensiones. |

Las categorías solo se pueden obtener si la dimensión se encuentra visible en el reporte.

**Procedimiento para obtener las categorías de una dimensión no visible:**

1. Se deben obtener de la dimensión, las categorías que se quieren hacer visibles.

   *Como obtener las categorías con los componentes de PowerPlay:*

   **CategoryList** ← **Add**(Level, Dimension, Category, ...)

   *Este método permite cargar en el objeto **CategoryList** categorías de una dimensión dando el camino de categorías hasta el nivel deseado, y especificando hasta cuantos niveles desde ese nivel.*

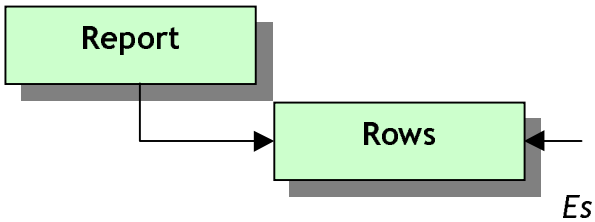
   Se deben cargar en el objeto **CategoryList** aquellas categorías que se desean hacer visibles. Esto se puede hacer de dos maneras:

   1. Si las categorías de la dimensión son únicas, especificando solo la última categoría para alcanzar el nivel donde se quieren obtener dichas categorías.

2. Si las categorías de la dimensión no son únicas, se debe especificar el camino de categorías en orden, para alcanzar al nivel donde se quieren obtener dichas categorías.

2. Luego de tener el objeto **CategoryList** con las categorías deseadas, se podrán hacer visibles tanto en las columnas, filas, o layers del reporte.

*Como hacer visibles las categorías con los componentes de PowerPlay:*




*Este método permite cargar en el objeto **Rows** asociado a las filas, un conjunto de categorías, especificando el objeto **CategoryList**.*

En caso de querer cargarse las categorías en las columnas, se tiene el objeto **Columns**, y para cargarse en los layers, el objeto **Layers**.

Todo este procedimiento es solo para obtener las categorías actuales de una dimensión, y el proceso debe repetirse con todas las demás si se quisieran visualizar las categorías de las mismas.

3.3.5.1 Inconvenientes de este procedimiento:

1. Se debe hacer visible una dimensión en el reporte, solo para obtener las categorías actuales.
2. Si las categorías no son únicas en una dimensión, se debe saber el camino de categorías hasta la categoría padre en el nivel actual de dicha dimensión. Pero no existen funcionalidades para saber cuales son las categorías seleccionadas en una dimensión para llegar al nivel actual, ni tampoco cual es el nivel actual de una dimensión.
3. Luego de obtenidas las categorías de una cierta dimensión, al querer obtener las de otra dimensión, se perderán las categorías de la primera, salvo que se la cargue en otra parte del reporte.

### 3.3.5.2 Alternativa a este procedimiento

Una forma de resolver los inconvenientes planteados, es implementar una capa de aplicación sobre los objetos OLE de **PowerPlay** que provea las funcionalidades que "faltarían" actualmente.


| Que debe proveer la capa de aplicación |
|---|
| La obtención de las categorías de cualquiera de las dimensiones del reporte, sin importar si es visible o no, debe ser una operación sencilla. |
| Poder saber el nivel actual de cualquiera de las dimensiones. |
| Poder saber cuales fueron las categorías seleccionadas para alcanzar el nivel actual de cualquiera de las dimensiones. |
| Implementar todas las operaciones que se pueden realizar sobre un reporte. |

⇒ Se requiere mantener en dicha capa de aplicación, información adicional sobre el estado del reporte para poder implementar estas funcionalidades.

La mínima información necesaria para satisfacer estas nuevas funcionalidades son, por cada dimensión:

- El nivel actual de la dimensión.
- Las categorías seleccionadas para alcanzar el nivel actual.
- Las categorías pertenecientes al nivel actual.

Con el nivel actual de una dimensión y las categorías seleccionadas para alcanzar dicho nivel, es posible obtener las categorías pertenecientes al nivel actual de dicha dimensión, usando los componentes de PowerPlay.

**Se requiere:**

- Actualizar esta información luego de cada operación que modifique el estado del reporte.
- Salvar esta información junto con el reporte.

Estos requerimientos son necesarios para mantener la coherencia entre la información adicional sobre las dimensiones, y el estado del reporte.

# 4 Desarrollos realizados

## *4.1 Proyecto Geo-DW (1ª Versión)*

### 4.1.1 Objetivo

Desarrollar un aplicativo que permite vincular la capacidad de análisis OLAP con la representación geográfica de esos datos, en un aplicativo específico para seguimiento de Censo.

### 4.1.2 Descripción

Construir un aplicativo VB que utilice las funciones de MapObject para presentar en pantalla datos provenientes del seguimiento del Censo.

La interface usuario debe estar en un único programa. La conexión entre este programa VB y PowerPlay se hará por OLE. Para ello se debe poder indicar las dimensiones o medidas a ser calculadas en PP desde la interface VB. En base a esa información, PP recibe los parámetros y calcula, devolviendo a VB los resultados. Esos resultados son tomados para presentar en pantalla, sobre el mapa manejado con MapObject.

### 4.1.3 Responsables

Inti Rebelese
Apoyo de Verónica Oliveira
Apoyo de Leonardo Laureiro

### 4.1.4 Análisis

En este aplicativo se combinan dos herramientas con el fin de colorear distintas zonas de un mapa, de acuerdo a valores que se obtienen de un data warehouse.

***Características:***

- El acceso al data warehouse se hace a través del uso de objetos OLE de la herramienta de consulta OLAP **PowerPlay**.
- Usando las funcionalidades de **MapObject** se pintan ciertas zonas de un mapa de acuerdo a colores asociados a rangos de valores, definidos previamente por el usuario.
- El lenguaje elegido para la integración de estas dos herramientas es Visual Basic (versión 5.0).

#### 4.1.4.1 Requerimientos

- ❖ Los datos usados para colorear los mapas se deben obtener de un data warehouse.
- ❖ El color asignado a cada zona dependerá, de los rangos de valores asociados a cada color, asi como de la medida del reporte que se seleccione.
- ❖ Interfase geográfica usando MapObjects.
- ❖ Se debe poder ir accediendo a mayor detalle en el mapa actual seleccionando una zona sobre la cual se quiera tener más información respecto a las subzonas que la componen.
- ❖ La selección de una zona del mapa debe hacer que el enfoque se ajuste sobre dicha zona, sobresaltándola del resto, y permitir visualizar con mayor detalle las subzonas existentes dentro de la zona en cuestión.
- ❖ La aplicación debe ser independiente del reporte usado para obtener los valores, así como de los layers que conforman la interfase geográfica. Lo cual debe ser configurable en un archivo de inicio.

#### 4.1.4.2 Desarrollo

Es un aplicativo donde solo existe interaccción desde los componentes GIS a los componentes OLAP:

- Debe haber una interfase geográfica que tomará valores de un data warehouse, para colorear zonas de un mapa.
- La "navegación" a través de las zonas del mapa, debe ser sincronizada con el reporte de donde se toman los valores.
- Los valores dependerán de la medida seleccionada.

| Dimensiones de interés |
| --- |
| Dimensión geográfica |
| Dimensión de medidas |

**Definición de la dimensión geográfica:**

Debido a que solo se manejaran zonas de un mapa, los únicos aspectos geográficos que se representarán en la dimensión geográfica son aquellos del tipo polígonos. Se deberán definir tantos niveles como detalle de zonas se tengan en el mapa.

### 4.1.5 Diseño

#### 4.1.5.1 Componentes principales

De acuerdo a la realidad planteada se pueden distinguir dos componentes principales en este aplicativo:

- Un componente se encarga del manejo del reporte.
- Un componente se encarga del manejo de la interfase geográfica.

El **componente encargado del manejo del reporte** debe permitir las siguientes operaciones:

- Asignar el reporte a manejar.
- Modificar la categoria actual de la dimensión geográfica.
- Modificar la medida actual.
- Obtener las medidas.

Las funcionalidades básicas para el manejo del reporte deben permitir modificar el nivel de una dimensión, seleccionando una categoría para incrementar de nivel, o pasar al nivel inferior. Por lo cual deben ser definidas operaciones que permitan manejar cualquiera de las dimensiones.

El **componente encargado de manejar la interfase geográfica** debe permitir las siguientes operaciones:

- Asignar los layers a ser usados, y el orden de los mismos.
- Manejar de los distintos layers, modificando el layer visible de acuerdo a la "navegación" que se realice en la interfase geográfica.
- Colorear las zonas de los layers de acuerdo a los valores que se toman del data warehouse, asi como del rango valores asociados a colores.
- Resaltarse las zonas que se corresponden con las categorías del data warehouse.

Los layers que serán manejados por esta clase deben cumplir los requerimientos planteados en el capítulo sobre Formas de integración, en el punto sobre Condiciones para la integración.

**Objetivos buscados en el diseño de los componentes:**

La interacción entre los componentes debe ser tal que que se logre la mayor independencia entre estos:

- Facilmente poder cambiar uno de los componentes y que esto no repercute en el funcionamiento de los demás.
- El componente que maneja la interfase debe ser independiente respecto del origen de los valores que se usan para colorear, ya que en esta implementación particular, los valores se van a tomar de un reporte.

La finalidad de este diseño es:

- Tener un conjunto de componentes lo más generales e independientes posibles. Pudiendo integrarlos independientemente en cualquier aplicación.
- Escalabilidad de los componentes.
- Reusablidad de los componentes.
- Previsibilidad al cambio.

Dichos objetivos son claramente más alcanzables con un desarrollo orientado a objetos, pasando a ser cada una de las componentes una clase. Este enfoque será aplicado en la implementación de esta aplicación.

Como forma de lograr la mayor independencia entre los componentes principales, se debe lograr la menor cohesión posible entre los mismos:

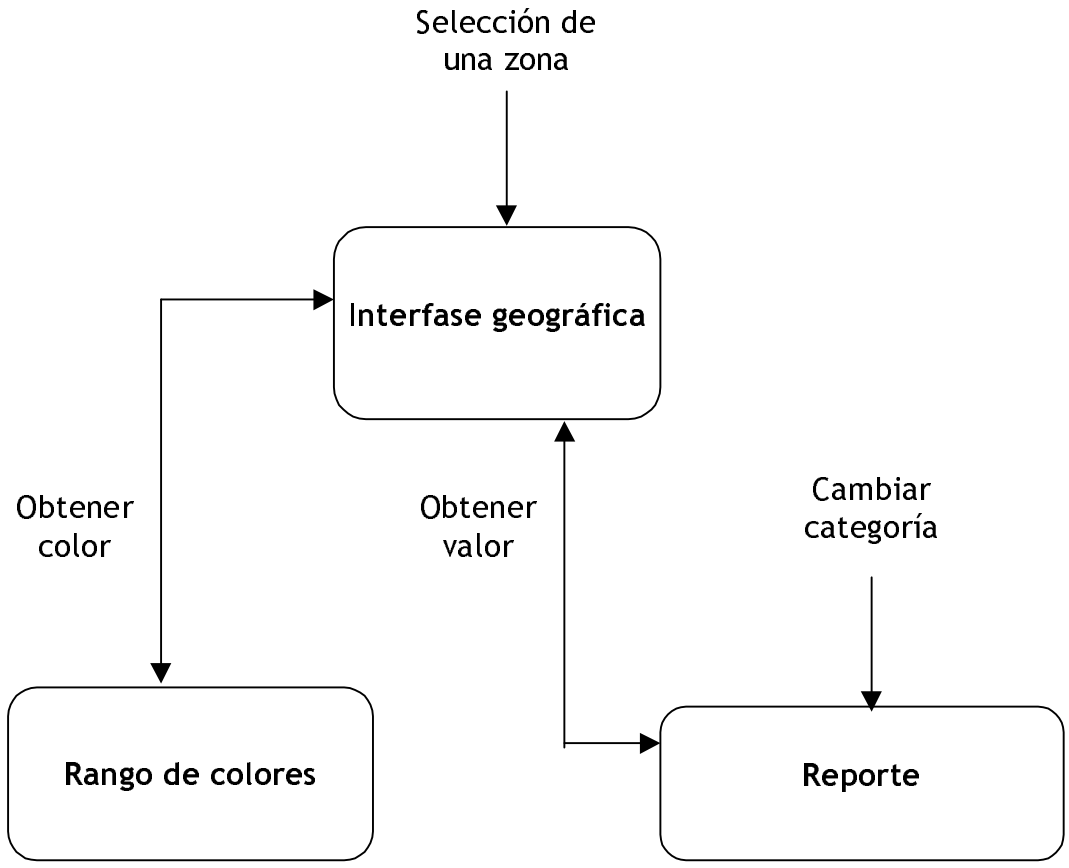
| **Interacción mínima de los componentes GIS ⇒ OLAP** |
|---|
| Obtener valores para la coloración de los layers. |
| **Interacción mínima de los componentes OLAP ⇒ GIS** |
| Ninguna. |

#### 4.1.5.2 Componentes secundarios

Para que la clase que maneja la interfase geográfica pueda pintar cierta zona, no solo necesita de los valores que obtiene de la clase que maneja el reporte, sino que debe obtener el color correspondiente de acuerdo a los rangos de valores que se hayan definido. Por lo cual definimos otra **clase para el manejo de rangos de valores y sus colores asociados**. Dicha clase debe ofrecer funcionalidades tales como:

- Agregar rangos de valores con su color asociado.
- Modificar los rangos.
- Modificar los colores.
- Dado un valor devolver el color que le corresponde. En el caso que un cierto valor no se encuentra en ninguno de los rangos definidos, devolver un color por defecto.

4.1.5.3 Interacción entre las clases hasta ahora especificadas



#### 4.1.5.4 Refinamiento del diseño

MapObjects permite manejar distintos criterios para la asignación de atributos a los aspectos geográficos de los layers. Entre esos atributos están los colores a asignar a los aspectos geográficos de un layer. La asignación de atributos a un layer se hace asignando un tipo de objeto particular, llamado ***Renderer***. Según los atributos que se quieran asignar, se asignará cierto tipo de objeto Renderer al objeto MapLayer asociado al layer.

> Una característica común de todos los tipos de objetos Renderer, es que se debe especificar cuál es el campo de la tabla asociada al layer que se usará como referencia para la asignación de los atributos.

Los tipos de objetos Renderer básicos existentes en MapObjects son los siguientes:

| Tipo de objeto Renderer | Características |
|---|---|
| ValueMapRenderer | Permite asignar un conjunto de atributos especificando determinados aspectos geográficos. |
| ClassBreaksRenderer | Permite asignar un conjunto de atributos por rangos de valores del campo especificado. |
| LabelRenderer | Permite asignar etiquetas, que son tomadas de la tabla. Se aplica a todos los aspectos geográficos del layer. |

Todos los objetos Renderer se basan sobre un cierto campo de la tabla asociada al layer, pero en nuestra situación no nos basamos en valores que se tomen de la propia tabla sino de valores que se encuentran en otra fuente (data warehouse).

El tipo de objeto Renderer apropiado para la coloración es aquel que permite asignar a un determinado aspecto geográfico un determinado conjunto de atributos, que en nuestro caso es el color.

⇒ el tipo de Renderer apropiado es el ***ValueMapRenderer***.

**Procedimiento para la coloración de las zonas:**

- Definir un objeto ValueMapRenderer.
- Especificar a dicho objeto que el campo donde obtener los valores de referencia es aquel que identifica en forma única a las zonas en el layer.
- Para cada zona a colorear tomar la categoría que se encuentra asociada a dicha zona en la tabla.
- Del objeto que maneja el reporte, con la categoría obtenida, obtener el valor usado para colorear.
- Del objeto que maneja los rangos de valores asociados a colores, con el valor obtenido, obtener el color correspondiente.
- Asignar el color obtenido a dicha zona en el objeto ValueMapRenderer.

**Importante:** El campo usado por el objeto ValueMapRenderer para obtener las referencias de las zonas a pintar, debe tener identificadores únicos de las zonas en todo el layer. Ya que sino al asignar un determinado color a una zona, puede tener como efecto colateral, que otras zonas sean pintadas con el mismo color.

*4.1.5.4.1 Nueva clase para manejar los objetos Renderer*

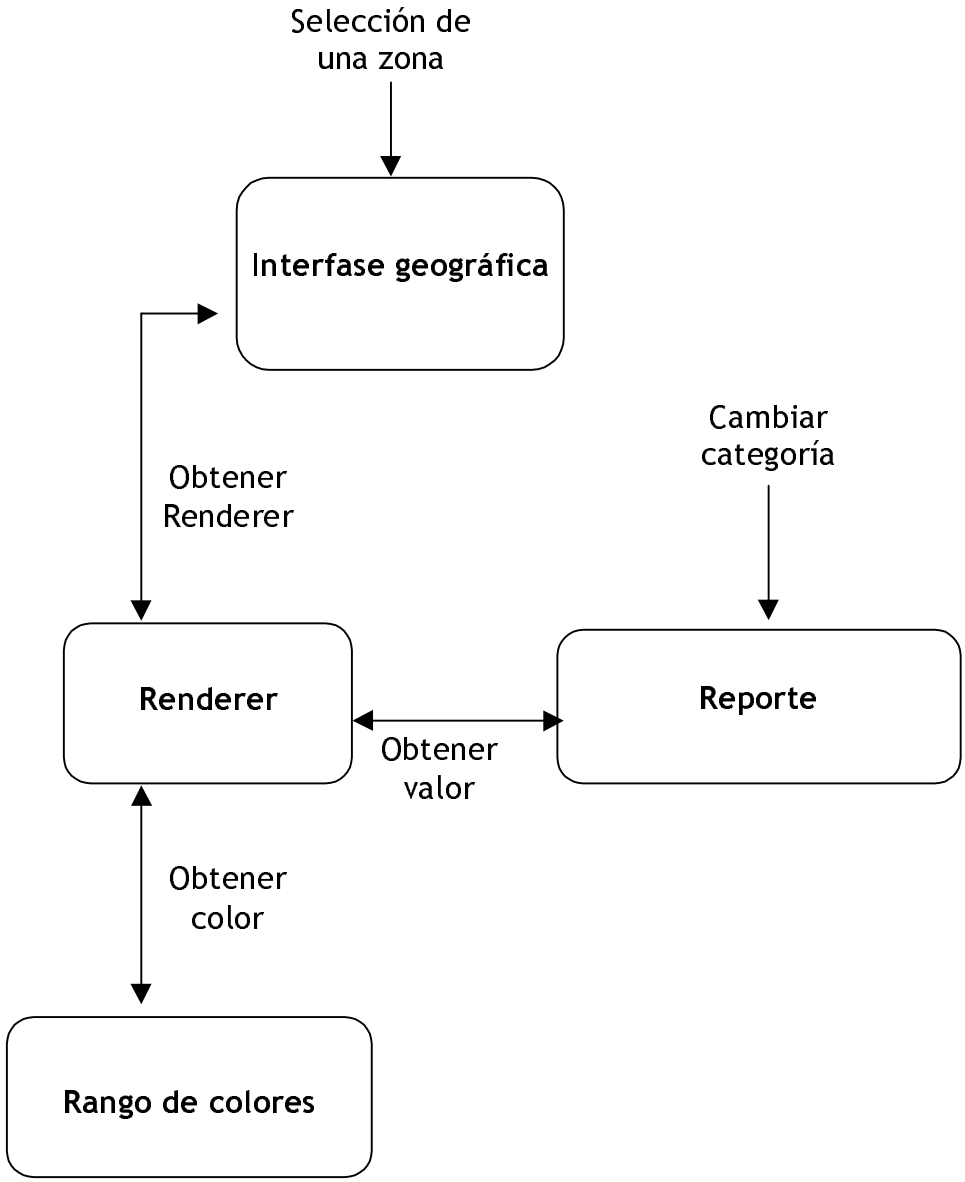
La asignación de atributos a los aspectos geográficos de los layers depende de los criterios que se quieran manejar. De forma de independizar a la clase que maneja la interfase geográfica del criterio usado para la asignación de atributos a cada uno de los aspectos geográficos, se podría definir una nueva clase que tenga como cometido proveer los objetos Renderer a usar en la coloración. De esta forma se logra mantener los objetivos buscados en el diseño de los componentes.

Por lo tanto, tendremos una nueva **clase para manejar los objetos Renderer** a asignar a cada layer.

**Características de la nueva clase para nuestra situación:**

- Se le debe especificar el objeto de donde tomará los valores para determinar los colores a asignar.
- Se le debe especificar el objeto de donde obtendrá el color a asignar según cierto valor.

4.1.5.5 Interacción entre las clases hasta ahora especificadas:



### 4.1.6 Archivo de configuración

Uno de los requerimientos de este aplicativo es el poder configurar en un archivo de configuración, distintos parámetros que permitan especificar ciertas características:

- Camino donde se encuentra el reporte.
- Nombre de la dimensión geográfica.
- Camino donde se encuentran los layers.
- Cantidad de layers.
- Campo, por cada layer, de los identificadores de los aspectos geográficos.
- Campo, por cada layer, de las categorías asociadas a los aspectos geográficos.

**Requisitos:**

El archivo de configuración debe llamarse igual al nombre del ejecutable de la aplicación, pero con la extensión "ini". Además debe estar ubicado en el mismo directorio que el ejecutable.

## *4.2 Proyecto Geo-DW (2ª Versión)*

### 4.2.1 Objetivo

Combinar en una única aplicación dos formas de explorar y analizar la información contenida en un data warehouse, una a través de un reporte de PowerPlay, y otra a través de una interfase geográfica basada en MapObjects. En ambas interfaces la dimensión geográfica se maneja mediante una interfase geográfica, siendo esta una forma más intuitiva y amigable para filtrar esta dimensión.

### 4.2.2 Descripción

Desarrollar un aplicativo que permita visualizar de dos maneras distintas la información de un data warehouse, ofreciendo para ello dos interfases:

- Una interfase que contenga un reporte de PowerPlay, pudiendo realizar distintas operaciones básicas sobre el mismo. Para filtrar aquella dimensión que se corresponde con la información geográfica, usar una interfase geográfica basada en MapObjects.

- Una interfase geográfica que permita colorear las áreas de un mapa eligiendo, además de los rangos de valores asociados a colores, la dimensión así como la categoría de la misma, y la medida para obtener los valores.

### 4.2.3 Responsables

Inti Rebelese
Apoyo de Verónika Peralta
Apoyo de Leonardo Laureiro

### 4.2.4 Análisis

En este aplicativo se debe poder acceder a un data warehouse de dos formas posibles, una a través de un reporte, pudiendo realizar operaciones sobre el mismo, y otra a través de una interfase geográfica.

La interfase con el reporte debe ser más sencilla que la de PowerPlay, con las operaciones más básicas. La misma será enriquecida con una parte geográfica para el manejo de la dimensión asociada a la información geográfica.

La interfase geográfica es del estilo al proyecto Geo-DW 1ª Versión, pero con la posibilidad de seleccionar la dimensión por la cuál filtrar y colorear los mapas. Permitiendo así mayores opciones de análisis a través de esta interfase.

#### 4.2.4.1 Requerimientos

- Interfase con reporte, con las operaciones necesarias para el análisis del mismo.
- Filtrar la dimensión asociada a la información geográfica, usando una interfase geográfica. Sincronizando los cambios hechos en esta con el reporte, y viceversa.
- Interfase geográfica con la posibilidad de seleccionar la dimensión por la cual filtrar y la categoría de la misma, para la coloración de un mapa. Debe ser posible también cambiar de nivel en dicha dimensión y recorrer así las categorías pertenecientes a la misma.
- Poder ver etiquetas, asociadas a las categorias, sobre cada zona del mapa de acuerdo al nivel actual de la dimensión que describe la geografía, tanto en la interfase geográfica como en la de reporte.
- Salvar el estado en el que se encuentra el reporte asi como la geografía.
- La aplicación debe ser independiente del reporte usado para obtener los valores, así como de los layers que conforman la interfase geográfica. Además se debe poder configurar cuales son las dimensiones del reporte a visualizar, así como las medidas. Lo cual debe ser configurable en un archivo de inicio.

#### 4.2.4.2 Desarrollo

Existen distintos requerimientos para cada una de las interfases de la aplicación.

##### *4.2.4.2.1 Interfase reporte*

Cada uno de los requerimientos que deben ser satisfechos, determinan distintos problemas a ser resueltos:

- Interfase con reporte, con las operaciones necesarias para el análisis del mismo.

  1. Para poder visualizar un reporte en un aplicativo se debe usar un contenedor OLE. Las operaciones aplicadas sobre el reporte deben reflejarse en la interfase.

  **Problema:**

  Uno de los mayores problemas que surgen de manejar un reporte a través de un contenedor OLE, es que no se tienen las barras de scroll que permiten recorrer el reporte. Lo cual es imprescindible para la visualzación correcta del reporte.

  2. Las operaciones necesarias para poder analizar un reporte deben ser implementadas. Con el agregado de que se configura a través de un archivo de inicio (.INI), que dimensiones del reporte serán visibles asi como las medidas.

  **Problema:**

  El hecho de hacer una nueva interfase para el reporte del tipo de PowerPlay pero más sencilla, lleva a tener que resolver los problemas planteados en el capítulo sobre Formas de Integración, en el punto que trata sobre la integración de un reporte a una aplicación.

- Salvar el estado en el que se encuentra el reporte.

  1. Salvar el estado de un reporte de PowerPlay es sencillo ya que los componentes proveen operaciones para ello.

**Problema:**

Se debe salvar toda la información adicional para que el aplicativo pueda recuperar el estado en que se encuentra.

- Filtrar la dimensión asociada a la información geográfica, usando una interfase geográfica.

1. Usar una interfase geográfica para filtrar la dimensión geográfica, logrando asi una forma más intuitiva y amigable al usuario de manejar dicha dimensión.

**Problema:**

Se deben mantener sincronizado los cambios hechos a la dimensión geográfica a través del reporte, con las vistas geográficas que se tienen en la interfase. Así como los cambios hechos a través de la interfase geográfica, con el reporte.

*4.2.4.2.2 Interfase geográfica*

- Interfase de visualización geográfica, pero con el agregado de poder seleccionar la dimensión y categoría por la cual filtrar.

    1. Se debe permitir seleccionar la dimensión de la cual se quiere, según la categoría y medida seleccionada, obtener los valores para colorear el mapa.

    **Problema:**

    Poder seleccionar la dimensión de la cual obtener los valores para la coloración del mapa, implica tener que hacer dicha dimensión visible en el reporte.

- Poder ver etiquetas, asociadas a las categorias, sobre cada zona del mapa.

    1. Se debe asignar a cada layer un nuevo tipo de objeto Renderer, que tome en cuenta dichas etiquetas así como los colores a asignar a las zonas.

    **Problema:**

    Solo las zonas que se corresponden con el nivel actual de la dimensión geográfica deben aparecer etiquetadas.

- Salvar el estado en el que se encuentra la interfase geográfica

    1. El estado en el que se encuentra la interfase geográfica solo depende del estado en el que se encuentra la dimensión geográfica, asi como del estado de la otra dimensión visible en el reporte.

    **Problema:**

    Se deben salvar los rangos de valores definidos y sus colores asociados, ya que son parte de la información necesaria para la coloración del mapa.

### 4.2.5 Diseño

Debido a los nuevos requerimientos planteados, deben modificarse las clases definidas en la 1ª versión, así como definir nuevas clases.

#### 4.2.5.1 Modificaciones a las clases principales

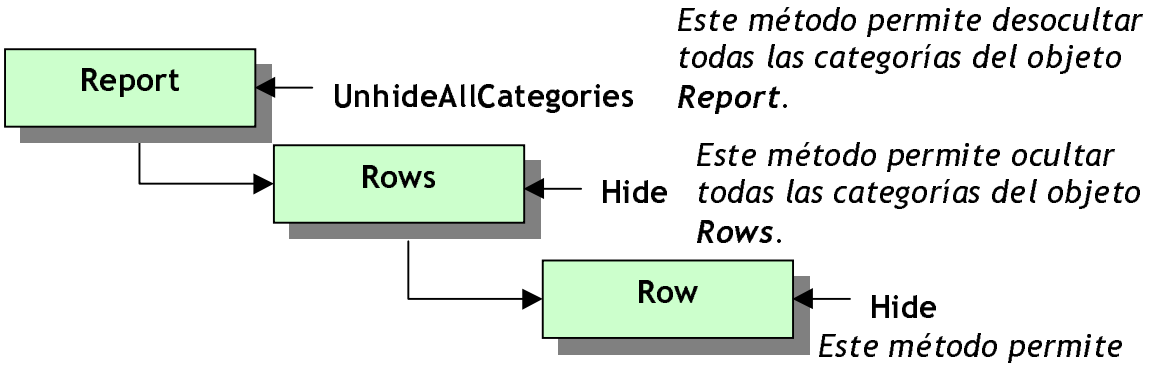
##### *4.2.5.1.1 Clase que maneja el reporte*

- Operaciones para ocultar categorías en las dimensiones visibles del reporte.

Debido a que la integración de un reporte de PowerPlay a través de un contenedor OLE, no permite visualizar la barra de scroll necesaria para recorrerlo , se debe simular el scroll del reporte ocultando y desocultando las categorías de las dimensiones, de acuerdo a que parte del reporte se quiere visualizar.

⇒ Se deben agregar a la clase que maneja del reporte las operaciones necesarias para ocultar y desocultar categorías de cualquiera de las dimensiones visibles en el reporte. Las mismas se basarán en las operaciones que proveen los componentes de PowerPlay.

**Operaciones provistas por los componentes de PowerPlay:**




Las operaciones para ocultar existen también para los objetos **Columns** y **Layers**.

El problema con estas operaciones es que la única que permite desocultar se aplica a todas las categorías del reporte.

**Procedimiento para simular el scroll del reporte:**

En el aplicativo se deben agregar las barras de scroll en los dos sentidos, vertical y horizontal.
Por cada sentido:

- Al avanzar la barra de scroll se deben ocultar la primeras categorías, si el índice de dichas categorías es:
  - mayor a la cantidad de categorías visibles en ese sentido en el contenedor OLE.
  - menor a la cantidad total de categorías de la dimensión visible en ese sentido, menos la cantidad de categorías visibles en el contenedor OLE en ese sentido.

- Al retroceder la barra de scroll se deben desocultar todas las categorías y ocultar aquellas categorías anteriores a la última categoría que estaba oculta.

Para facilitar el procedimiento, la operación para ocultar categorías debe poder ocultar categorías dando un rango de las mismas. Se logra así mejorar la funcionalidad que ofrece la operación de ocultar categorías provista por los componentes de PowerPlay.

- Mostrar en el reporte las medidas que se hayan especificado en el archivo de configuración.

Para ello se deben ocultar dichas las medidas de la dimensión correspondiente. Esto se logra con las operaciones de ocultar y desocultar que se definieron en el punto anterior.
Es importante tomar en cuenta en el procedimiento para simular el scroll del reporte, que al desocultar las categorías se deben ocultar las medidas no especificadas en el archivo de configuración.

- Proveer las operaciones básicas que permitan manejar el reporte, y proveer nuevas operaciones que no son provistas por los componentes de PowerPlay.

La clase que maneja el reporte debe implementar la capa de aplicación que se propuso en el capítulo sobre Formas de Integración, en el punto sobre integración de un reporte a una aplicación.

| Operaciones básicas para el manejo del reporte |
| --- |
| Pasar al nivel superior, especificando la dimensión y la categoría. |
| Pasar al nivel inferior, especificando la dimensión. |

| Operaciones nuevas |
| --- |
| Obtener las categorías que pertenecen al nivel actual de una dimensión, especificando la dimensión y el índice de la categoría.<br>Obtener el nivel actual de una dimensión, especificando la dimensión.<br>Obtener las categorías seleccionadas para alcanzar el nivel actual de una dimensión, especificando la dimensión y el nivel.<br>Hacer visible una dimensión, especificando la dimensión y en que parte del reporte debe hacerse visible. |

Como se especificó en el punto de integración de un reporte a una aplicación, se debe mantener información adicional al reporte para poder ofrecer estas nuevas funcionalidades. Esta información debe mantenerse sincronizada con las modificaciones al estado del reporte, a través de las operaciones aplicadas a la clase que maneja el reporte.

⇒ Se definirá una nueva clase que mantendrá dicha información, y la misma será usada internamente por la clase que maneja al reporte.

**Operaciones que actualizan la información adicional:**

- Pasar al nivel superior especificando la categoría.
- Pasar al nivel inferior.

Al salvar el estado del reporte se debe almacenar esta información adicional.

Al abrir un reporte se debe recuperar la información almacenada y cargarla en la clase que la maneja

#### *4.2.5.1.2 Nueva clase para manejar la información adicional sobre las dimensiones*

De forma que la clase que maneja el reporte pueda proveer nuevas funcionalidades no provistas por los componentes de PowerPlay, se debe mantener cierta información adicional sobre el estado de las dimensiones del reporte.

La información que se debe mantener está detallada en el punto sobre la integración de un reporte a una aplicación.

Para mantener y actualizar esta información, la clase debe proveer las siguientes funcionalidades:

- Aumentar el nivel actual de una dimensión, especificando la categoría seleccionada.
- Decrementar el nivel actual de una dimensión.
- Especificar las categorías pertenecientes al nivel actual de una dimensión.

Estas funcionalidades no modifican el estado del reporte, sino solo de la información que se mantiene sobre las dimensiones del reporte.

#### *4.2.5.1.3 Clase que maneja la interfase geográfica*

- Modificar el layer actual especificando una categoría.

Este requerimiento es necesario para la interacción entre la clase que maneja el reporte y la que maneja la interfase geográfica. El procedimiento para implementar este requerimiento fue detallado en el capítulo sobre Formas de Integración, en el punto sobre Interacción entre los componentes OLAP $\Rightarrow$ GIS.

- Obtener la categoría correspondiente a la zona seleccionada.

Luego de seleccionar una zona, se debe obtener la categoría asociada a dicha zona para cambiar el nivel de la dimensión geográfica en la clase que maneja el reporte. Los detalles de implementación de este requerimiento fue detallado en el capítulo sobre Formas de Integración, en el punto sobre Interacción entre los componentes GIS $\Rightarrow$ OLAP.

*4.2.5.1.4 Clase que maneja los objetos Renderer*

- Poder ver etiquetas sobre las zonas del mapa asociadas a las categorias.

Este requerimiento implica que la clase que maneja los objetos Renderer a asignar a los layer, deba usar otro tipo de objeto Renderer. El nuevo tipo debe permitir:

- Asignar a cada zona que se corresponda con las categorías del nivel actual, etiquetas correspondientes a los nombres de dichas categorías.
- Asignar colores de acuerdo a los valores que se obtienen del reporte, y de los rangos de valores asociados a colores.

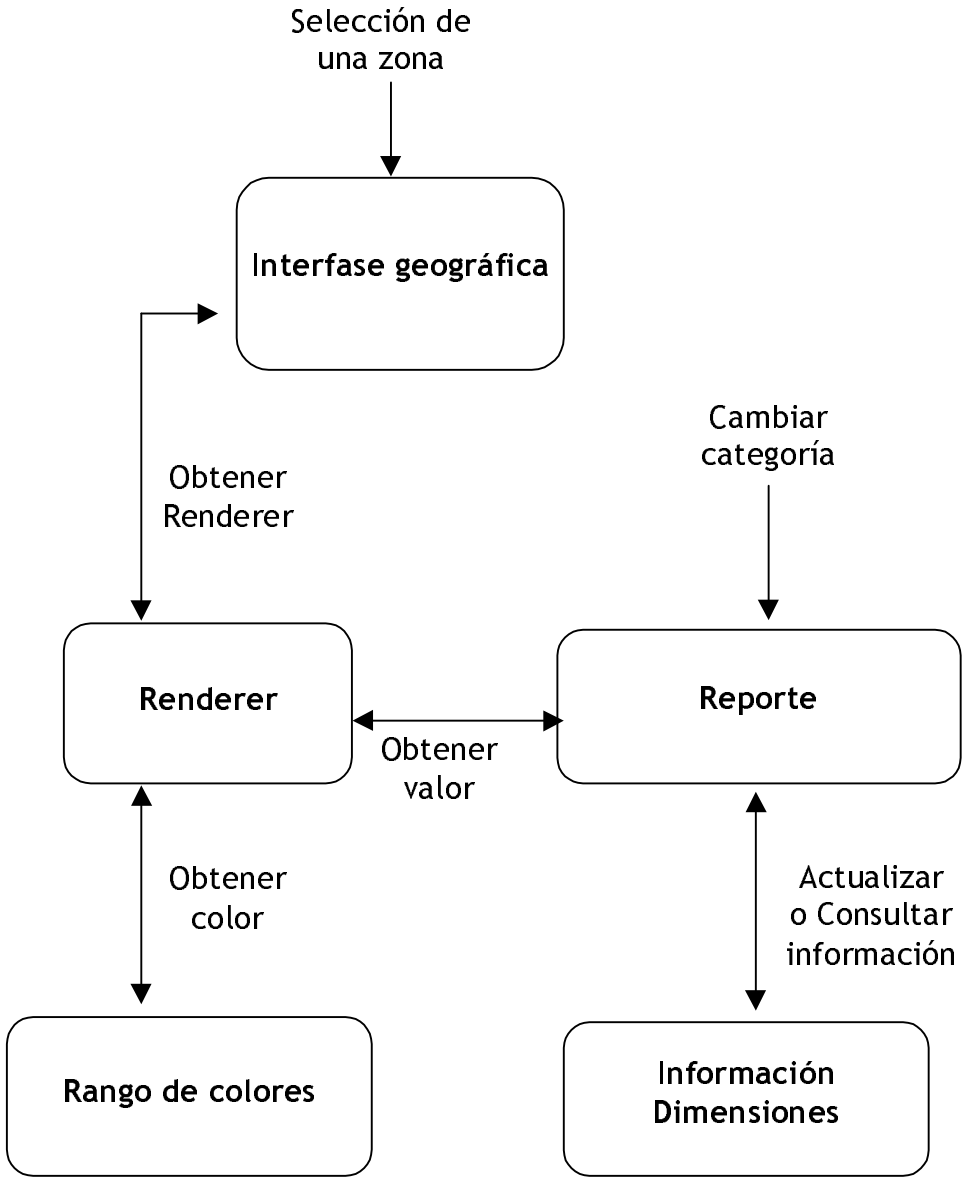
Existe un tipo de objeto Renderer, llamada **LabelRenderer**, que permite asignar estos atributos a los layers.

| Características del objeto LabelRenderer |
| --- |
| Permite especificar a cuales aspectos geográficos se les debe asignar etiquetas.<br>Se debe especificar un campo de la tabla, cuyos valores serán las etiquetas.<br>Se le puede asociar otro objeto Renderer que determinará otros atributos además de las etiquetas. |

De forma de colorear las zonas usando este tipo de objeto Renderer, usaremos la propiedad que tiene de poder especificar otro objeto Renderer. Por lo cual asignaremos el tipo de objeto Renderer usado para la coloración hasta ahora, el ValueMapRenderer.

**Observación:** Este objeto no viene en el conjunto de componentes de MapObjects, sino que viene en otro ActiveX, llamado MoPlus.

4.2.5.2 Interacción entre las clases hasta ahora especificadas:



### 4.2.6 Archivo de configuración

Uno de los requerimientos de este aplicativo es el poder configurar en un archivo de configuración, distintos parámetros que permitan especificar ciertas características:

- Camino donde se encuentra el reporte.
- Nombre de la dimensión geográfica.
- Cantidad de dimensiones visibles en la aplicación.
- Dimensiones visibles en la aplicación.
- Cantidad de medidas visibles.
- Medidas visibles.
- Camino donde se encuentran los layers.
- Cantidad de layers.
- Campo, por cada layer, de los identificadores de los aspectos geográficos.
- Campo, por cada layer, de las categorías asociadas a los aspectos geográficos.

**Requisitos:**

El archivo de configuración debe llamarse igual al nombre del ejecutable de la aplicación, pero con la extensión "ini". Además debe estar ubicado en el mismo directorio que el ejecutable.

## *4.3 Especificación de las clases implementadas*

**Notación:**

***Métodos***:

Procedimiento

Función

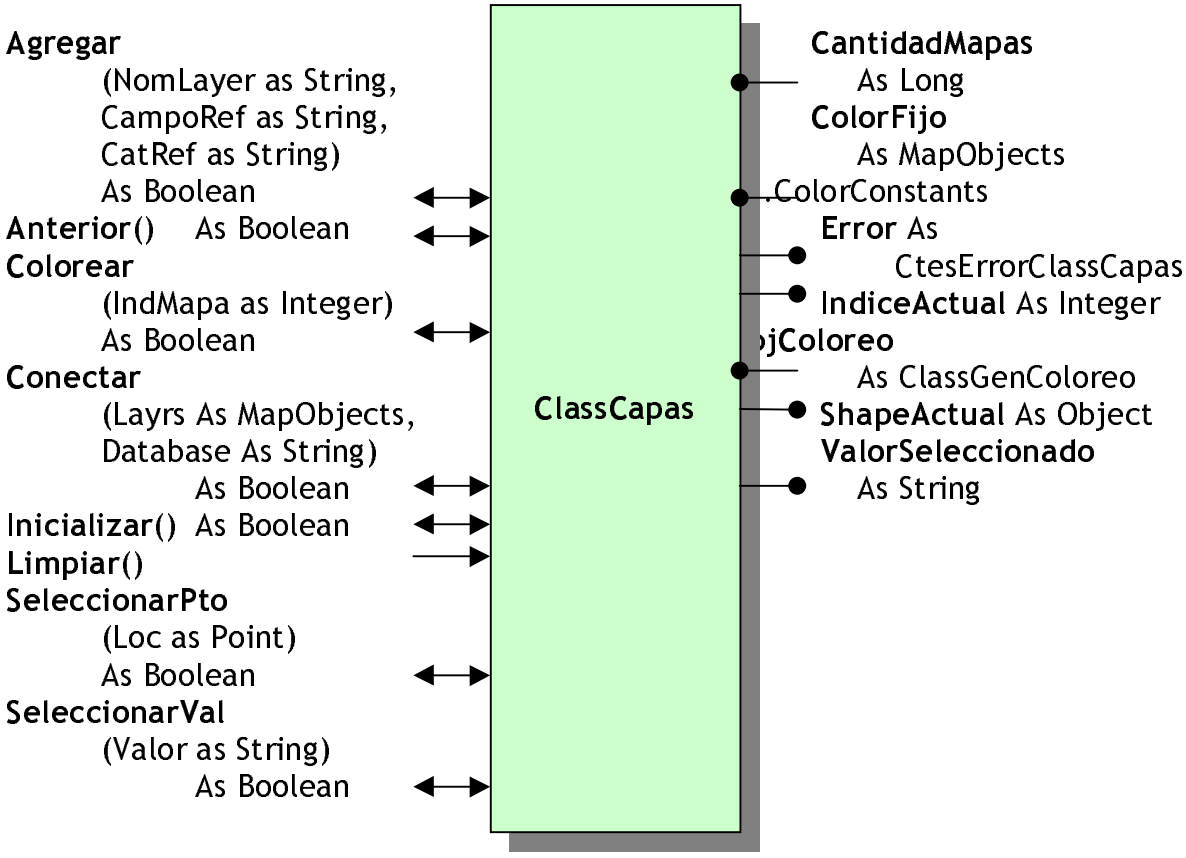
***Propiedades***:

Read only

Write only

Read/Write

### 4.3.1 Clase encargada de manejar la interfase geográfica (ClassCapas)



4.3.1.1 Métodos:

**Agregar**( NomLayer as String, CampoRef as String, CatRef as String) As Boolean

Dado el nombre del archivo asociado a un Layer (.shp), el nombre del campo que identifica a cada una de las zonas del layer, y el nombre del campo que identifica cada una de las zonas en el reporte, agrega dicho layer a la interfase. El primer layer que se agrega representa la vista superior, los siguientes layers que se agregan representan en orden las vistas inferiores.
Si no se produce ningún error devuelve True, en caso contrario False.

**Anterior**() As Boolean

Cambia la vista actual a la anterior, si puede devuelve True, sino devuelve False.

**Colorear**() As Boolean

Colorea el layer correspondiente al valor de la propiedad **IndiceActual**.
Si no se produce ningún error devuelve True, sino False.

**Conectar**(Layrs as Layers, Database as String) As Boolean

Dado un objeto Layers y el camino completo donde se ubican los layers, establece una conexión con la base de datos.
Si no se produce ningún error devuelve True, en caso contrario False.

**Inicializar**() As Boolean

Carga la primer vista de acuerdo a los layers que se hayan agregado a la interfase, queda a cargo de quien use este objeto de refrescar el layer correspondiente. Al aplicar este método se setea la propiedad **IndiceActual** en el valor correspondiente al índice del primer layer cargado.
Si no se produce ningún error devuelve True, en caso contrario False.

**Limpiar**()

Elimina todos los layers cargados.

**SeleccionarPto**(Loc as Point) As Boolean

Dado un objeto Point, cambia a la vista siguiente seteando la propiedad **ShapeActual** con el objeto Shape asociado a la zona seleccionada y pintando dicha zona de acuerdo a los valores del reporte y de los colores asociados a los rangos de valores definidos.
Si no se produce ningún error devuelve True, sino False.

**SeleccionarVal**(Valor as String) As Boolean

Dado el valor, cambia a la vista siguiente seteando la propiedad **ShapeActual** con el objeto Shape asociado a la zona cuyo valor en el campo CatRef es igual al mismo y pintando dicha zona de acuerdo a los valores del reporte y de los colores asociados a los rangos de valores definidos.
Si no se produce ningún error devuelve True, sino False.

4.3.1.2 Propiedades:

**CantidadMapas** As Integer

Establece la cantidad de layers que van a ser cargados en el objeto.

**ColorFijo** As MapObjets.ColorConstants

Establece el color que se va a usar en el coloreo en caso de que no se setee el objeto **ClassGenColoreo** usando la propiedad **ObjColoreo**.

**Error** As CtesErrorClassCapas

Devuelve ciertos valores que indican si hubo error o no al aplicar cierta operación. En el caso de que haya error, el valor devuelto depende del tipo de error que haya ocurrido. Dichos valores se pueden comparar con las constantes definidas para dicho fin.

**IndiceActual** As Integer

Devuelve el valor correspondiente al índice del layer que actualmente puede visualizarse, dicho layer corresponde a la vista que actualmente se seleccionó. Dicho valor puede ser usado con el método **RefreshLayer** del objeto MapControl, con lo cual se logra hacer visible dicho layer, y ocultar el anterior, usar este método permite evitar que se tenga que refrescar cada uno de los layers, usando el método **Refresh**, cuando el único que se desea refrescar es el actual y el anterior.

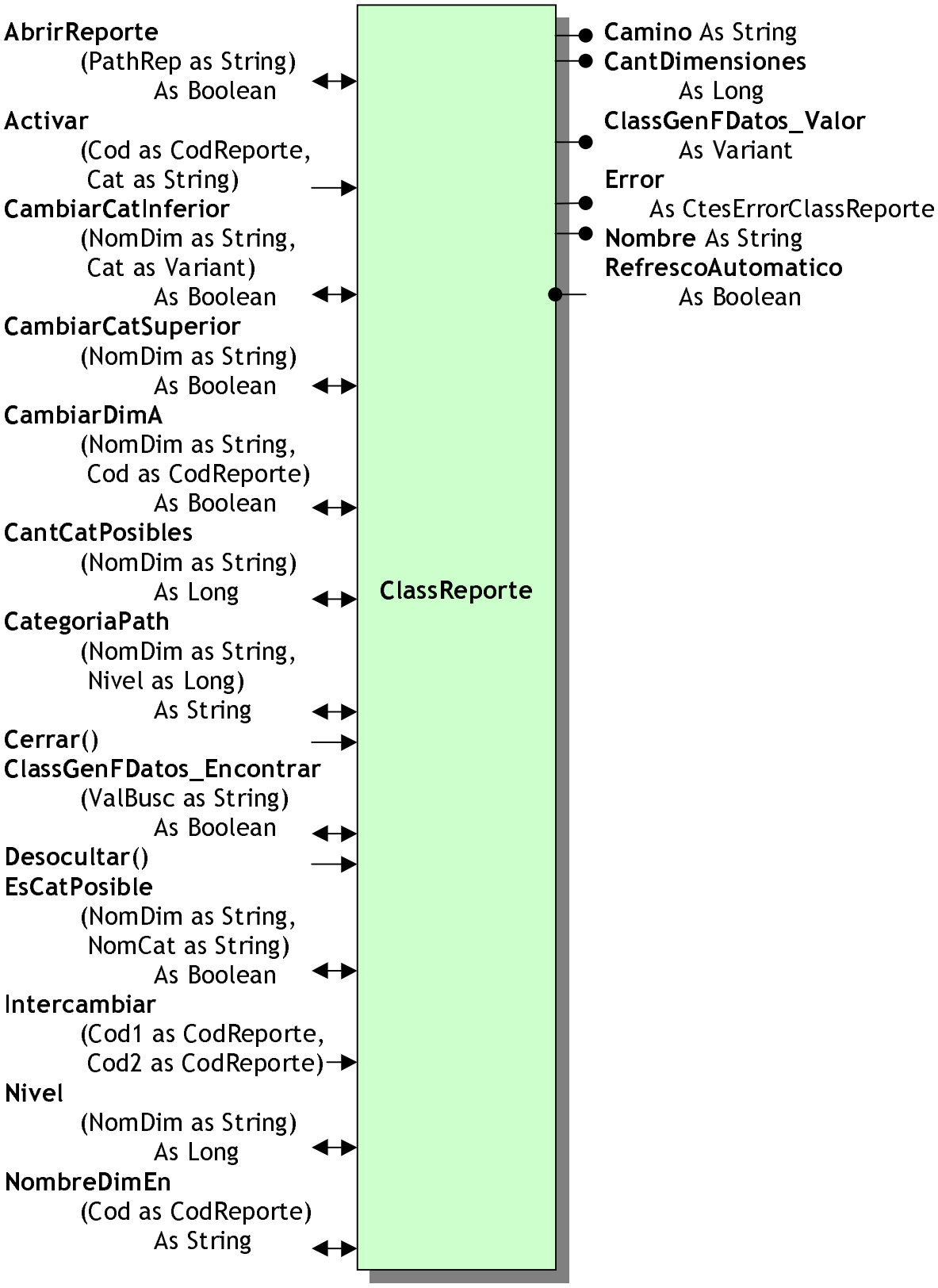
**ObjColoreo** As ClassGenColoreo

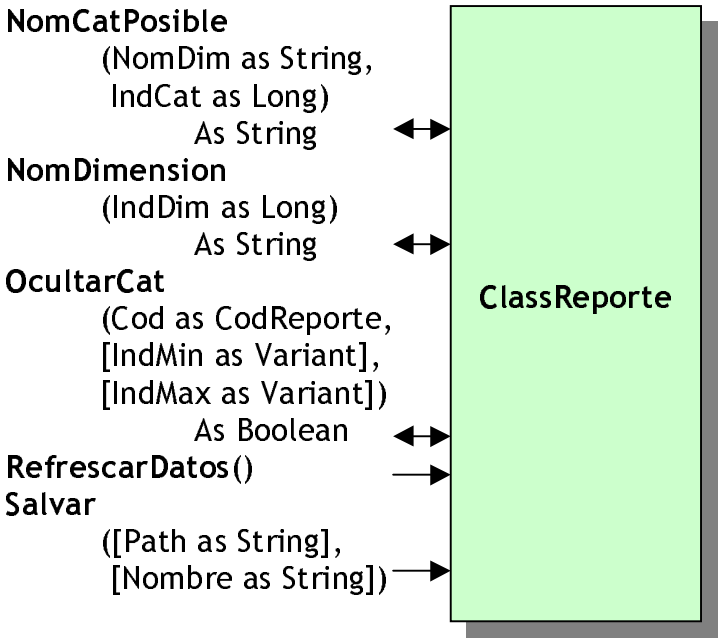
Establece el objeto ClassGenColoreo que proporcionará los objetos Renderer para la coloración de los mapas en el método **Coloreo**, tomando los valores del objeto ClassGenFDatos y los colores del objeto ClassGenRColores asociados a dicho objeto. (Ver Interfases)

**ShapeActual** As Object

Devuelve el objeto shape asociado a la última zona seleccionada. A partir de dicho objeto se puede obtener cierta propiedades como el enfoque, permitiendo establecer el enfoque del objeto MapControl usado, al mismo. El objeto shape puede ser usado para crear otro que defina el perímetro de una zona, y usando el método **DrawShape**, del objeto MapControl, sobresaltar el perímetro de la zona que se ha seleccionado.

### 4.3.2 Clase encargada de manejar el reporte (ClassReporte)






4.3.2.1 Métodos:

**AbrirReporte**(PathRep as String) As Boolean

Dado el path a un reporte de PowerPlay, lo abre, pudiendo así aplicar las operaciones disponibles en la clase ClassReporte.
Si no se produce ningún error devuelve True, sino False.

**Activar**(Cod as CodReporte, Cat as String)

Activa una columna, fila o layer, asociado a la categoría.

**CambiarCatSuperior**(NomDim as String, Cat as Variant) As Boolean

Cambia al nivel superior en la dimensión con la categoría especificada, la categoría puede ser especificada por el nombre (String) o por el índice de la misma (Long).
Si no se produce ningún error devuelve True, sino False.

**CambiarCatInferior**(NomDim as String) As Boolean

Cambia al nivel inferior en la dimensión especificada.
Si no se produce ningún error devuelve True, sino False.

**CambiarDimA**(NomDim as String, Cod as CodReporte) As Boolean

Cambia la dimensión especificada a una columna, fila o layer, según el valor de Cod.
Si no se produce ningún error devuelve True, sino False.

**CantCatPosibles**(NomDim as String) As Long

Devuelve la cantidad de categorías que existen en el nivel actual de la dimensión especificada.
Si se produce un error devuelve -1.

**CategoriaPath**(NomDim as String, Nivel as Long) As String

Dada una dimensión y un nivel, que debe ser menor o igual al nivel actual de dicha dimensión, devuelve la categoría que fue seleccionada a dicho nivel.
Si se produce un error devuelve el string vacío.

**Cerrar**()

Cierra el reporte actualmente abierto.

**ClassGenFDatos_Encontrar**(ValBusc as String) As Boolean

Este método forma parte de la interfase ClassGenFDatos implementada tmabién por esta clase. (Ver Interfases)
El valor que se devuelve se puede obtener a través de la propiedad **ClassGenFDatos_Valor**.
Si no se produce ningún error devuelve True, sino False.

**Desocultar**()

Desoculta todas las categorías actualmente ocultas.

**EsCatPosible**(NomDim as String, Cat as String) As Boolean

Devuelve True si la categoría es una de las actuales de dicha dimensión, sino devuelve False.

**Intercambiar**(Cod1 as CodReporte, Cod2 as CodReporte)

Intercambia filas con columnas, filas con layers o columnas con layers, de acuerdo a la combinación de parámetros con que se invoque.

**Nivel**(NomDim as String) As Long

Devuelve el nivel actual de la dimensión.
Si se produce un error devuelve -1.

**NombreDimEn**(Cod as CodReporte) As String

Devuelve el nombre de la dimensión que se encuentra en las filas, columnas o layers, de acuerdo al valor del parámetro con que se invoca.

**NomCatPosible**(NomDim as String, IndCat as Long) As String

Devuelve el nombre de la categoria correspondiente al indice que se encuentra en el nivel actual de la dimensión. El índice debe ser menor o igual al valor dado por el método **CantCatPosibles**.
Si se produce un error devuelve el string vacío.

**NomDimension**(IndDim as Long) As String

Devuelve el nombre de la dimensión correspondiente al valor del índice. Dicho índice debe ser menor o igual al valor dado por la propiedad **CantDimensiones**.
Si se produce un error devuelve el string vacío.

**OcultarCat**(Cod as CodReporte, [IndMin as Variant], [IndMax as Variant]) As Boolean

Oculta las categorías ubicadas en las filas, columnas o layers, de acuerdo al valor especificado. Las categorías se pueden especificar dando un índice o nombre inicial de categoría, y/o un índice o nombre final de categoría. En caso de no especificar ningún valor se ocultará toda las categorías correspondientes.
Si no se produce ningún error devuelve True, sino devuelve False.

**RefrescarDatos**()

Refresca los valores del reporte.

**Salvar**([Path as String],[Nombre as String])

Salva el estado del reporte actualmente abierto. El camino donde salvar y el nombre del reporte se pueden especificar, si se omite cualqiuera de ellos se usará el actual.

#### 4.3.2.2 Propiedades:

**Camino** As String

Devuelve el camino actual al reporte actualmente abierto.

**CantDimensiones** As Long

Devuelve la cantidad de dimensiones del reporte actualmente abierto.

**ClassGenFDatos_Valor** As Variant

Esta propiedad pertenece a la interfase ClassGenFDatos, implementada por esta clase. (Ver Interfases)
Devuelve el valor obtenido luego de la última invocación al método **ClassGenFDatos_Encontrar**.

**Error** as CtesErrorClassReporte

Devuelve el código de error corresondiente a la última operación realizada.

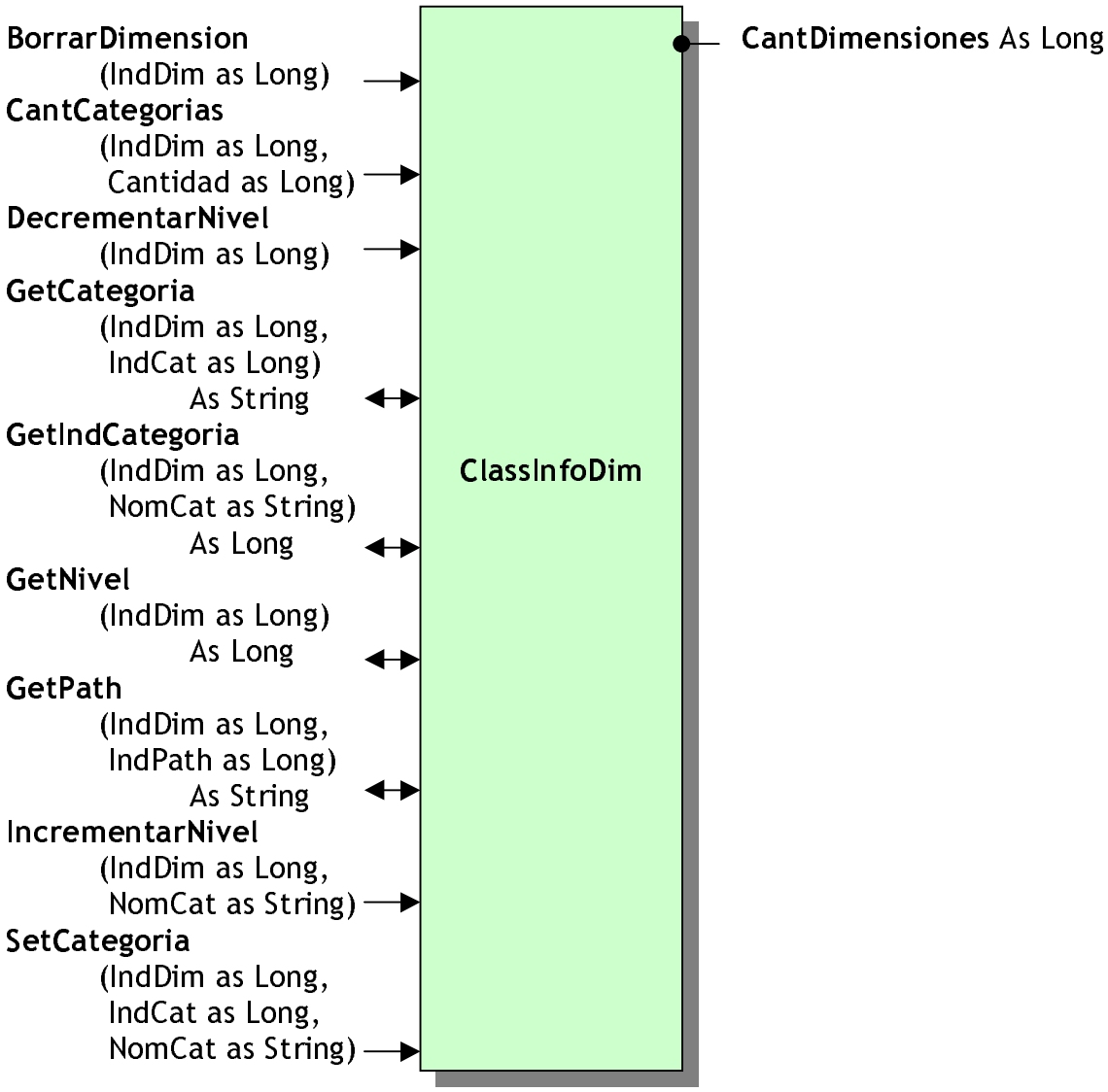
**Nombre** As String

Devuelve el nombre del reporte actualmente abierto.

**RefrescoAutomatico** As Boolean

Especifica si el refresco de los valores del reporte va a ser automático o no.

### 4.3.3 Clase encargada de manejar la estructura interna usada por ClassReporte (ClassInfoDim)



#### 4.3.3.1 Métodos

**BorrarDimension**(IndDim as Long)

Elimina la información de la dimensión correspondiente a dicho índice, o sea, queda con nivel 0 y 0 categorias.

**CantCategorias**(IndDim as Long, Cantidad as Long)

Especifica la cantidad de categorias que va a tener la dimensión correspondiente a dicho índice, al nivel que se encuentra actualmente.

**DecrementarNivel**(IndDim as Long)

Decrementa el nivel actual de la dimension correspondiente a dicho índice, la cantidad de categorias queda en 0. Asi mismo se actualiza el camino actual al nuevo nivel de dicha dimensión. Si el nivel actual es 1, la operación no modifica nada.

**GetCategoria**(IndDim as Long, IndCat as Long) As String

Devuelve el nombre de la categoria correspondiente al índice de categoria, perteneciente a la dimensión dada por el índice, en el nivel actual de dicha dimensión.
Si se produce un error devuelve el string vacío.

**GetIndCategoria**(IndDim as Long, NomCat as String) As Long

Devuelve el índice de la categoria correspondiente al nombre de categoria, perteneciente a la dimensión dada por el índice, en el nivel actual de dicha dimensión. El valor de los índices va de 1 hasta la cantidad de categorías especificadas al nivel actual de dicha dimensión.
Si se produce un error devuelve 0.

**GetNivel**(IndDim as Long) As Long

Devuelve el nivel actual de la dimensión correspondiente al índice.
Si se produce un error devuelve -1.

**GetPath**(IndDim as Long, IndPath as Long) As String

Devuelve el nombre de la categoría correspondiente al índice, perteneciente al camino al nivel actual de la dimensíon correspondiente al índice especificado.
Si se produce un error devuelve el string vacío.

**IncrementarNivel**(IndDim as Long, NomCat as String)

Incrementa el valor del nivel de la dimensión correspondiente al índice especificado, agregando en el camino al nuevo nivel de dicha dimensión el nombre de la categoría especificada. La cantidad de categorías queda en 0.

**SetCategoria**(IndDim as Long, IndCat as Long, NomCat as String)

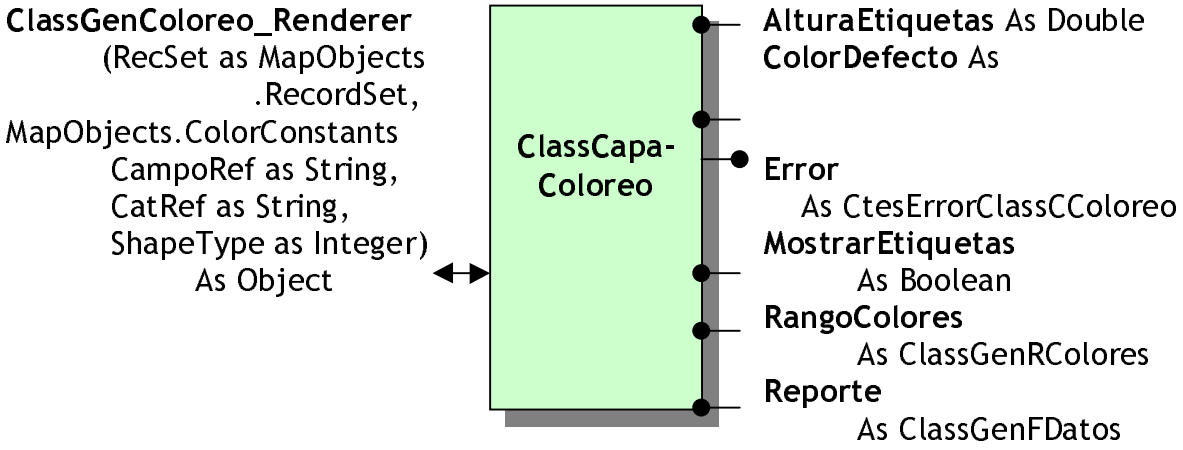
Especifica el nombre de la categoría correspondiente al índice dado, perteneciente a la dimensión correpsondiente al índice dado, y en el nivel actual. Si el índice dado es incorrecto la operación no realiza nada.

#### 4.3.3.2 Propiedades

**CantDimensiones** As Long

Especifica la cantidad de dimensiones que van a existir en la clase. En caso de usarse esta propiedad cuando ya habían dimensiones especificadas, se elimina toda información existente a las mismas.

### 4.3.4 Clase encargada de colorear los layers (ClassCapaColoreo)




#### 4.3.4.1 Métodos

**ClassGenColoreo_Renderer**(RecSet as MapObjects.RecordSet, CampoRef as String, CatRef as String, ShapeType as Integer) As Object

Este método pertenece a la interfase ClassGenColoreo, implementada por esta interfase. (Ver Interfases)
Dado un RecordSet de un layer, devuelve un objeto de tipo Renderer que tiene asignado a las zonas especificadas por el campo CampoRef, los colores correspondientes de acuerdo a los valores asociados a las categorías en el campo CatRef. En el caso de estar habilitada la propiedad **MostrarEtiquetas**, se definirán en el Renderer las etiquetas, de acuerdo al campo CatRef, de aquellas zonas seleccionadas en el RecordSet.

#### 4.3.4.2 Propiedades

**AlturaEtiquetas** As Double

Especifica el tamaño de las etiquetas asociados a los valores del Campo CatRef, que se definirán al aplicar el método **ClassGenColoreo_Renderer** al RecordSet especificado, y si está habilitada la propiedad **MostrarEtiquetas**.

**ColorDefecto** As MapObjects.ColorConstants

Especifica el color por defecto para aquellas zonas cuyos valores no están dentro de ningún rango, en caso de que no se haya especificado el objeto ClassGenRColores. Si no se especifica el color por defecto será el blanco.

**MostrarEtiquetas** As Boolean

Especifica si se mostrarán las etiquetas al aplicar el método **ClassGenColoreo_Renderer** a aquellos valores del campo CatRef del RecordSet especificado.

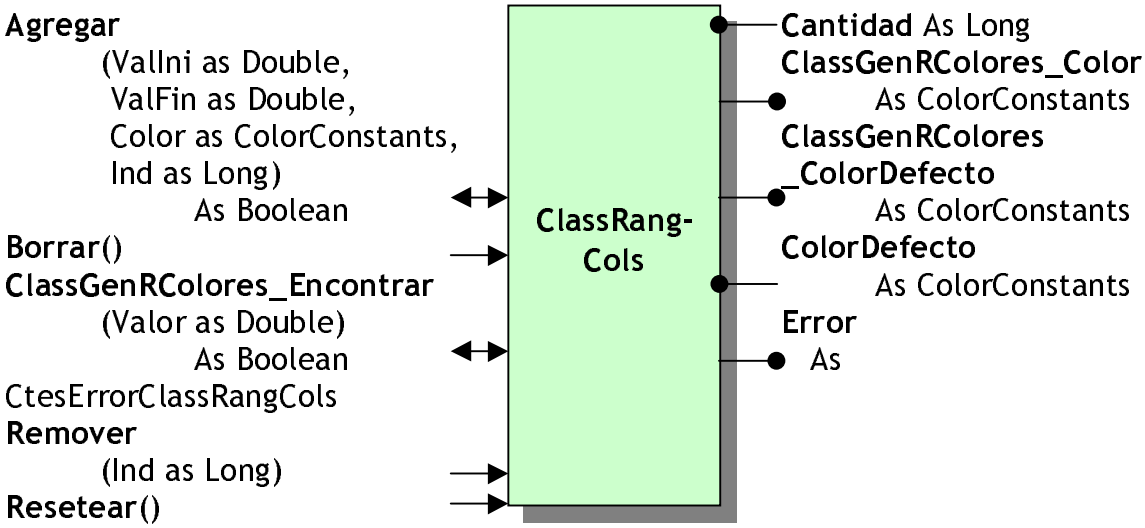
**RangoColores** As ClassGenRColores

Especifica el objeto que proveera los colores asociados a rangos de valores, usado para el coloreo. (Ver Interfases)

**Reporte** As ClassGenFDatos

Especifica el objeto que proveera los valores asociados a las zonas, que determinarán los colores de acuerdo a los rangos definidos.
(Ver Interfases)

### 4.3.5 Clase encargada de manejar los rangos de valores asociados a colores (ClassRangCols)




#### 4.3.5.1 Métodos

**Agregar**(ValIni as Double, ValFin as Double, Color as ColorConstants, Ind as Long) As Boolean

Agrega un nuevo rango de valores, especificado por los valores iniciales y finales, y su color asociado asi como el índice correspondiente a dicho rango en el objeto.
Si no se pproduce ningún error devuelve True, sino False.

**Borrar**()

Borra todos los rangos actualmente definidos, asi como sus colores asociados. La cantidad de rangos definidos sigue siendo la misma que antes.

**ClassGenRColores_Encontrar**(Valor as Double)

Este método pertenece a la interfase ClassGenRColores, implementada por esta clase. (Ver Interfases)
Dado un valor devuelve True si existe algún rango al que pertenezca, sino False. El color asociado a dicho rango se puede obtener usando la propiedad **ClassGenRColores_Color**.

**Remover**(Ind as Long)

Elimina el rango de valores correspondiente al índice especificado.

**Resetear**()

Borra todos los rangos actualmente definidos, asi como sus colores asociados. La cantidad de rangos definidos queda en 0.

4.3.5.2 Propiedades

**Cantidad** As Long

Especifica la cantidad de rangos de valores que va a tener el objeto.

**ClassGenRColores_Color** As ColorConstants

Esta propiedad pertenece a la interfase ClassGenRColores, implementada por esta clase. (Ver Interfases)
Devuelve el color asociado al rango de valores donde se encontró el valor usando el método **ClassGenRColores_Encontrar**.

**ClassGenRColores_ColorDefecto** As ColorConstants

Esta propiedad pertenece a la interfase ClassGenRColores, implementada por esta clase. (Ver Interfases)
Devuelve el color por defecto asignado al objeto.

**ColorDefecto** As ColorConstants

Especifica el color por defecto para aquellos valores que no pertenecen a ningún rango definido.

## *4.4 Interfases definidas*

De forma de lograr la independencia entre las clases principales definidas, se definieron clases que sirven de interfase entre otras. Este es el mecanismo que ofrece Visual Basic, en su versión 5.0, para implementar la "herencia" entre clases.

Las clases usadas de interfase solo tienen definidas los cabezales de sus métodos y propiedades, de esta forma cualquier otra clase que quiera implementar dicha interfase debe declararlo, e implementar por lo menos todos los métodos y propiedades definidas en dicha interfase, luego se pueden agregar los métodos y propiedades propios de la clase.

Las interfases definidas son estas:

### 4.4.1 ClassGenFDatos*:*

Interfase entre las clases ClassCapaColoreo y ClassReporte, en este caso la clase ClassReporte es quien implementa esta interfase. De esta forma se logra la independencia del origen de los datos que serán usados para la generación de los objetos Renderer, los cuales permiten colorear los mapas.

4.4.1.1 Métodos:

**Encontrar**(ValBusc as String) As Boolean

Dado un valor a buscar, devuelve True si se pudo encontrar dicho valor, sino devuelve False.

4.4.1.2 Propiedades:

**Valor** As Variant

Devuelve el valor asociado a la última búsqueda realizada con exito, luego de aplicar el método **Encontrar**.

### 4.4.2 ClassGenColoreo

Interfase entre las clases ClassCapas y ClassCapaColoreo, en este caso la clase ClassCapaColoreo es quien implementa esta interfase. De esta forma se logra la independencia de como se generan los objetos Renderer usados para el coloreo de los mapas.

4.4.2.1 Métodos:

**Renderer**(RecSet as MapObjects.Recordset, CampoUbic as String, CampoRef as String, ShapeType as Integer) As Object

Dado un conjunto de registros de una tabla asociada a un layer, e indicando cual es el campo que identifica la zona usado por el objeto Renderer para colorear, y cual es el campo que identifica en el origen de los datos dicha zona, asi como el tipo de layer que se trata, devuelve el objeto Renderer correspondientes.

### 4.4.3 ClassGenRColores

Interfase entre las clases ClassCapaColoreo y ClassRangCols, en este caso la clase ClassRangCols es quien implementa esta interfase. De esta forma se logra la independencia de como se manejan los rangos de valores asociados a colores.

4.4.3.1 Métodos:

**Encontrar**(Valor as Double) As Boolean

Dado un valor a buscar en los rangos definidos, devuelve True si se encuentre en alguno de ellos, sino False.

Propiedades:

**Color** As ColorConstants

Devuelve el color asociado al rango de valores donde se encontró el valor a la última aplicación del método **Encontrar**.

**ColorDefecto** As ColorConstants

Devuelve el color usado por defecto en caso de que un valor no se encuentre en ninguno de los rangos definidos.

# 5 Conclusiones

## *5.1 Resumen*

En este proyecto se ha analizado la integración de herramientas para el manejo de Sistemas de Información Geográfica (GIS), con herramientas de consulta a Data Warehouse (OLAP).
El estudio se enfoca en dos aplicaciones existentes, de estas herramientas:

- MapObjects de ESRI como herramienta GIS.
- PowerPlay de COGNOS como herramienta OLAP.

Cada una de estas herramientas proveen componentes de software, para la integración con otras aplicaciones:

- MapObjects es un conjunto de componentes ActiveX.
- PowerPlay es un servidor OLE.

Se distinguieron y estudiaron dos formas de integración posible entre estas herramientas, considerando las características particulares que presentan las mismas:

- Integración a nivel de datos.
- Integración a nivel de aplicación.

### 5.1.1 Integración a nivel de datos

El objetivo a este nivel de integración era estudiar las posibilidades de que ambas herramientas, puedan compartir la información que manejan. Y para ello se tuvo que considerar el tipo y estructuración de la información, en ambas situaciones:

- Por un lado las herramientas GIS, manejan bases de datos geográficas, que en el caso de MapObjects tienen una estructura especial.
- Por otro, las herramientas OLAP manejan data warehouse.

Para la integración de la información almacenada en un data warehouse, en una base de datos geográfica, se plantearon diversos inconvenientes:

- La estructuración de la información de la base de datos geográfica, en MapObjects, no permite representar el complejo nivel de relacionamiento que tiene la información de un data warehouse. Para ello se debería trabajar a nivel de aplicación.
- No resulta particularmente de interés ya que se trata de pasar información de un data warehouse, a una base de datos relacional.

Cuando un data warehouse está pensado como un repositorio especial de información, con un alto nivel de análisis, que no permite un modelo relacional.

La integración de la información geográfica a un data warehouse, permite agregar nuevas dimensiones de análisis. Se distinguen en este tipo de información distintos aspectos geográficos, con diferentes propiedades. Estos aspectos fueron considerados para la definición de las dimensiones en el data warehouse.
La definición de estas dimensiones puede variar según como se organizen los distintos aspectos geográficos, permitiendo distintos tipos de análisis en cada caso:

- **Una única dimensión**: los aspectos se organizan reflejando la estructura que tienen a nivel geográfico. El análisis por esta dimensión está restringida a esa organización.
- **Varias dimensiones**: se definen dimensiones por cada combinación posible, de acuerdo a la geografía, de los distintos aspectos. La selección de categorías para el análisis en estas dimensiones, puede llevar a que no se obtengan resultados.

### 5.1.2 Integración a nivel de aplicación

La integración a nivel de aplicación permite alcanzar el mayor nivel posible de integración. Logrando combinar las funcionalidades de ambas herramientas, en nuevas funcionalidades que no son posibles de proveer con las herramientas por separado. Se combinan así las posibilidades de análisis OLAP y el análisis geográfico.

Para ello se determinó que debe haber un mínimo de integración a nivel de los datos, necesario por las características que tienen ambas herramientas. Luego se describió para cada sentido de interacción entre los componentes de cada herramienta, el procedimiento necesario, usando las funcionalidades respectivas, para lograr que ambas herramientas interactúen.
Se destacaron en detalle, los problemas de integración que se dan con los componentes de la herramienta OLAP, PowerPlay. Y se plantearon alternativas para superarlos, y permitir que se integren dichos componentes, de forma más apropiada, a otras aplicaciones.

### 5.1.3 Aplicaciones desarrolladas

Se presentaron dos aplicaciones basadas en la integración de la herramienta OLAP, PowerPlay, y la herramienta GIS, MapObjects:

- La primer aplicación tenía la finalidad de ser usada para un proyecto de la empresa ICA, en el análisis de la información censal, para una licitación para el censo de Brasil.
- La segunda aplicación tenía la finalidad de ser usada en forma amplia en distintos proyectos, por ejemplo para Coca-Cola, con distintas realidades donde se tengan que integrar data warehouses con información geográfica.

En ambas aplicaciones se aplicaron los estudios realizados para la integración tanto a nivel de datos como a nivel de aplicación.

#### 5.1.3.1 Componentes desarrollados:

**Objetivo:** Uno de los principales objetivos en el desarrollo de ambas herramientas era desarrollar un conjunto de componentes, que pudieran ser integradas fácilmente entre ellos y en cualquier aplicación, combinando funcionalidades GIS con funcionalidades OLAP, y viceversa.

Estos componentes fueron pensados como una capa de aplicación tanto sobre los componentes GIS, como los componentes OLAP. Permitiendo así, tener funcionalidades de mayor nivel de abstracción, y ocultando el manejo de cada uno de los componentes, en el desarrollo de nuevas aplicaciones.

| Componentes | Características |
|---|---|
| Manejo de interfase geográfica | • Independiencia de los criterios de asignación de atributos a los aspectos geográficos.<br>• Independencia del origen de los datos. |
| Manejo del reporte | • Funcionalidades de mayor nivel de abstracción. |
| Manejo de los objetos Renderer | • Independencia del origen de los datos.<br>• Indpendencia de los criterios para asignación de colores. |

### 5.1.4 Trabajos futuros

5.1.4.1 Extensión de las clases:

- En el desarrollo de la clase que maneja la interfase geográfica se tomó en cuenta solo el manejo de aspectos geográficos del tipo polígono. Pero se podría extender para poder manejar layers de todos los tipos, ya que por el lado del data warehouse se estudiaron distintas alternativas de definición de dimensiones geográficas, considerando todos los aspectos geográficos.

- Gracias al diseño de las clases, el origen de los datos para la asignación de atributos a los aspectos geográficos, puede ser variada:

  - Bases de datos relacionales.
  - Basees de datos relacionales en combinación con Data Warehouse.
  - De las propias bases de datos geográficas.
  - ...

- En la experiencia de las aplicaciones desarrolladas, se tuvieron que implementar los controles necesarios para poder realizar las operaciones básicas sobre los reportes de PowerPlay. Por ejemplo, los controles para modificar el nivel de análisis en cada una de las dimensiones, el contenedor OLE con las barras de scroll que permitieran recorrer el reporte como si se hiciera desde PowerPlay. Estos controles no son específicos de estas aplicaciones y tendrían que volverse a implementar en cualquier otra aplicación donde se quiera integrar un reporte.

  De forma de proveer componentes para integrar en aplicaciones, se podrían desarrollar controles ActiveX, que implimentarán los controles básicos para el manejo tanto de los reportes de PowerPlay, así como de la interfase geográfica. De esta forma se tendrían un conjunto de controles que acelerarían notablemente el desarrollo de nuevas aplicaciones.

  ***Observación:*** El conjunto de componentes que ofrece MapObjects, implementado como componentes ActiveX, permite una mejor integración de las funcionalidades GIS en otra aplicación. Mientras que los objetos OLE que ofrece PowerPlay, carecen de un componente visual, lo cual implica que deba implementarse en la propia aplicación con resultados inferiores a los que se podrían lograr.

#### 5.1.4.2 Actualización

En el momento de empezar este proyecto, las versiones existentes de las dos herramientas usadas para integrar eran:

- PowerPlay, versión 5.0
- MapObjects, versión 1.2

Al final del año surgía al mercado la versión 6.0 de PowerPlay, con nuevas prestaciones. Lamentablemente, las aplicaciones desarrolladas no funcionaron correctamente como con la versión 5.0. Lo cuál implica que se deberá estudiar y modificar la clase que maneja el reporte para que funcione con la nueva versión. Además al haber nuevas prestaciones, estas deberán integrarse, y pueden haber algunas que solucionen los problemas de integración estudiados.

Así mismo, para mediados del año 1999, se espera que la empresa ESRI saque al mercado una nueva versión de MapObjects, la versión 2.0. También implicará estudiar las nuevas prestaciones para agregarlas a la clase que maneja la interfase geográfica.

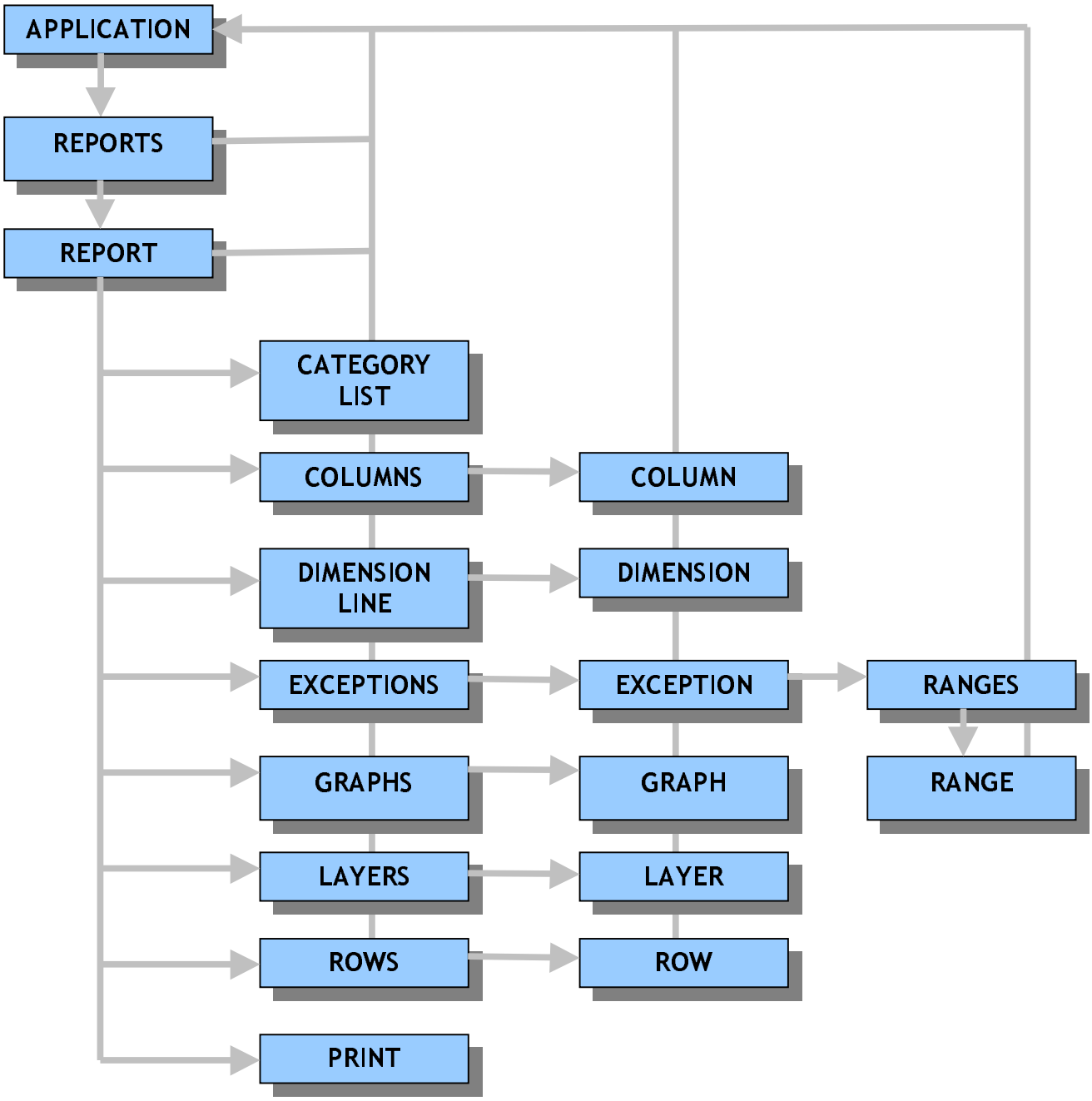
# 6 Referencias


- Información en Internet: "The Data Warehousing Information Center" (http://pwp.starnetinc.com/larryg/index.html)
- "Data Warehousing: Introducción a la tecnología y propuesta de *I.C.A.*", documentación elaborada por ICA.
- "Building Applications with MapObjects", manual de referencia publicado por ESRI (Environmental Systems Research Institute, Inc.).
- Manual en línea de CognosScript Editor, producto de COGNOS.

# 7 Apéndice

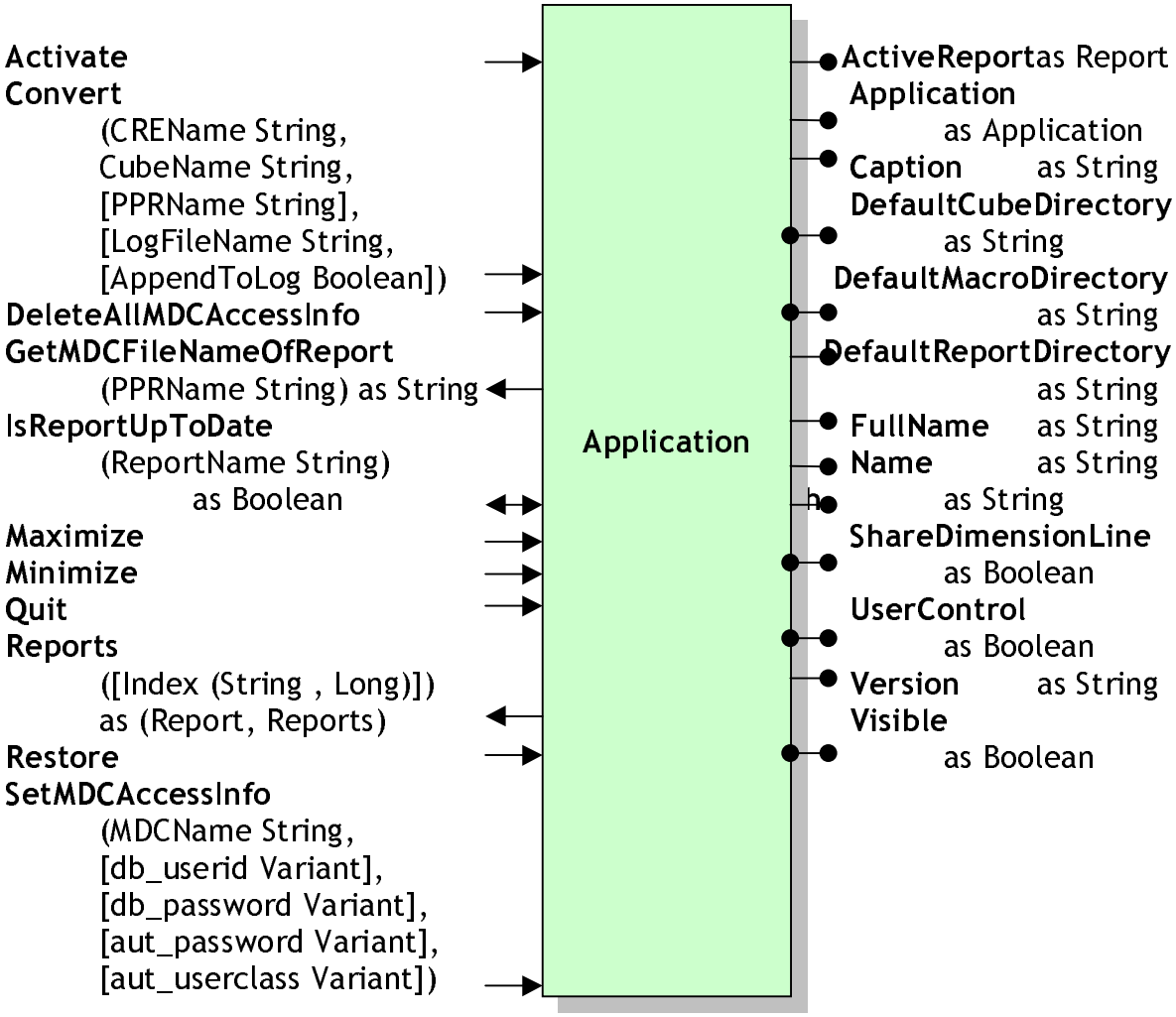
## *7.1 Diagrama de Objetos OLE de PowerPlay*

Este diagrama de los objetos que forman parte de los componentes de PowerPlay, fue elaborado a partir de la documentación existente, de forma de ser aprovechado para facilitar el entendimiento y uso de estos componentes.

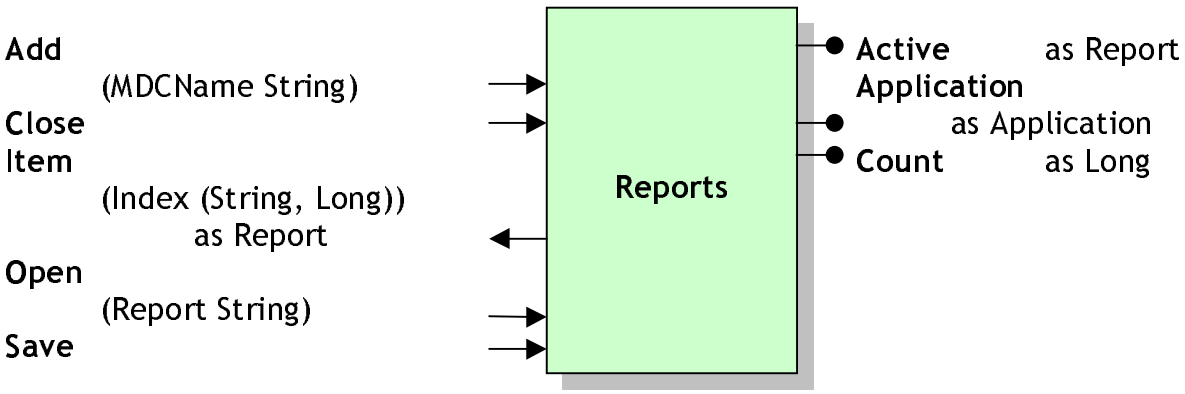


## *7.2 Especificación de los Objetos OLE de PowerPlay*

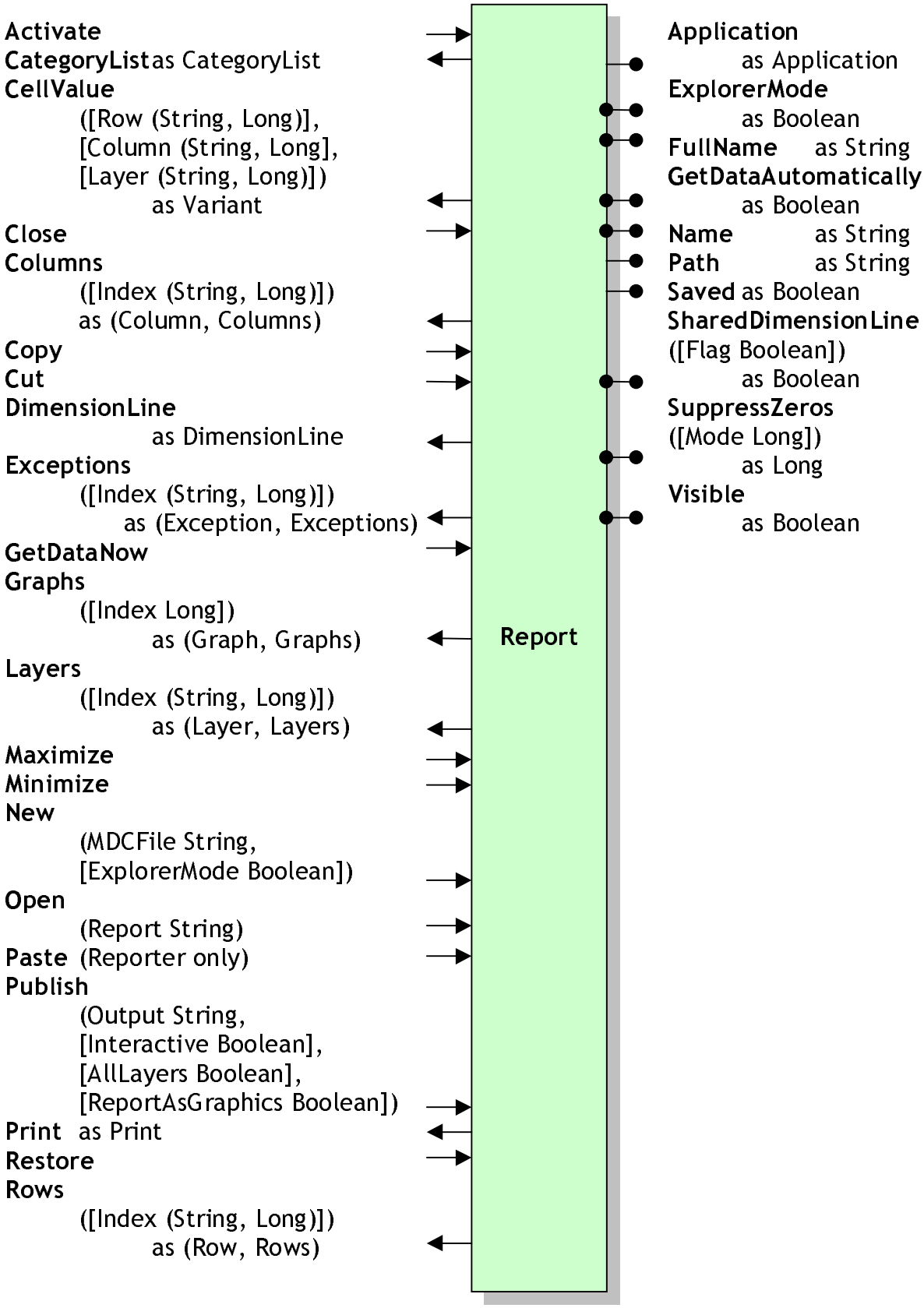
### 7.2.1 Clase Application

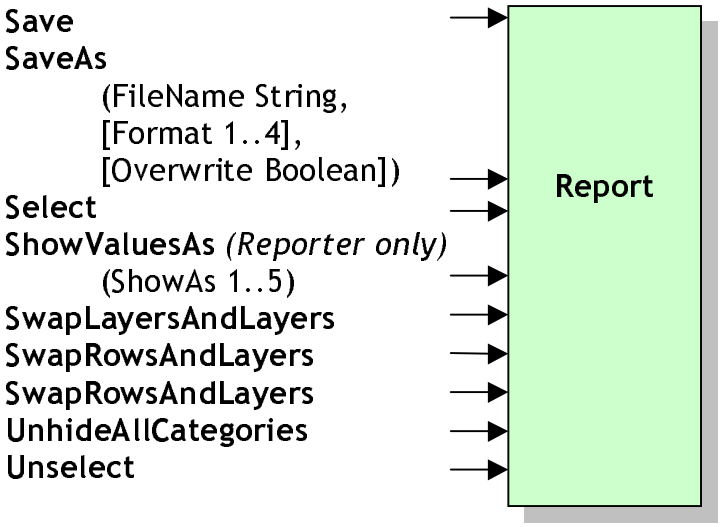


### 7.2.2 Clase Reports

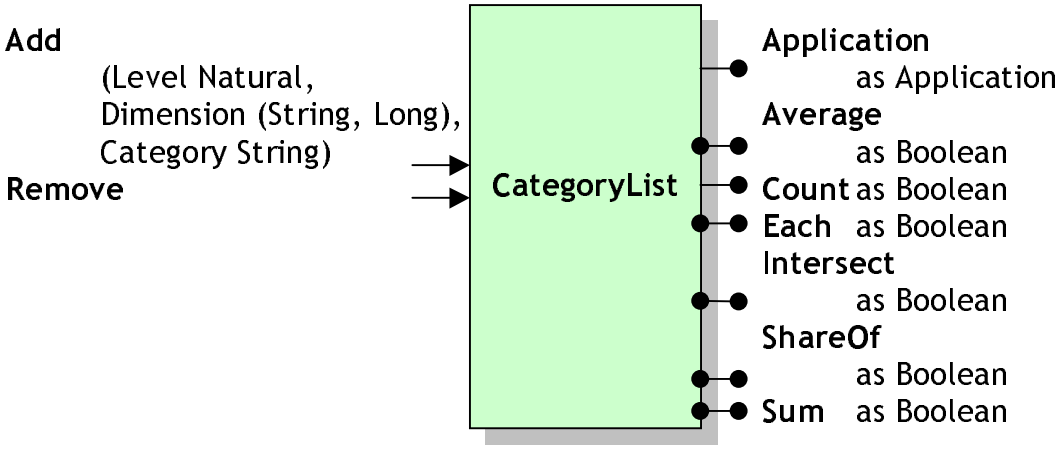


### 7.2.3 Clase Report

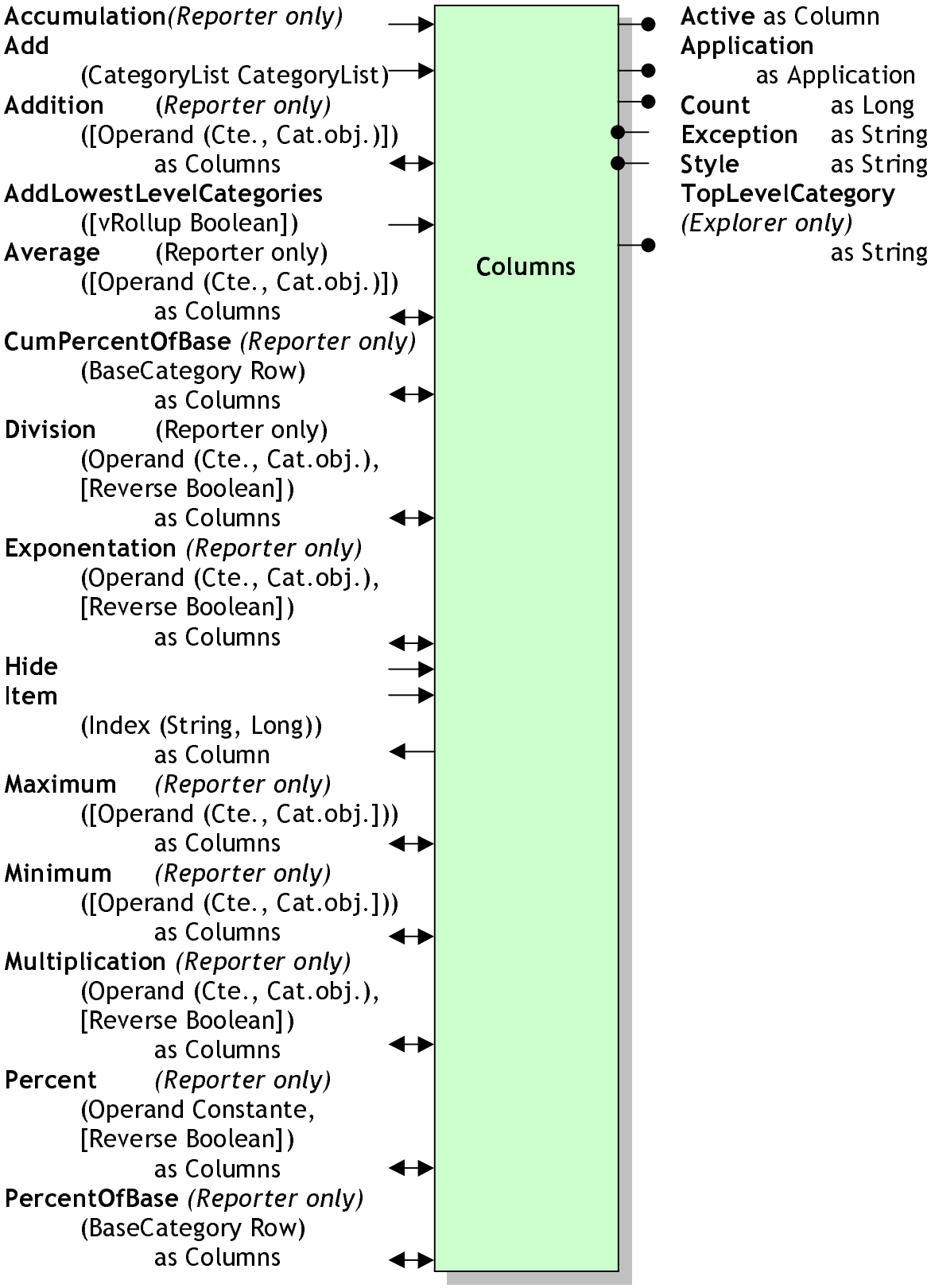


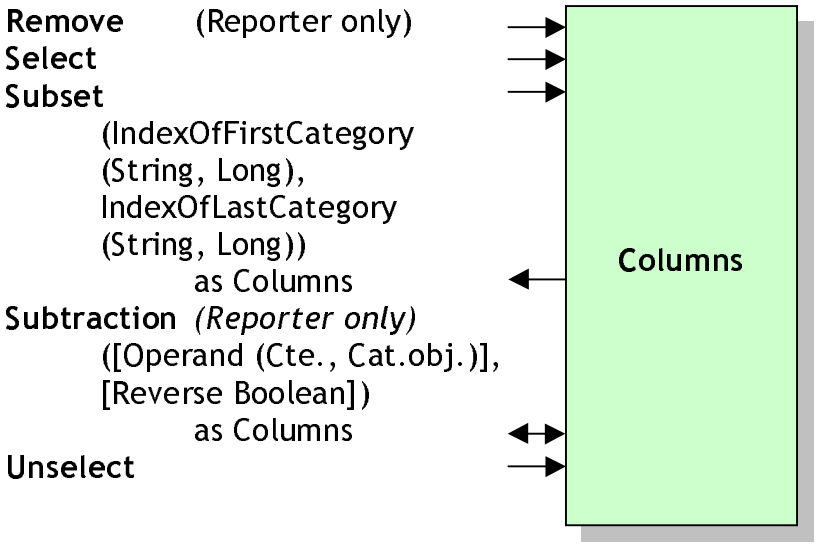

Save
SaveAs
(FileName String,
[Format 1..4],
[Overwrite Boolean])
Select
ShowValuesAs (Reporter only)
(ShowAs 1..5)
SwapLayersAndLayers
SwapRowsAndLayers
SwapRowsAndLayers
UnhideAllCategories
Unselect
Report

### 7.2.4 Clase CategoryList

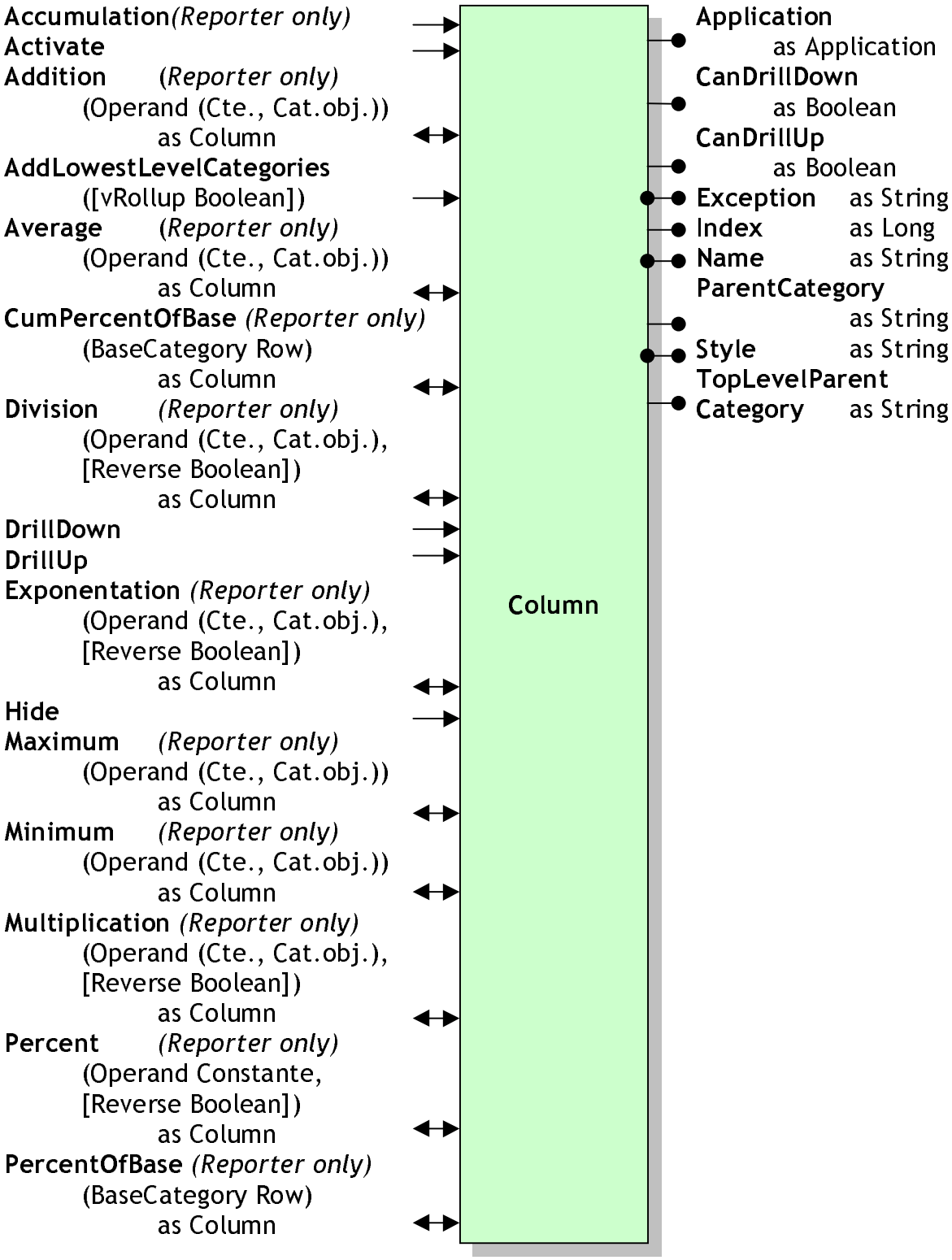


### 7.2.5 Clase Columns

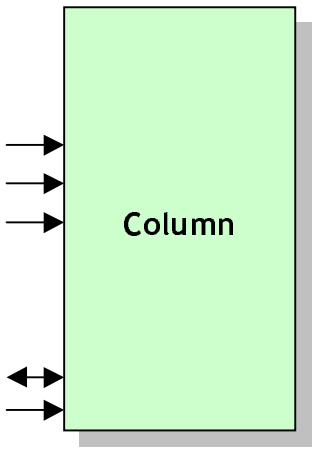



Remove (Reporter only)
Select
Subset
(IndexOfFirstCategory
(String, Long),
IndexOfLastCategory
(String, Long))
as Columns
Subtraction (Reporter only)
([Operand (Cte., Cat.obj.)],
[Reverse Boolean])
as Columns
Unselect
Columns

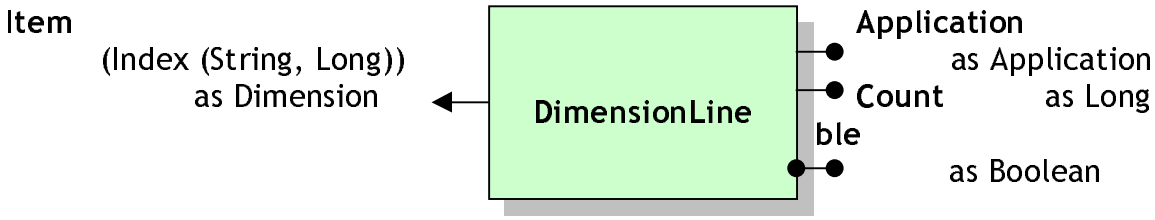
### 7.2.6 Clase Column



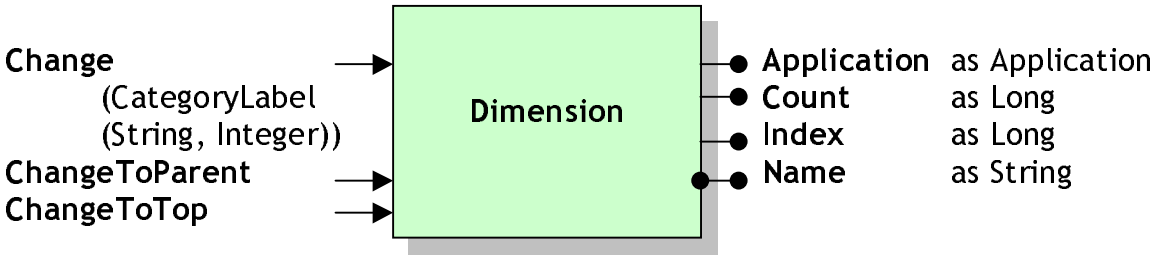
**Rank**
([ShowN Natural],
[PortionToShow 0..2],
[ShowRankCategory
Boolean])
**Remove** (Reporter only)
**Select**
**Subtraction** *(Reporter only)*
(Operand (Cte., Cat.obj.),
[Reverse Boolean])
as Column
**Unselect**



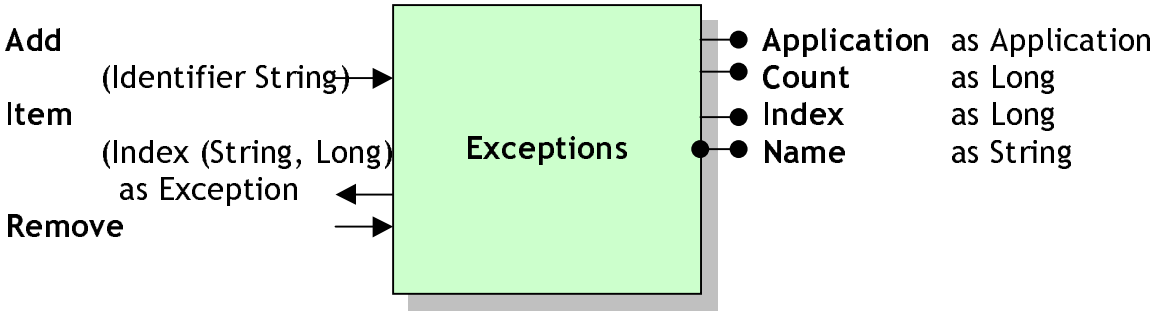
### 7.2.7 Clase DimensionLine



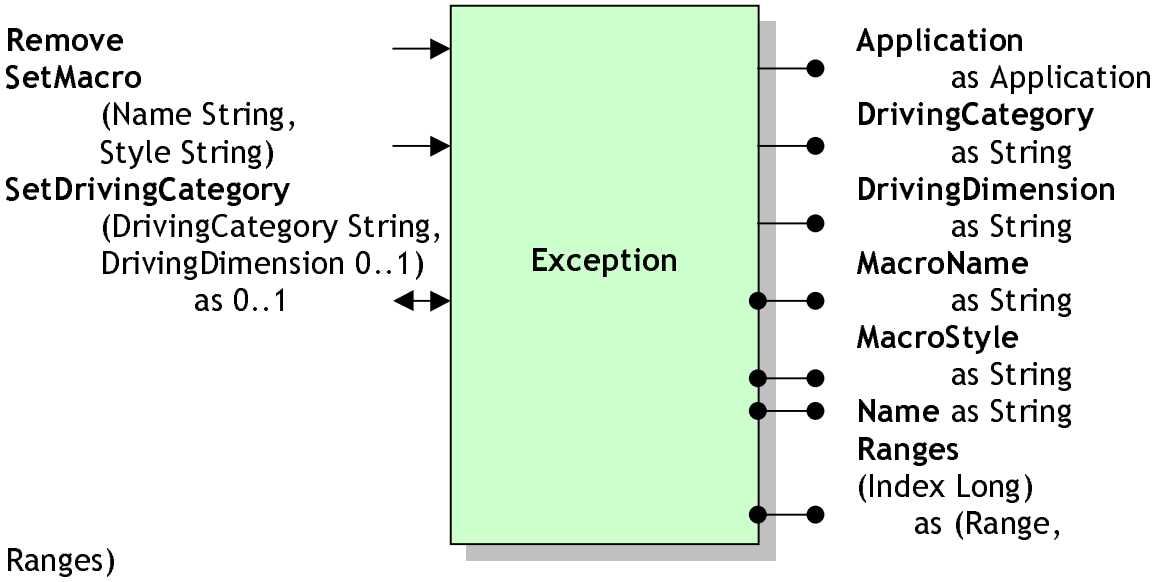
### 7.2.8 Clase Dimension



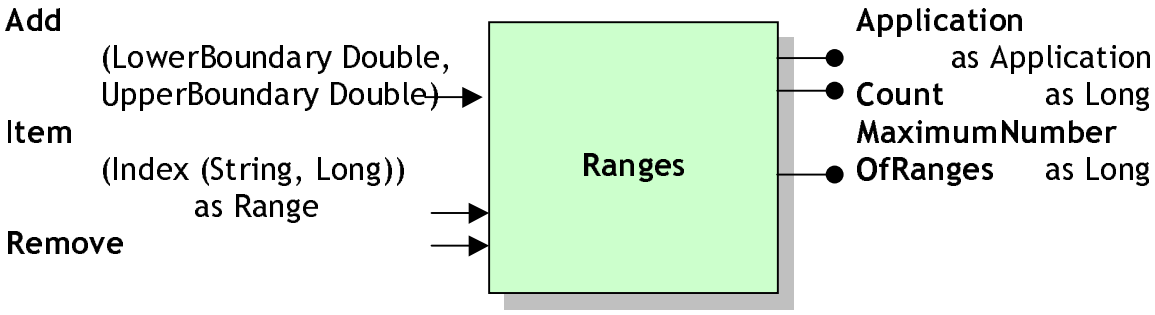
### 7.2.9 Clase Exceptions



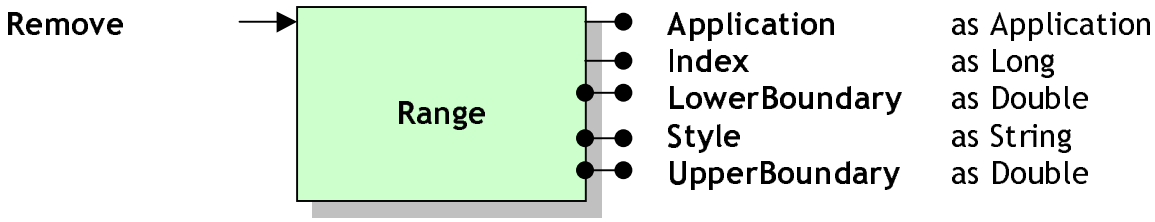
### 7.2.10 Clase Exception



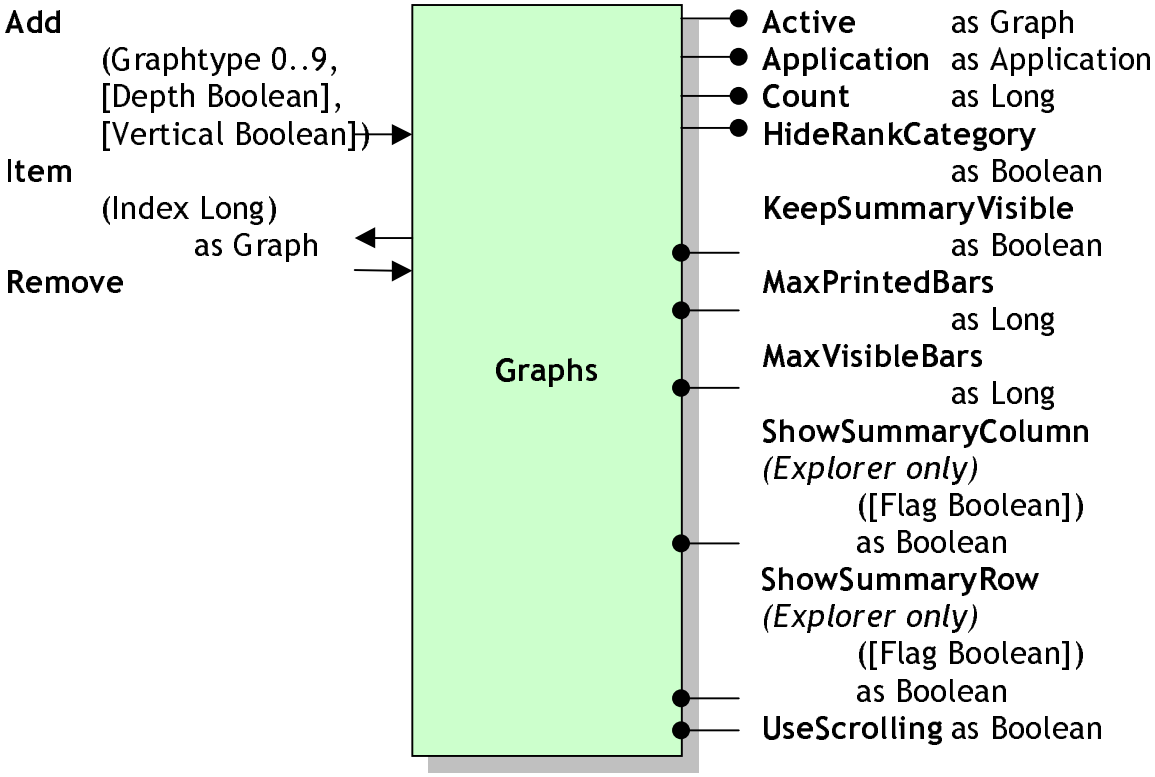
### 7.2.11 Clase Ranges



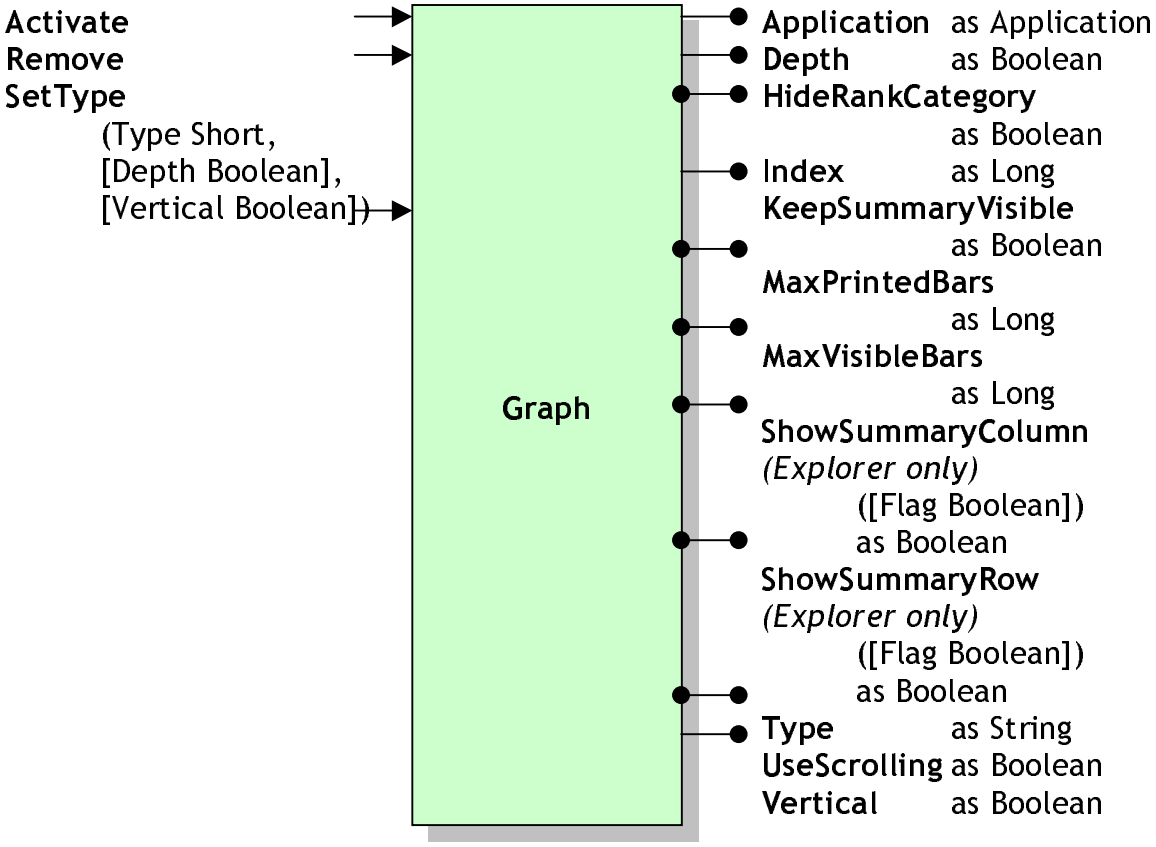
### 7.2.12 Clase Range



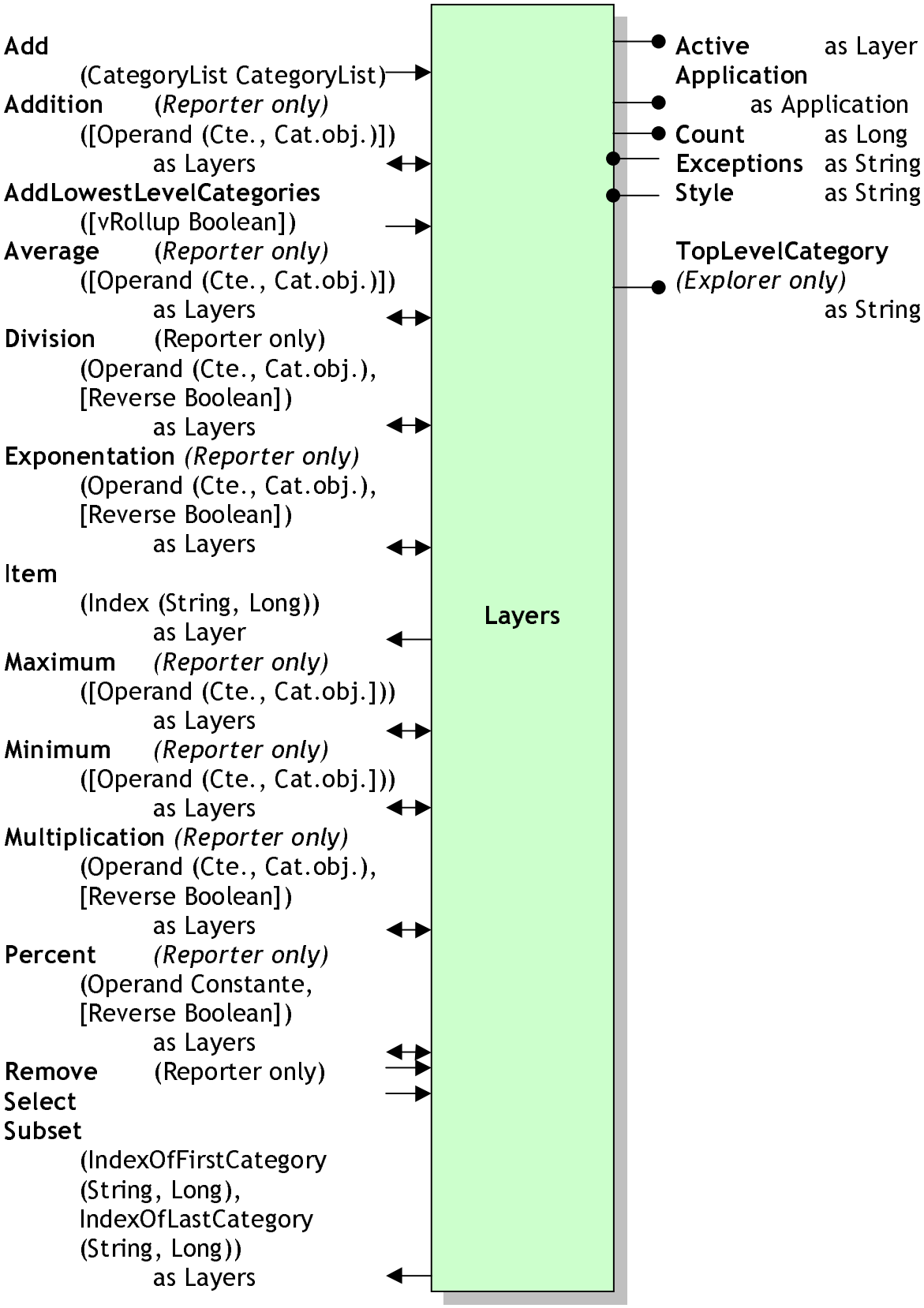
### 7.2.13 Clase Graphs



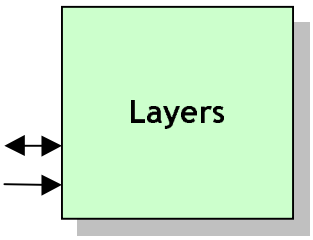
### 7.2.14 Clase Graph



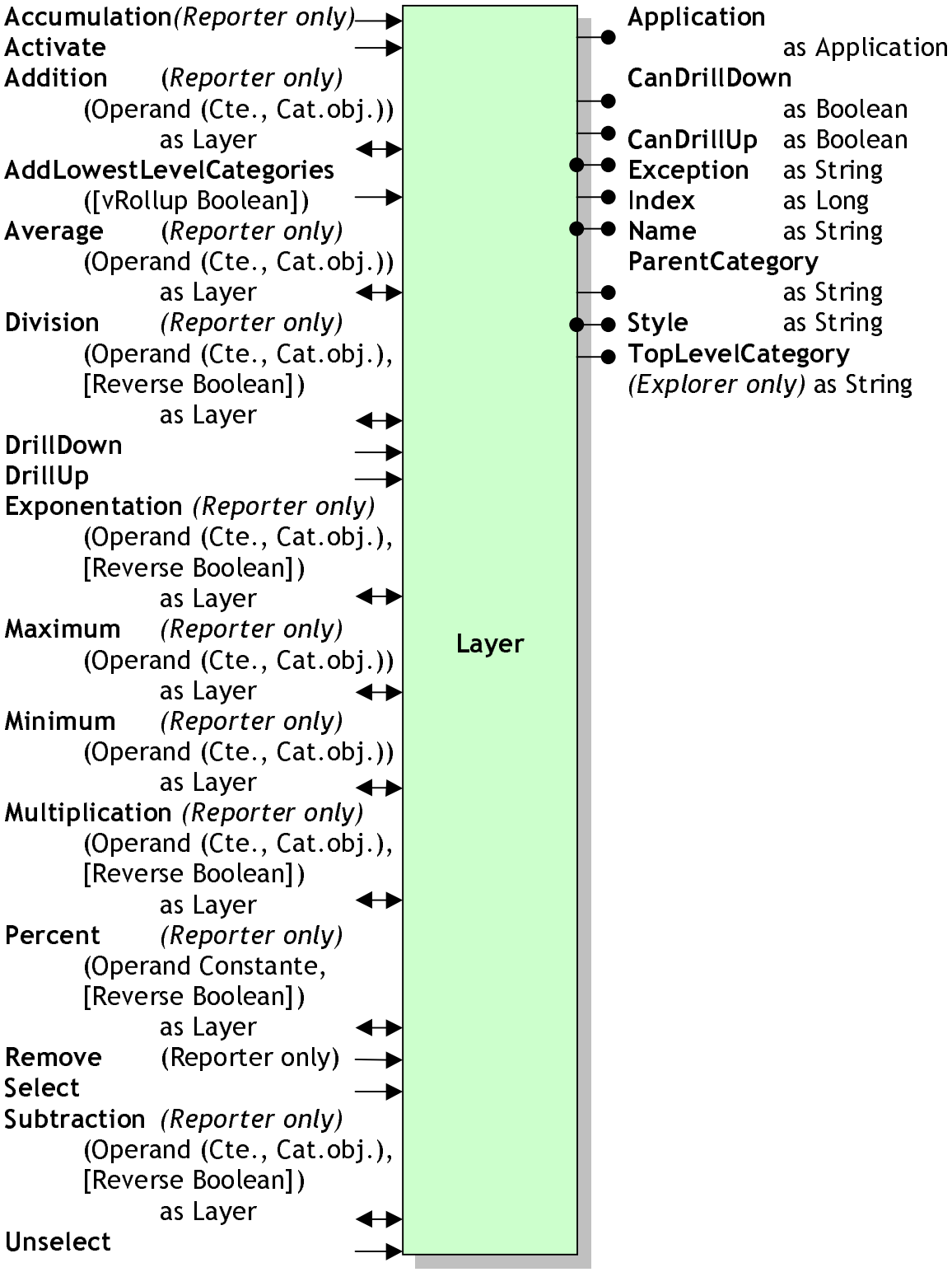
### 7.2.15 Clase Layers



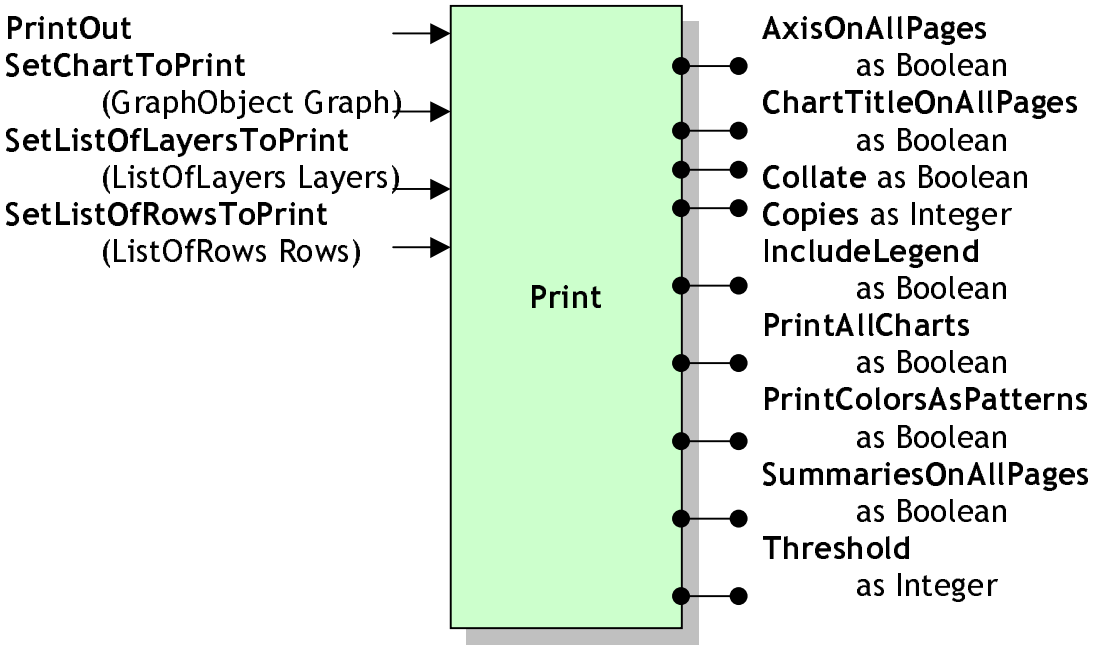
**Subtraction** *(Reporter only)*
([Operand (Cte., Cat.obj.)],
[Reverse Boolean])
as Layers
**Unselect**



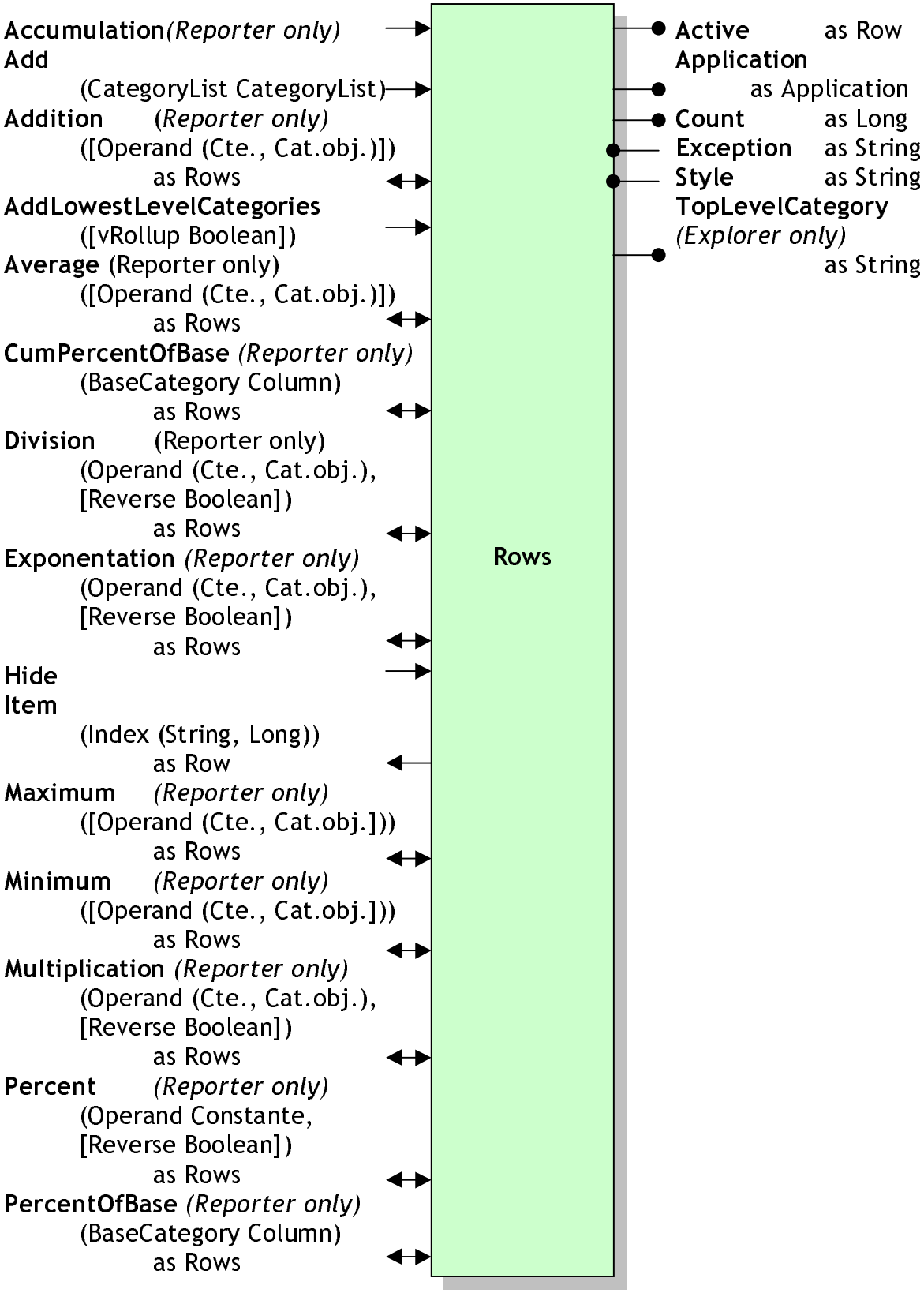
### 7.2.16 Clase Layer

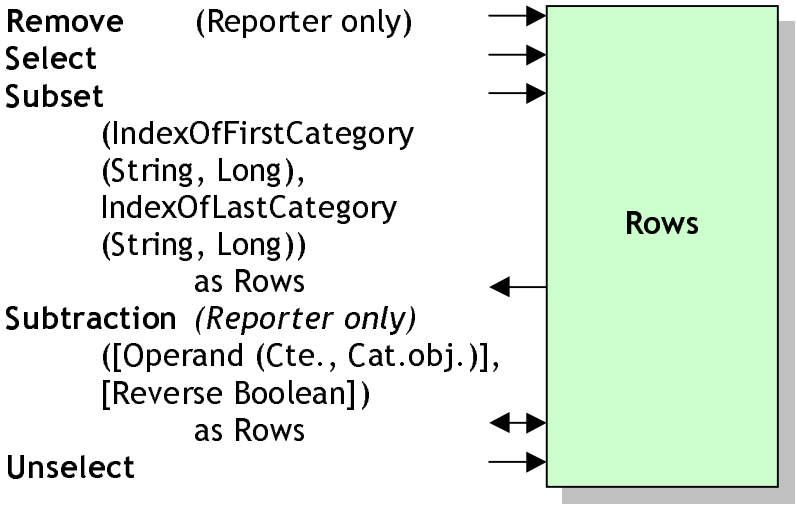


### 7.2.17 Clase Print

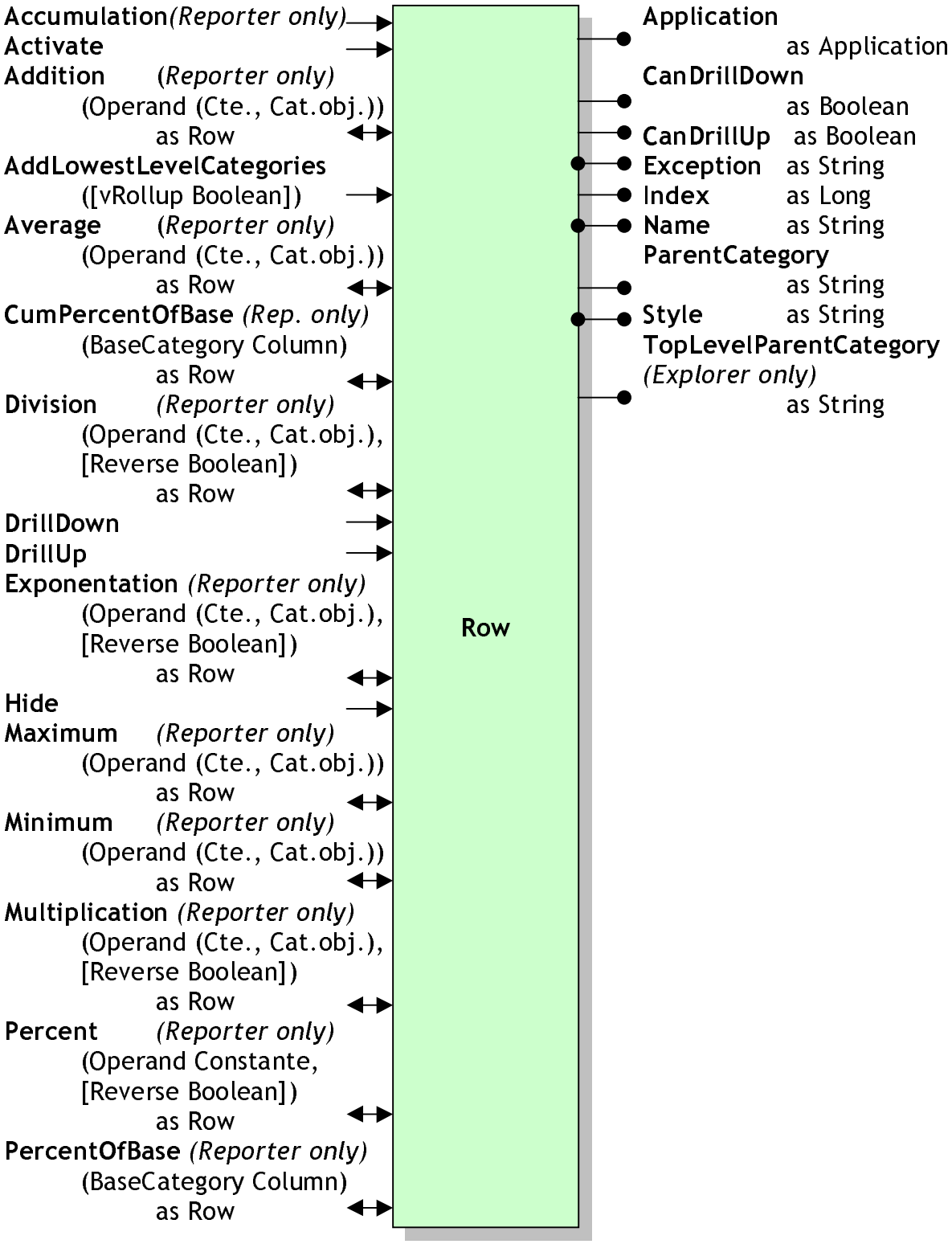


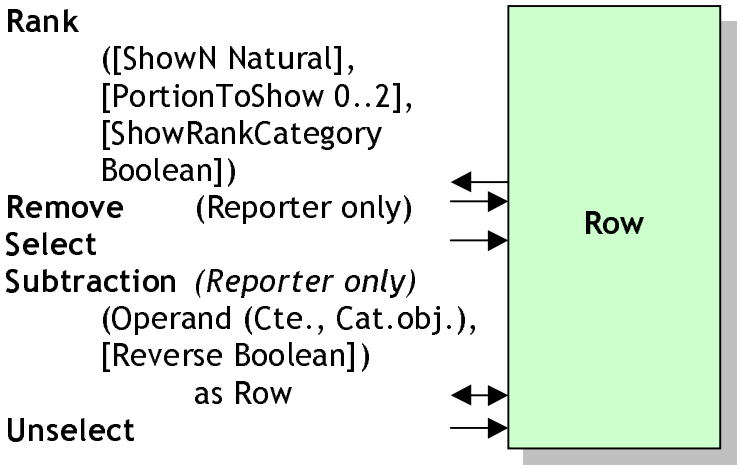
### 7.2.18 Clase Rows




Remove (Reporter only)
Select
Subset
(IndexOfFirstCategory
(String, Long),
IndexOfLastCategory
(String, Long))
as Rows
Subtraction (Reporter only)
([Operand (Cte., Cat.obj.)],
[Reverse Boolean])
as Rows
Unselect
Rows

### 7.2.19 Clase Row




Rank
([ShowN Natural],
[PortionToShow 0..2],
[ShowRankCategory
Boolean])
Remove (Reporter only)
Select
Subtraction (Reporter only)
(Operand (Cte., Cat.obj.),
[Reverse Boolean])
as Row
Unselect
Row