El Modelo de Geometrías

gvSIG: Avanzando Juntos

http://www.gvsig.gva.es
http://www.gvsig.org

Jorge Piera Llodrá
Iver T.I.
jpiera@gvsig.org

Índice de la Presentación

1. El modelo anterior.
2. Las normas ISO.
3. En modelo actual.
   - 3.1. Geometrías primitivas.
   - 3.2. Geometrías complejas.
   - 3.3. Geometrías múltiples.
4. Acceso al API de geometrías.

Índice de la Presentación

Índice de la Presentación

1. El Modelo Anterior

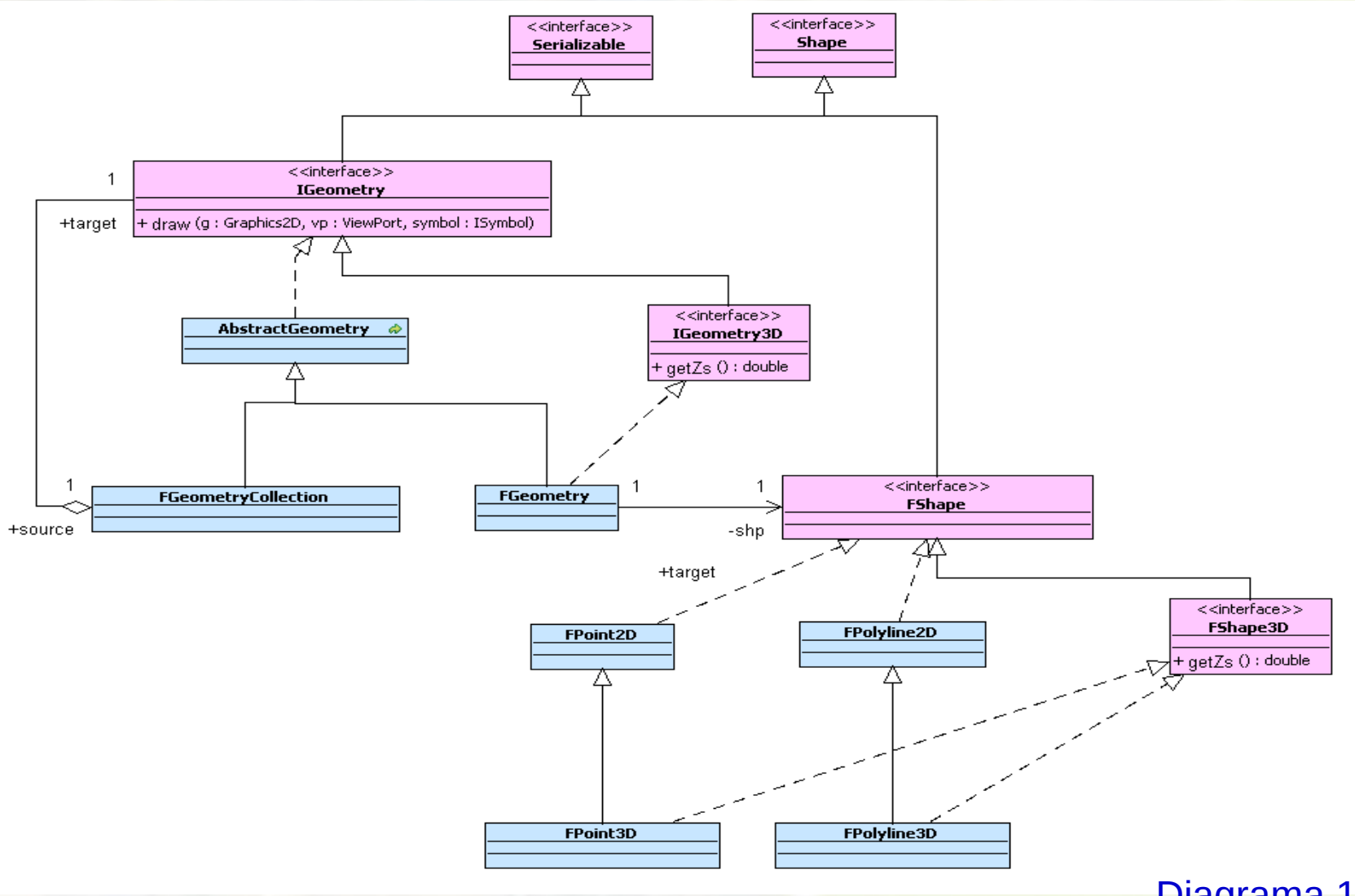
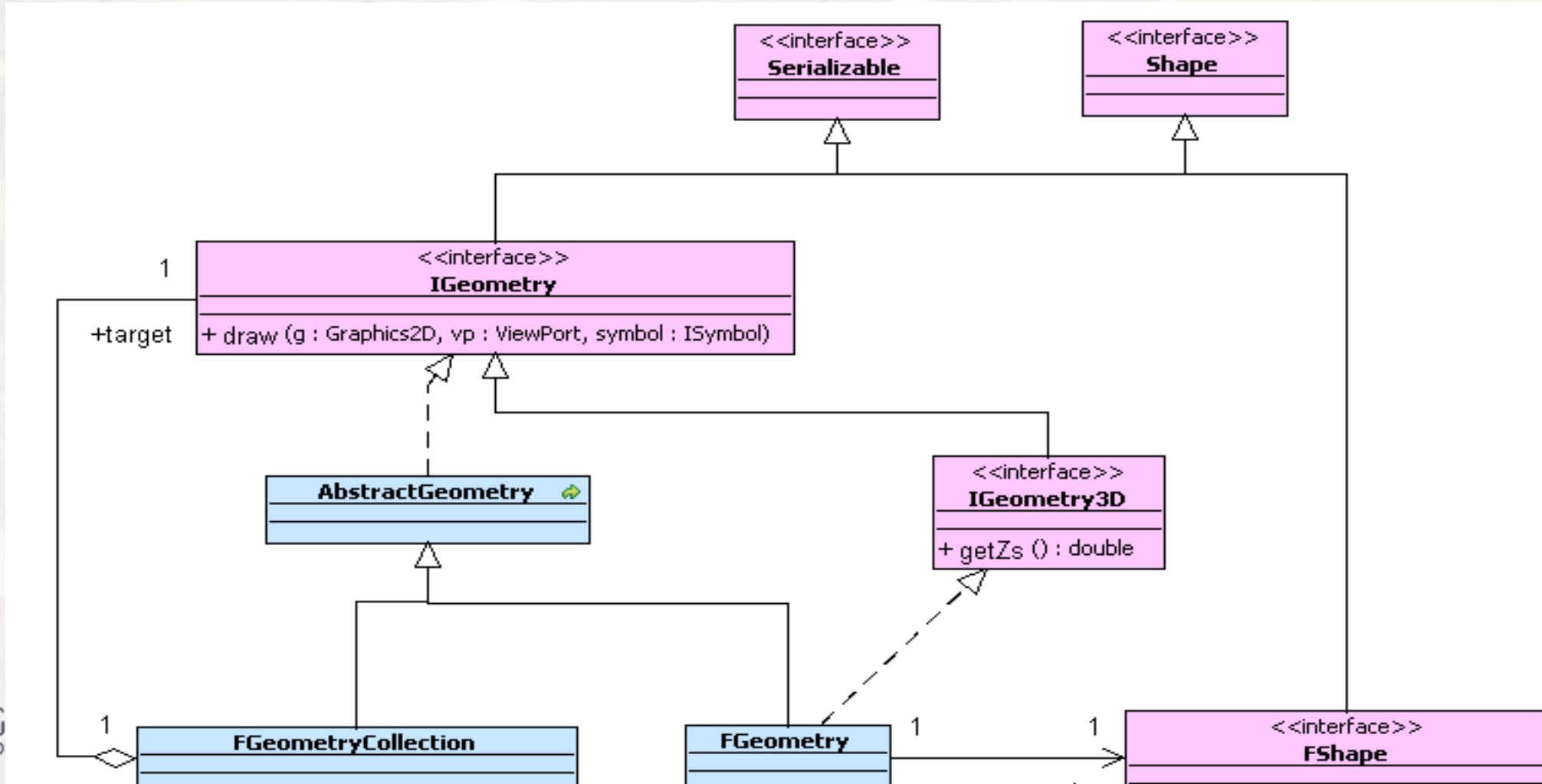


Diagrama 1

1. El Modelo Anterior

Dependencia con el dibujado (en 2D).



1. El Modelo Anterior

Difícil de extender

- Desventajas de usar un patrón de “delegación”.
- Para introducir el nuevo método hay que modificar las interfaces IGeometry y FShape (p.e: getZ[]).
- Si se mete un nuevo tipo de geometría hay que modificar todas las partes de gvSIG dónde se hace alguna operación en función del tipo.

Índice de la Presentación

Las Normas ISO

- ISO 19103: Tipos básicos y unidades de medida.
- ISO 19107: Modelo de datos de geometrías y topología.
- ISO 19108: Objectos temporales.
- ISO 19111: Sistemas de referencia.
- ISO 19123: Coberturas.
- ISO 19136: GML

Las Normas ISO

ISO 19107

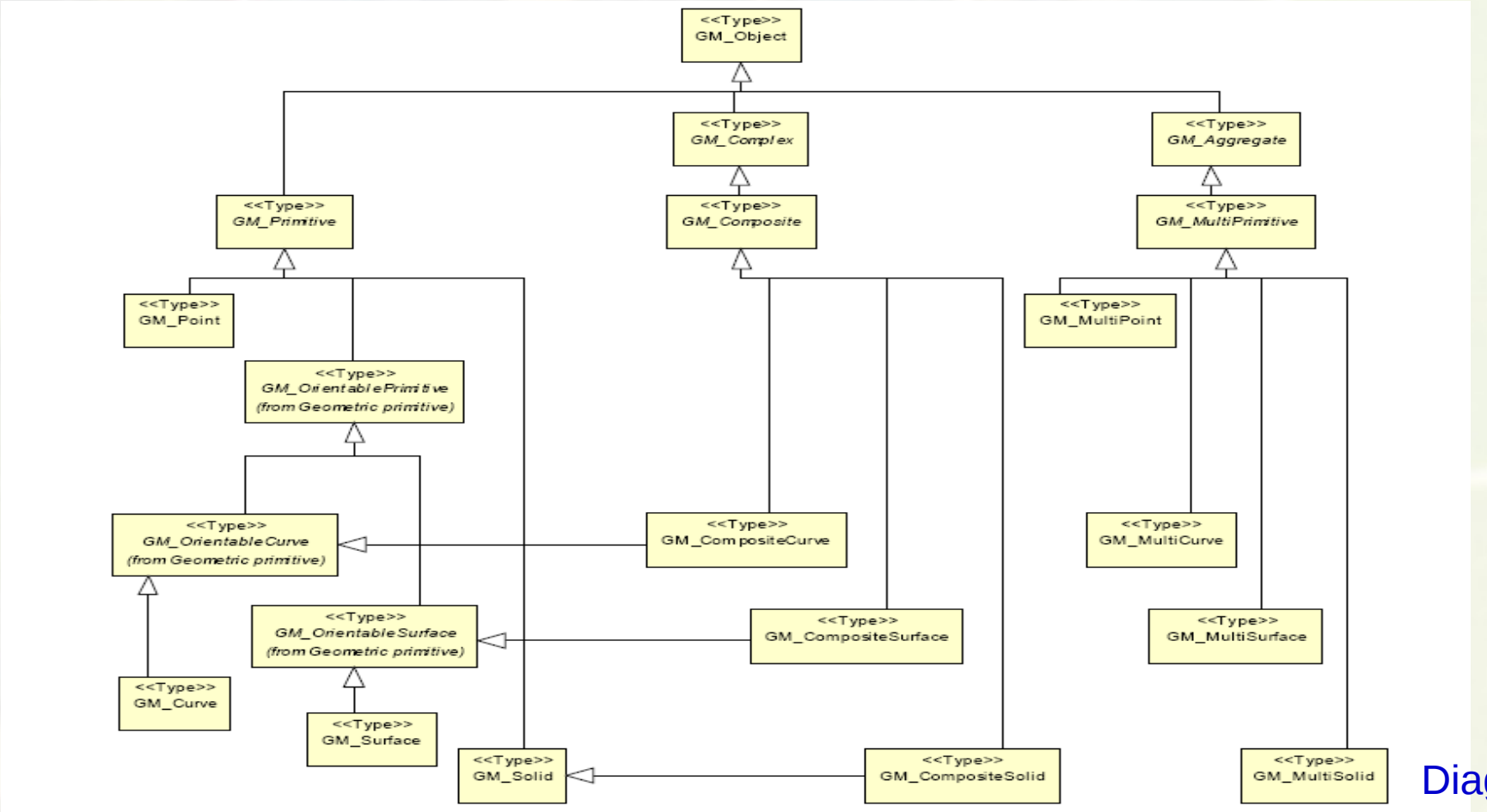


Diagrama 2

Las Normas ISO

ISO 19136: GML

- GML es un formato que soporta todas las demás ISO's.
- Extensible mediante el uso de perfiles.
  - City GML
  - EuroRoads

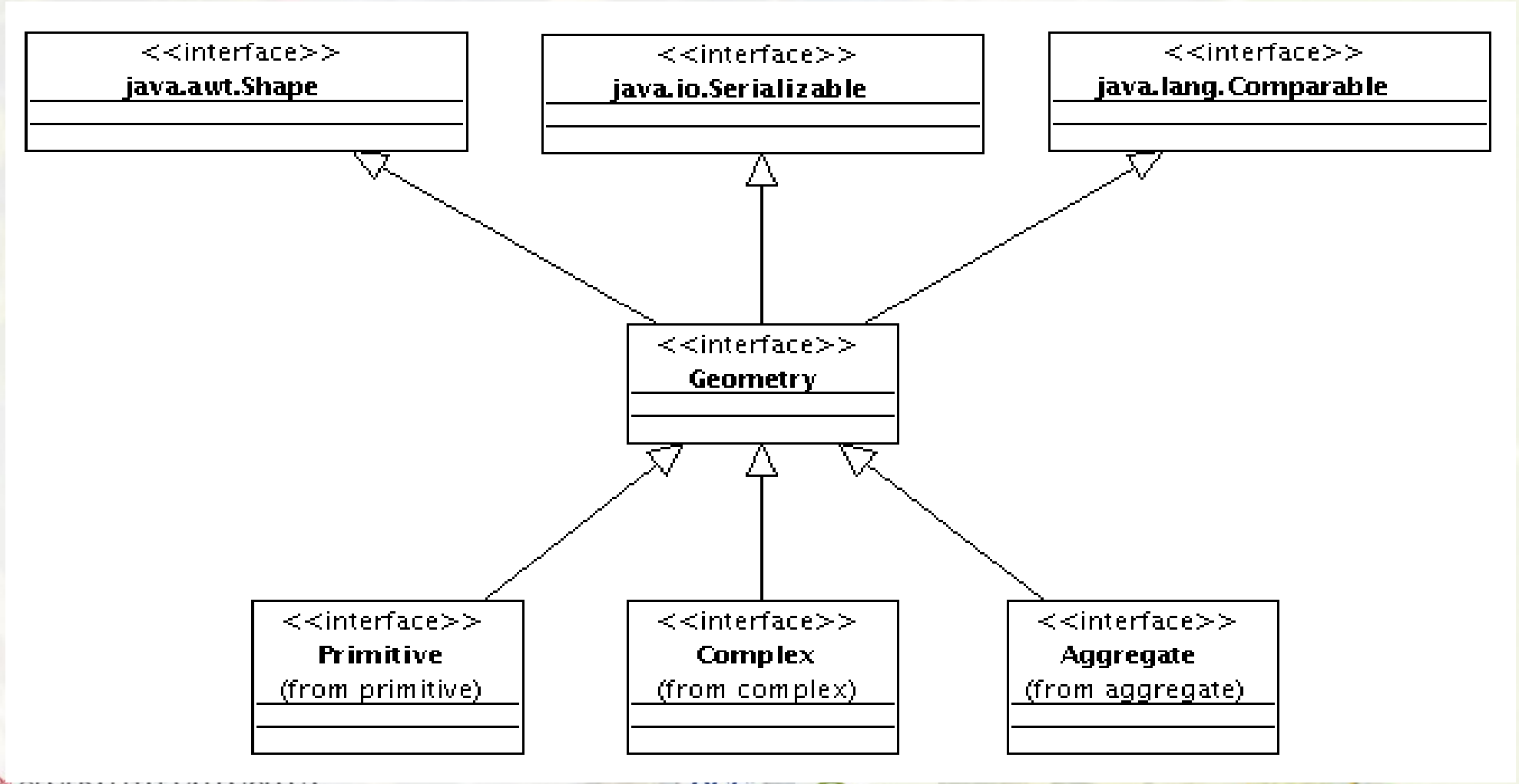
Índice de la Presentación

El Modelo Actual

- Creado a partir del viejo modelo de geometrías de gvSIG intentado que el impacto sobre el código ya desarrollado fuera el menor posible.
- Se intenta aproximar al modelo de la ISO19107.
- Separación en API - Implementación.

El Modelo Actual

La clase Geometry (GM_Object)



El Modelo Actual

Los tipos primitivos

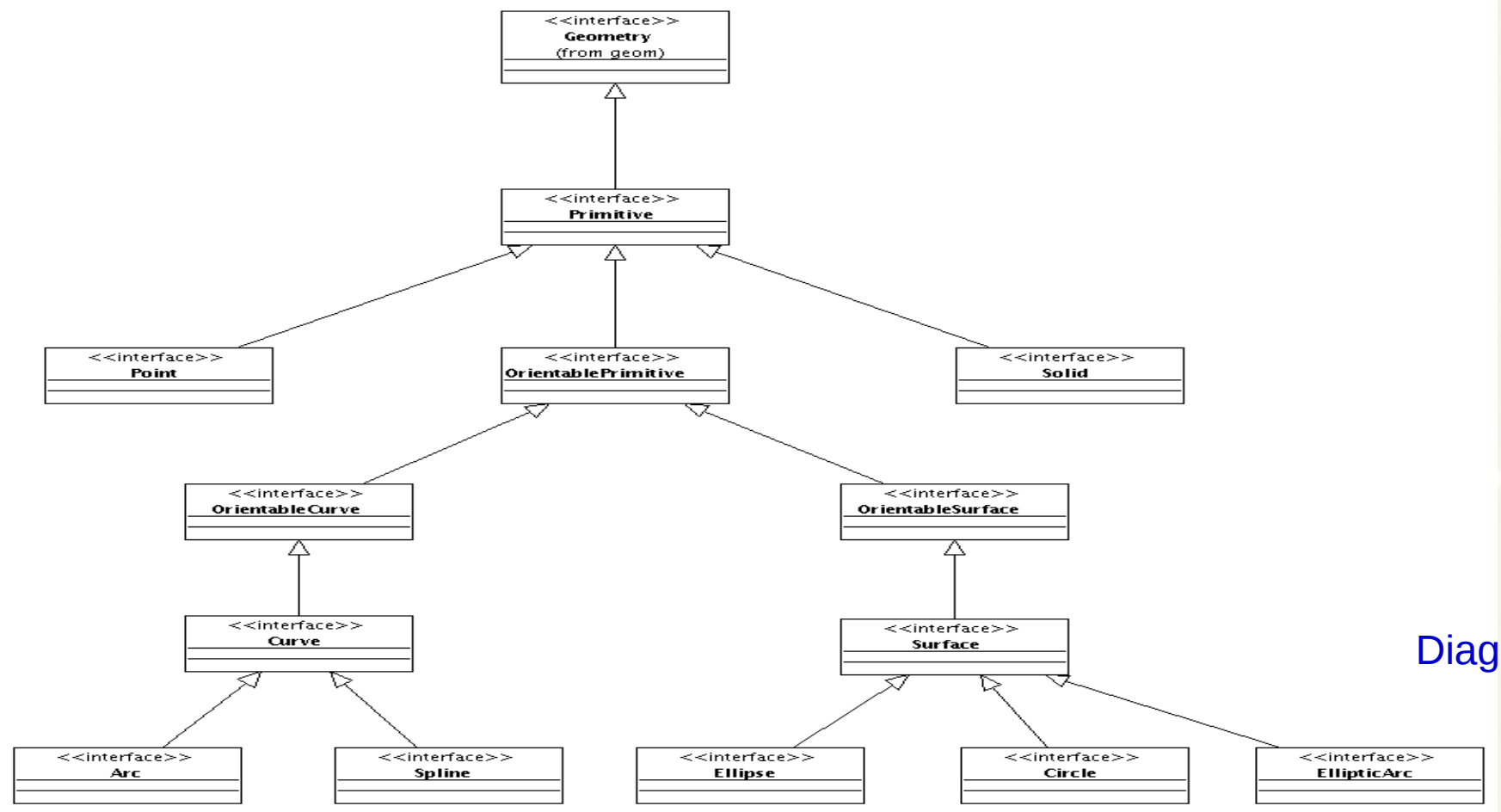


Diagrama 3

El Modelo Actual

Las geometrías complejas

- Tienen que tener un comportamiento similar al de las geometrías primitivas (tienen el mismo interfaz), pero están formadas por la unión de varias de ellas.
- Un ejemplo de este tipo de geometrías puede ser un sólido formado por varios polígonos, un polígono con polígonos de relleno, una curva formada por varias curvas...

El Modelo Actual

Las geometrías múltiples

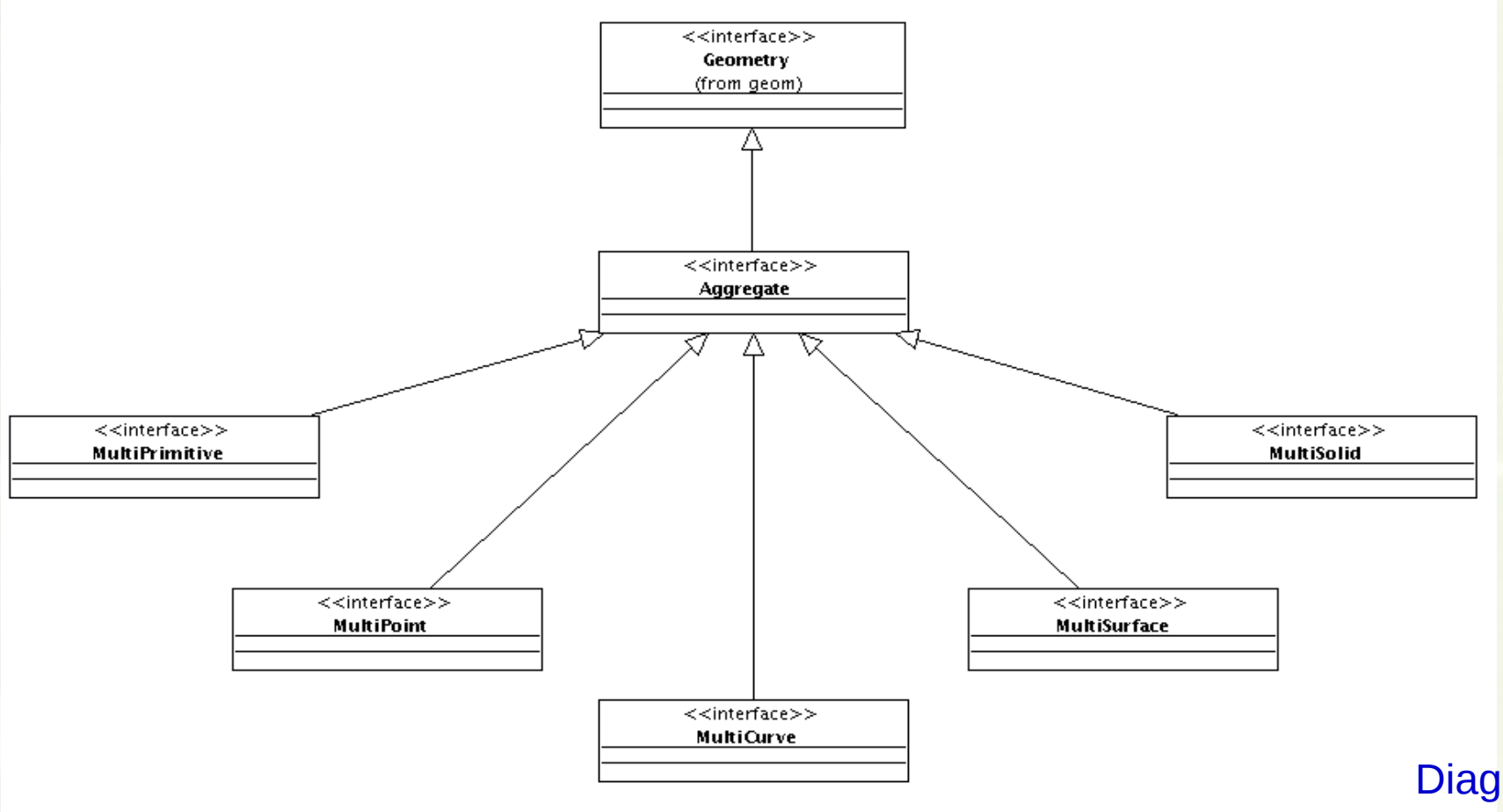


Diagrama 4

Índice de la Presentación

1. El modelo anterior.
2. Las normas ISO.
3. En modelo actual.
   - 3.1. Geometrías primitivas.
   - 3.2. Geometrías complejas.
   - 3.3. Geometrías múltiples.
4. Acceso al API de geometrías.

Acceso al API de geometrías

Clases para gestionar las geometrías:

- GeometryLocator: Se trata del locator de la librería. Nos proporciona los servicios de localización del GeometryManager a usar por la librería.
- GeometryManager: Se trata de la factoría que nos da acceso al API de geometrías. A partir de él podemos registrar las geometrías, crear nuevas, registrar operaciones... Es el punto de entrada a todo el modelo de objetos.

Acceso al API de geometríass

- Acceso al GeometryManager:

```
GeometryManager geometryManager =
  GeometryLocator.getGeometryManager();
```

- Para poder acceder al GeometryManager antes se ha tenido que registrar una implementación del mismo:

```
GeometryLocator.registerGeometryManager
  (MyGeometryManager.class);
```

Índice de la Presentación

5.Tipos de geometrías.

5.1. Registro de un tipo de geometría.

6.Las geometrías.

6.1. Creación de geometrías.

6.2. Geometrías primitivas.

6.3. Geometrías múltiples.

7.Operaciones.

7.1. Registro de operaciones.

7.2. Ejecución de una operación.

Tipos de Geometrías

Tipo de geometría:

- Según su definición geométrica, una geometría tiene que pertenecer a un tipo que se definirá mediante una constante.
- Los tipos por defecto de gvSIG se encuentran en: Geometry.TYPES
- Se pueden añadir nuevos tipos de geometrías.

Tipos de Geometrías

SubTipo de geometría:

- Según las dimensiones de la geometría, una geometría tiene que tener un subtipo.
- Los subtipos por defecto de gvSIG se encuentran en: Geometry.SUBTYPES
- Se pueden añadir nuevos subtipos de geometrías.

Tipos de Geometrías

La clase GeometryType:

- Se crea a partir de un Geometry.TYPES y de un Geometry.SUBTYPES.
- Todas las geometrías de la aplicación tienen asociado un objeto de esta clase.
- Se utilizará para mantener la asociación entre tipo de geometría y operaciones.
- Existirá una única instancia de GeometryType por tipo y por subtipo.

Tipos de Geometrías

Ejemplos de GeometryType:

TYPE	SUBTYPE	GeometryType
POINT	GEOM2D	Punto 2D
POINT	GEOM3D	Punto 3D
POINT	GEOM2DM	Punto 2DM
CURVE	GEOM2D	Curva en 2D
SURFACE	GEOM3D	Polígono en 3D

Tipos de Geometrías

Registro de Geometrías

- Para que una geometría pueda ser utilizada su tipo ha debido der registrado.
- Los tipos se registran al arrancar la aplicación en la clase donde se tienen que registrar los objetos relacionados con las geometrías: GeometryLibrary .

Tipos de Geometrías

```
geometryManager.registerGeometryType
  (Arc2D.class, "Arc2D",
  TYPES.ARC, SUBTYPES.GEOM2D);

geometryManager.registerGeometryType
  (Circle2D.class, "Circle2D",
  TYPES.CIRCLE, SUBTYPES.GEOM2D);

geometryManager.registerGeometryType(
  Curve2DZ.class, "Curve2DZ", TYPES.CURVE,
  SUBTYPES.GEOM2DZ);
```

Índice de la Presentación

Las geometrías

Creación de geometrías

- Existe un método create en el GeometryManager que se utiliza para crear cualquier tipo de geometría a partir del tipo y del subtipo.
- El GeometryManager devolverá una geometría vacía con el GeometryType correspondiente.
- Las geometrías implementan el interfaz Geometry.

Las geometrías

Creación de geometrías

```
Point point =
  (Point)geometryManager.create(
  TYPES.POINT, SUBTYPES.GEOM2D);

Curve curve =
  (Curve)geometryManager.create(
  TYPES.CURVE, SUBTYPES.GEOM2D);
```

Las geometrías

Creación de geometrías

- Las geometrías creadas están vacías. Una vez creadas, hay que añadir los valores de las coordenadas:

```
Point point =
  (Point)geometryManager.create(
  TYPES.POINT, SUBTYPES.GEOM2D);

point.setCoordinateAt(0,1);

point.setCoordinateAt(1,2);
```

Las geometrías

Creación de geometrías

- Para evitar tener que hacer un casting cada vez que se crea una geometría, se han añadido algunos métodos en el GeometryManager para crear las geometrías más comunes.
- Estos métodos suelen permitir establecer los valores de “las dos primeras dimensiones” de la geometría.

Las geometrías

Creación de geometrías

```
Point point1 =
  (Point)geometryManager.create(
  TYPES.POINT, SUBTYPES.GEOM2D);

point1.setCoordinateAt(0,1);

point1.setCoordinateAt(1,2);

Point point2 =
  geometryManager.createPoint(1, 2,
  SUBTYPES.GEOM2D);
```

Las geometrías

Creación de geometrías

- Se puede utilizar un método directo para crear una geometría con más de dos dimensiones:

```
Point point =
  geometryManager.createPoint(1, 2,
  SUBTYPES.GEOM3D);

point.setCoordinateAt(2,3);
```

Las geometrías

Geometrías primitivas: Point

- Tiene métodos para obtener y/o establecer el valor de cualquier dimensión del punto.
- Existen métodos específicos para la primera y la segunda dimensión.

Las geometrías

Geometrías primitivas: Curve

- Conserva la dependencia con el GeneralPathX

```
Curve curve =
  (Curve)geometryManager.create(
  TYPES.CURVE, SUBTYPES.GEOM2D);

GeneralPathX generalPathX = new
  GeneralPathX();

generalPathX.moveTo(5, 5);

generalPathX.lineTo(10, 10);

curve.setGeneralPath(generalPathX);
```

Las geometrías

Geometrías primitivas: Curve

- A excepción del punto, todas las demás geometrías se tienen que poder construir utilizando objetos del propio modelo.
- De ese modo, podemos construir una Curve a partir de un conjunto de objetos de tipo Point.

Las geometrías

Geometrías primitivas: Curve

- Asumimos que se han creado previamente dos objetos point1 y point2 de tipo Point.

```
Curve curve =
  (Curve)geometryManager.create(
  TYPES.CURVE, SUBTYPES.GEOM2D);

curve.insertVertex(0, point1);

curve.insertVertex(1, point2);
```

Las geometrías

Geometrías primitivas: Curve

- Se pueden editar cualquiera de los puntos que forman la curva.
- En el ejemplo se elimina un nodo intermedio de una Curve y luego se añade uno nodo de tipo Point.

```
curve.removeVertex(4);
curve.insertVertex(4, point);
```

Las geometrías

Geometrías primitivas: Arc

- Se puede crear un arco a partir del centro y de los puntos de inicio y final.
- Asumimos que se han creado previamente dos objetos de tipo Point correspondientes.

```
Arc arc =
  (Arc)geometryManager.create(
  TYPES.ARC, SUBTYPES.GEOM2D);

arc.setPoints(centerPoint, startPoint,
  endPoint);
```

Las geometrías

Geometrías primitivas: Surface

- Se puede crear una surface a partir de un GeneralPathX.
- Asumimos que se han creado previamente un GeneralPathX con las coordenadas de la Surface.

```
Surface surface =
  (Surface)geometryManager.create(
   TYPES.SURFACE, SUBTYPES.GEOM2D);

surface.setGeneralPath(generalPath);
```

Las geometrías

Geometrías primitivas: Surface

- Una surface también se puede crear a partir de un conjunto de objetos de tipo Point.
- Asumimos que se han creado previamente un conjunto de objetos de tipo Point.

Las geometrías

Geometrías primitivas: Surface

```
Surface surface =
  (Surface)geometryManager.create(
   TYPES.SURFACE, SUBTYPES.GEOM2D);

surface.inserVertex(0, point1);

surface.inserVertex(1, point2);

surface.inserVertex(2, point3);

surface.inserVertex(3, point4);
```

Las geometrías

Geometrías primitivas: Surface

- Se pueden editar cualquiera de los puntos que forman la surface.
- En el ejemplo se elimina un nodo intermedio de una Surface y luego se añade uno nodo de tipo Point.

```
surface.removeVertex(3);
surface.insertVertex(3, point);
```

Las geometrías

Geometrías primitivas: Circle

- Se puede crear un círculo a partir del centro y del radio.
- Asumimos que se han creado previamente los objetos de tipo Point correspondientes.

```
Circle circle =
  (Circle)geometryManager.create(
  TYPES.CIRCLE, SUBTYPES.GEOM2D);

circle.setPoints(
  pointCenter, pointRadious);
```

Las geometrías

El Envelope.

- Representa el bounding box o el extent de un conjunto de datos.
- Se crea mediante el manager.
- No es una geometría.

```
Envelope envelope =
  (Envelope)geometryManager.createEnvelope(
      SUBTYPES.GEOM2D);

envelope.setLowerCorner(point1);

envelope.setUpperCorner(point2);
```

Las geometrías

Geometrías múltiples: Aggregate

- Es la clase raíz de todas las geometrías múltiples.
- Contiene métodos para recuperar cada una de las geomertías que forman las geometría múltiple, pero no para editarlas.

Las geometrías

Geometrías múltiples: MultiPoint

- Define una geometría formada por una colección de puntos.

```
MultiPoint multiPoint =
  (MultiPoint)geometryManager.create(
      TYPES.MULTIPOINT, SUBTYPES.GEOM2D);

multiPoint.addPoint(point1);

multiPoint.addPoint(point2);

multiPoint.addPoint(point3);
```

Las geometrías

Geometrías múltiples: MultiCurve, MultiSurface, MultiSolid y MultiPrimitive.

- Similares al MultiPoint, pero cambiando el tipo de geometría que contienen.

Las geometrías

Geometrías complejas: Solid

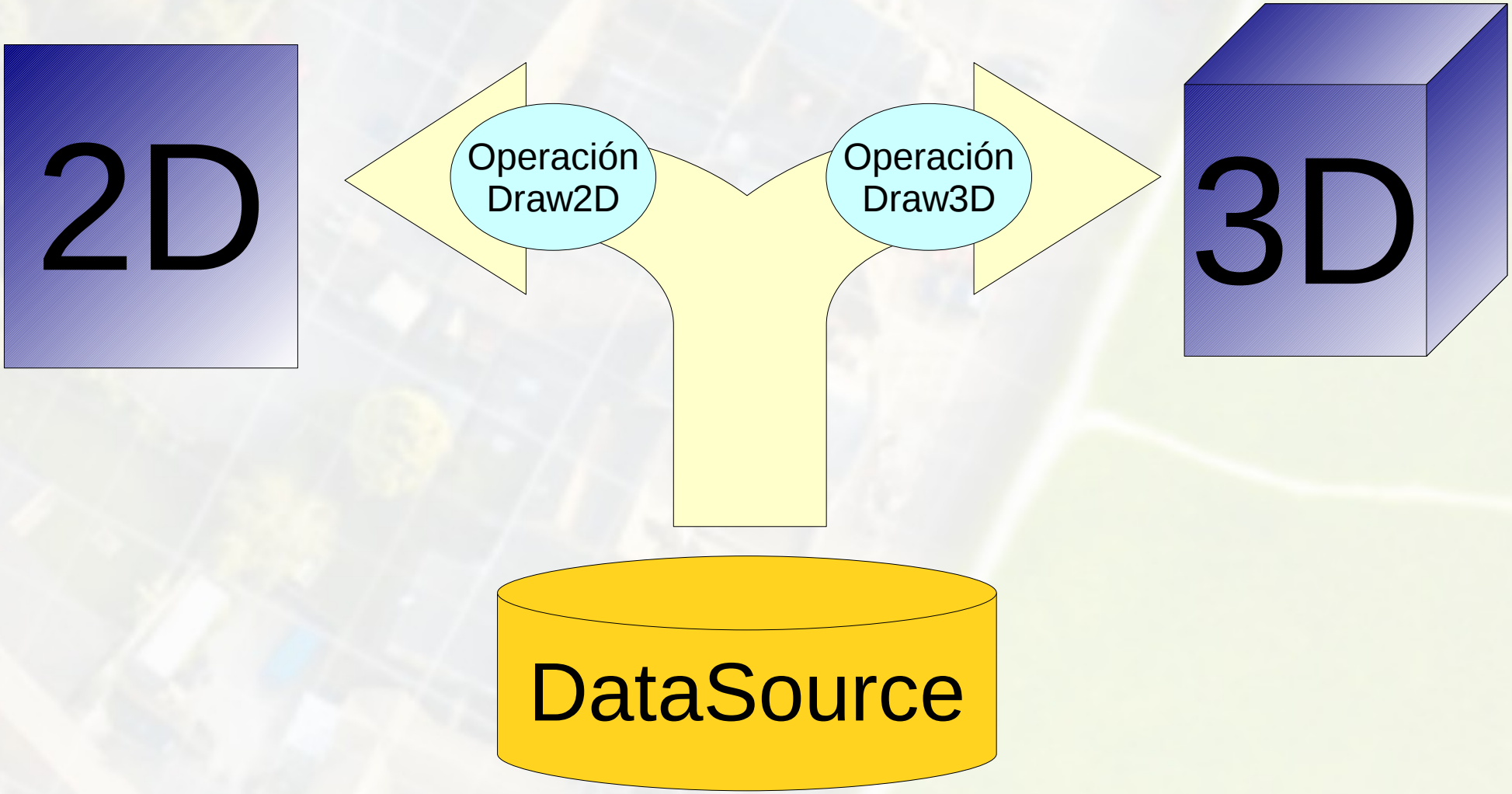
- Está compuesto por objetos de tipo Surface.
- Tiene una apariencia que define las texturas, el material...

Índice de la Presentación

Operaciones

- Uno de los objetivos del refáctoring era poder extender el modelo de geometrías dinámicamente.
- Para ello se crea el concepto de operación que actúa sobre un tipo geometría.
- Un ejemplo de operación es el dibujado en 2D y en 3D

Operaciones



Operaciones

Registro de operaciones

- Para que una operación pueda ser utilizada antes debe ser registrada en el GeometryManager.
- En el ejemplo se asume que existe una operación de dibujado de puntos en 2 dimensiones `Draw2DPointOperation`.

```
geometryManager.registerOperation(
  "Draw2D", Draw2DPointOperation,
  TYPES.POINT, SUBTYPES.GEOM2D);
```

Operaciones

Registro de operaciones

- Al registrar una operación el GeometryManager devuelve un entero con el código de la operación, de modo que todas las operaciones registradas con el mismo nombre presentarán el mismo código.

Operaciones

Registro de operaciones

- En el ejemplo code1 y code2 tienen el mismo valor.

```
int code1 =
  geometryManager.registerOperation(
   "Draw2D", Draw2DPointOperation,
  TYPES.POINT, SUBTYPES.GEOM2D);

int code2 =
  geometryManager.registerOperation(
   "Draw2D", Draw2DCurveOperation,
  TYPES.CURVE, SUBTYPES.GEOM2D);
```

Operaciones

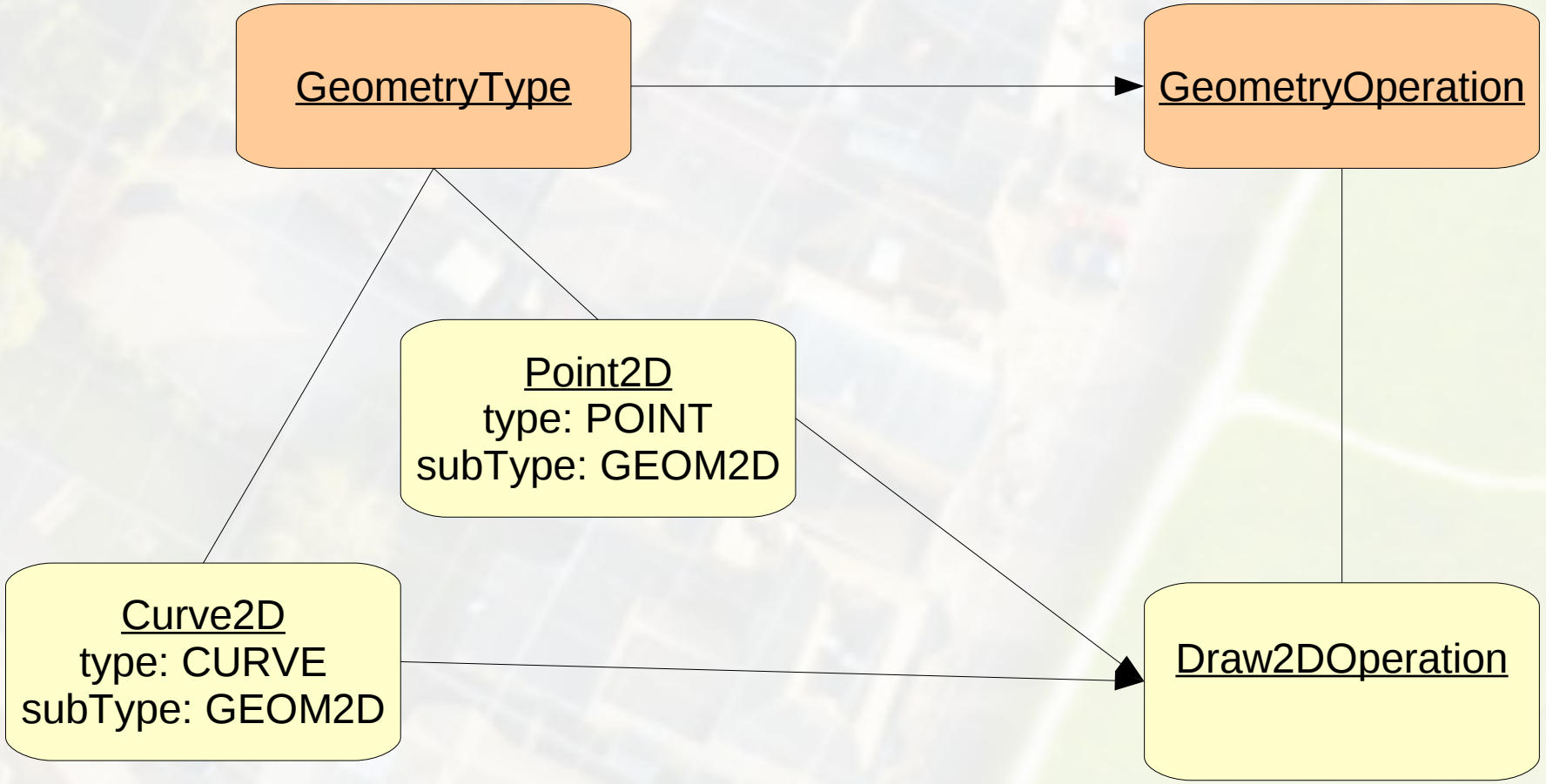
Registro de operaciones

- Se resgistran Objetos, no clases.
- Las operaciones se asocian a un GeometryType, de modo que cada tipo de geometría "conoce" todas las operaciones que se han registrado sobre ella.

```
GeometryType geometryType =
  point.getGeometryType();
GeometryOperation geometryOperation =
  geometryType.getGeometryOperation(code);
```

Operaciones

Registro de operaciones



Operaciones

Registro de operaciones

- Se puede asociar una operación a todos los tipos de geometría.
- En el ejemplo se asume que existe una operación de dibujado `Draw2DOperation`.

```
geometryManager.registerOperation(
  "Draw2D", Draw2DOperation);
```

Operaciones

Registro de operaciones

- Se puede asociar una operación a un tipo de geometría (sin importar la dimensión).
- En el ejemplo se asume que existe una operación de dibujado de puntos `DrawPoints` independiente de la dimensión.

```
geometryManager.registerOperation(
  "Draw2D", DrawPoints, TYPES.POINT);
```

Operaciones

Registro de operaciones

- Se puede asociar una operación a un subTipo de geometría (sin importar el tipo).
- En el ejemplo se asume que existe una operación de dibujado de geometrías en 2D `Draw2D`.

```
geometryManager.registerOperationBySubType(
  "Draw2D", Draw2D, SUBTYPES.GEOM2D);
```

Operaciones

Ejecución de operaciones

- Para ejecutar una operación se define el concepto de “contexto de operación”.
- El contexto incluye todos los parámetros que una operación necesita para ejecutarse.
- Es inecesario cuando la propia geometría contiene todo lo necesario para ejecutar la operación.
- La clase GeometryOperationContext se usa con este propósito.

Operaciones

Ejecución de operaciones

- El GeometryOperationContext no es más que un map con los parámetros que se necesitan para ejecutar la operación.
- Se pueden definir contextos concretos para operaciones concretas:

Operaciones

```
public class
  IntersectsGeometryOperationContext
       extends GeometryOperationContext {

 public
  IntersectsGeometryOperationContext(
  Geometry geom){
    setAttribute("geom2", geom);
  }

 public Geometry getGeom(){
   return (Geometry)getAttribute("geom2");
  }

}
```

Operaciones

Ejecución de operaciones

- Las operaciones se pueden ejecutar sobre el GeometryManager.
- En el ejemplo se asume que hay una geometría de tipo Point y un contexto adecuado para ejecutar la operación “Draw2D”:

```
geometryManager.invokeOperation(“Draw2D”,
  point, contextDraw2D);
```

Operaciones

Ejecución de operaciones

- El método devuelve un Object en caso de que tenga sentido en el contexto de la operación.
- Hay operaciones que no devuelven ningún resultado, como por ejemplo de operación de dibujado.

Operaciones

Ejecución de operaciones

- También se pueden ejecutar operaciones sobre el GeometryManager conociendo el código de operación.

```
geometryManager.invokeOperation(code,
  point, contextDraw2D);
```

Operaciones

Ejecución de operaciones

- Conociendo la geometría, se pueden invocar operaciones por nombre o por código:

```
point.invokeOperation("Draw2D",
  contextDraw2D);

point.invokeOperation(code,
  contextDraw2D);
```

Operaciones

Ejecución de operaciones

- Hay casos en los que hay que ejecutar una misma operación muchas veces. Para acelerar el proceso es conveniente ejecutar la operación sobre un GeometryOperation.

```
GeometryOperation operation =
  geometryManager.getGeometryOperation(
  code, TYPES.Point,SUBTYPES.GEOM2D);

operation.invoke(point, context);
```

gvSIG. Geographic Information System of the Valencian Government