



EIEL-Volcado

ESPECIFICACIÓN DEL SUBSISTEMA

Proyecto	EIEL-Volcado
Responsable	David Trillo Pérez
Versión	0.1
Fecha	17/04/2008
Estado	Final
Clasificación	Público

Índice

1. Introducción.....	3
2. Arquitectura global del sistema	10
3. Descripción de datos Calculados automáticamente para cada tabla.....	11
4. Descripción de datos del MAP Calculados automáticamente	14
Historial de Cambios	20

1. Introducción

Nombre del proyecto: [EIEL-Volcado](#)

Descripción:

EIEL-Volcado proporciona métodos para el cálculo automático de datos que se derivan de otros campos almacenados en la BDT-EIEL. El objetivo es calcular de forma automática campos que de otra forma serían costosos de obtener.

Operaciones principales:

- Para cada tipo de entidad contenida en un municipio concreto:
 - Calcular datos derivados: Obtiene el valor de aquellos campos que se pueden calcular a partir de otros campos de la misma entidad o de otras entidades utilizando operaciones alfanuméricas y/o espaciales.
 - Atributo orden.
 - Recogida basura: producción y calidad.
 - Municipios encuestados: déficit red saneamiento y abastecimiento, y número de puntos de luz en diseminado.
 - Núcleos encuestados: déficit red de saneamiento y abastecimiento, déficit de alumbrado en viviendas y calles, número de accesos por carretera.
 - Longitud de tramos.
 - Distancia desde los puntos de vertido a los núcleos más cercanos.
 - Longitud de los emisarios.
 - Superficie para cada tipo de planeamiento.
 - Puntos de luz y potencia para cada núcleo.
 - Número de viviendas con el acceso principal sin pavimentar.
 - Simplificación: Simplifica las geometrías de las entidades a partir de la geometría de más alta escala.
- Para cada carretera:
 - Calcular datos derivados:
 - Calcular los puntos kilométricos iniciales y finales, y longitud total de la carretera (si no está en proyecto) .
 - Simplificación.

Estructura de directorios:

- bin: Contiene los .class del subsistema
- dist: Contiene EIEL-Volcado.jar con las clases del subsistema.
- doc: Contiene la documentación javadoc
- lib: Contiene diferentes varias librerías utilizadas por el subsistema.
- src: Contiene los fuentes del subsistema.
- src/images: Contiene las imágenes que utiliza la aplicación.
- Scripts: Contiene los scripts que utiliza la aplicación para el volcado.

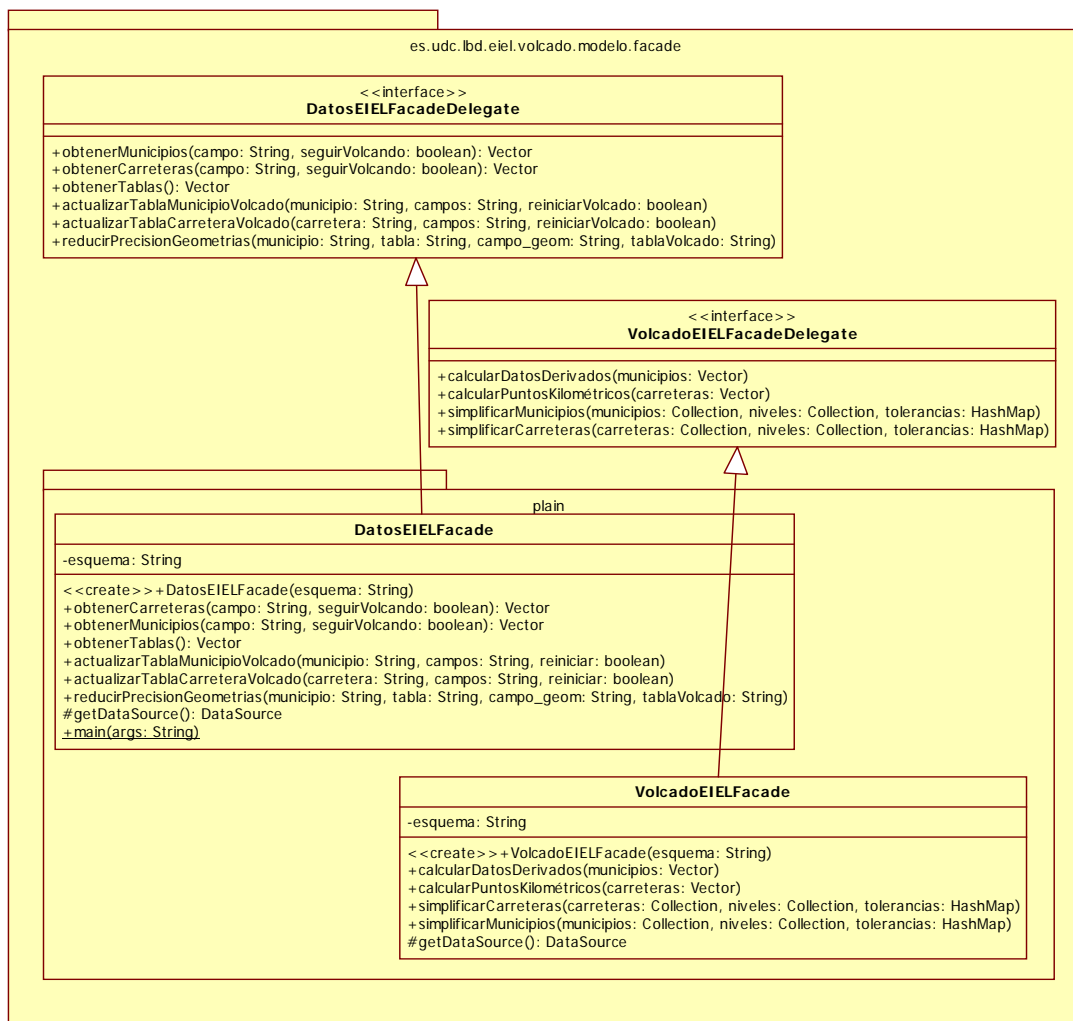
Estructura de paquetes:

- Directorio “base” del subsistema: **es.udc.lbd.eiel.volcado**.
- Clase principal del módulo:
 - Con autenticación:
 - **es.udc.lbd.eiel.volcado.modelo.facade.DatosEIELFacade**
Esta fachada reúne las operaciones que son utilizadas de forma genérica.
 - **es.udc.lbd.eiel.volcado.modelo.facade.VolcadoEIELFacade**.
Esta fachada reúne las operaciones específicas del volcado.

Dependencias con subsistemas:

- EIEL-Utilidades: **es.udc.lbd.eiel.util**
- EIEL-Autenticacion: **es.udc.lbd.eiel.autenticacion**

Fachada del Subsistema:



JavaDoc interfaz

1.1 es.udc.lbd.eiel.volcado.modelo.facade

Interface DatosEIELFacadeDelegate

All Known Implementing Classes:

[DatosEIELFacade](#)

```
public interface DatosEIELFacadeDelegate
```

Interfaz del subsistema EIEL-Volcado.

Proporciona métodos de utilidad genérica que proporcionan información para el volcado.

Author:

Verónica Fariña Iglesias

Method Summary

void	actualizarTablaCarreteraVolcado (java.lang.String carretera, java.lang.String[] campos, boolean reiniciarVolcado) Entradas: carretera:String con la carretera que se va a procesar.
void	actualizarTablaMunicipioVolcado (java.lang.String municipio, java.lang.String[] campos, boolean reiniciarVolcado) Entradas: municipio:String con el municipio que se va a procesar.
java.util.Vector	obtenerCarreteras (java.lang.String campo, boolean seguirVolcando) Entradas: seguirVolcando:boolean que indica si se sigue con el volcado anterior o se reinicia uno nuevo .
java.util.Vector	obtenerMunicipios (java.lang.String campo, boolean seguirVolcando) Entradas: seguirVolcando:boolean que indica si se sigue con el volcado anterior o se reinicia uno nuevo .
java.util.Vector	obtenerTablas () Acciones: devuelve un vector con las tablas que se pueden volcar.
void	reducirPrecisionGeometrias (java.lang.String municipio, java.lang.String tabla, java.lang.String campo_geom, java.lang.String tablaVolcado)

Method Detail

obtenerMunicipios

```
java.util.Vector obtenerMunicipios(java.lang.String campo,  
                                     boolean seguirVolcando)  
                                     throws  
es.udc.lbd.eiel.util.exceptions.InternalErrorException
```

Entradas: seguirVolcando:boolean que indica si se sigue con el volcado anterior o se reinicia uno nuevo . campo:String que indica que tipo de operación se va a realizar.

Acciones: obtiene los municipios a procesar en función del valor de la entrada.

Salidas:

- Vector de objetos MunicipioTO

Throws:

es.udc.lbd.eiel.util.exceptions.InternalErrorException

obtenerCarreteras

```
java.util.Vector obtenerCarreteras(java.lang.String campo,  
                                     boolean seguirVolcando)  
                                     throws  
es.udc.lbd.eiel.util.exceptions.InternalErrorException
```

Entradas: seguirVolcando:boolean que indica si se sigue con el volcado anterior o se reinicia uno nuevo . campo:String que indica que tipo de operación se va a realizar.

Acciones: obtiene las carreteras a procesar en función del valor de la entrada.

Salidas:

- Vector de objetos CarreteraTO

Throws:

es.udc.lbd.eiel.util.exceptions.InternalErrorException

obtenerTablas

```
java.util.Vector obtenerTablas()  
                                     throws  
es.udc.lbd.eiel.util.exceptions.InternalErrorException
```

Acciones: devuelve un vector con las tablas que se pueden volcar.

Salidas:

- Vector de String con los nombres de las tablas.

Throws:

es.udc.lbd.eiel.util.exceptions.InternalErrorException

actualizarTablaMunicipioVolcado

```
void actualizarTablaMunicipioVolcado(java.lang.String municipio,  
                                     java.lang.String[] campos,  
                                     boolean reiniciarVolcado)  
    throws  
es.udc.lbd.eiel.util.exceptions.InternalErrorException
```

Entradas: municipio:String con el municipio que se va a procesar.
campos:String[] campos que indican qué operación se va a realizar.
reiniciarVolcado:boolean que indica si seguirá o no con el volcado.

Acciones: pone a false o true los municipios que se han procesado.

Throws:

es.udc.lbd.eiel.util.exceptions.InternalErrorException

actualizarTablaCarreteraVolcado

```
void actualizarTablaCarreteraVolcado(java.lang.String carretera,  
                                     java.lang.String[] campos,  
                                     boolean reiniciarVolcado)  
    throws  
es.udc.lbd.eiel.util.exceptions.InternalErrorException
```

Entradas: carretera:String con la carretera que se va a procesar. campos:String[]
con los campos que indican qué operación se va a realizar.
reiniciarVolcado:boolean que indica si seguirá o no con el volcado.

Acciones: pone a false o true las carreteras que se han procesado.

Throws:

es.udc.lbd.eiel.util.exceptions.InternalErrorException

1.2 es.udc.lbd.eiel.volcado.modelo.facade**Interface VolcadoEIELFacadeDelegate****All Known Implementing Classes:**

[VolcadoEIELFacade](#)

```
public interface VolcadoEIELFacadeDelegate
```

Interfaz del subsistema EIEL-Volcado.

Define las operaciones que se llevan a cabo en el volcado.

1. Calcular datos derivados:

-Calcular atributo orden.

-Recogida basura.

-Déficit de red la red de saneamiento y abastecimiento, y número de puntos de luz en

diseminado.

- Calcular longitud de los distintos tramos (conduccion, distribucion, etc).
- Calcular déficit de alumbrado, déficit de red de saneamiento y abastecimiento, número de accesos por carretera para cada núcleo.
- Calcular distancia desde los puntos de vertidos a los núcleos.
- Calcular longitud de los emisarios.
- Calcular superficie para cada tipo de planeamiento.
- Calcular puntos de luz y potencia.
- Calcular viviendas con acceso principal sin pavimentar.

2. Calcular puntos kilométricos:

- Calcular puntos kilométricos iniciales y finales. – Calcular longitud de cada tramo de carretera y de toda la carretera.

3. Simplificación a nivel de municipios.

4. Simplificación a nivel de carreteras.

Author:

Verónica Fariña Iglesias

Method Summary

void	<u>calcularDatosDerivados</u> (java.util.Vector municipios) Entradas: municipios:Vector con los municipios a procesar.
void	<u>calcularPuntosKilométricos</u> (java.util.Vector carreteras) Entradas: carreteras:Vector con las carreteras a procesar.
void	<u>simplificarCarreteras</u> (java.util.Collection carreteras, java.util.Collection niveles, java.util.HashMap tolerancias) Entradas: carreteras:Vector con las carreteras a procesar.
void	<u>simplificarMunicipios</u> (java.util.Collection municipios, java.util.Collection niveles, java.util.HashMap tolerancias) Entradas: municipios:Vector con las carreteras a procesar.

Method Detail

calcularDatosDerivados

```
void calcularDatosDerivados(java.util.Vector municipios)
                               throws
es.udc.lbd.eiel.util.exceptions.InternalErrorException
```


Entradas: municipios: Vector con los municipios a procesar.

Acciones: para cada municipio, calcula los datos que se derivan del mismo.

Throws:

es.udc.lbd.eiel.util.exceptions.InternalErrorException

calcularPuntosKilométricos

```
void calcularPuntosKilométricos(java.util.Vector carreteras)
                                throws
es.udc.lbd.eiel.util.exceptions.InternalErrorException
```

Entradas: carreteras: Vector con las carreteras a procesar.

Acciones: para cada carretera, calcula la longitud de las mismas y los puntos kilométricos.

Throws:

es.udc.lbd.eiel.util.exceptions.InternalErrorException

simplificarMunicipios

```
void simplificarMunicipios(java.util.Collection municipios,
                           java.util.Collection niveles,
                           java.util.HashMap tolerancias)
                           throws
es.udc.lbd.eiel.util.exceptions.InternalErrorException
```

Entradas: municipios: Vector con las carreteras a procesar.

niveles: Vector niveles de geometría que se van a simplificar.

tolerancias: HashMap con las tolerancias de simplificación para cada nivel.

Acciones: simplificación de las geometrías a nivel de municipio.

Throws:

es.udc.lbd.eiel.util.exceptions.InternalErrorException

simplificarCarreteras

```
void simplificarCarreteras(java.util.Collection carreteras,
                           java.util.Collection niveles,
                           java.util.HashMap tolerancias)
                           throws
es.udc.lbd.eiel.util.exceptions.InternalErrorException
```

Entradas: carreteras: Vector con las carreteras a procesar.

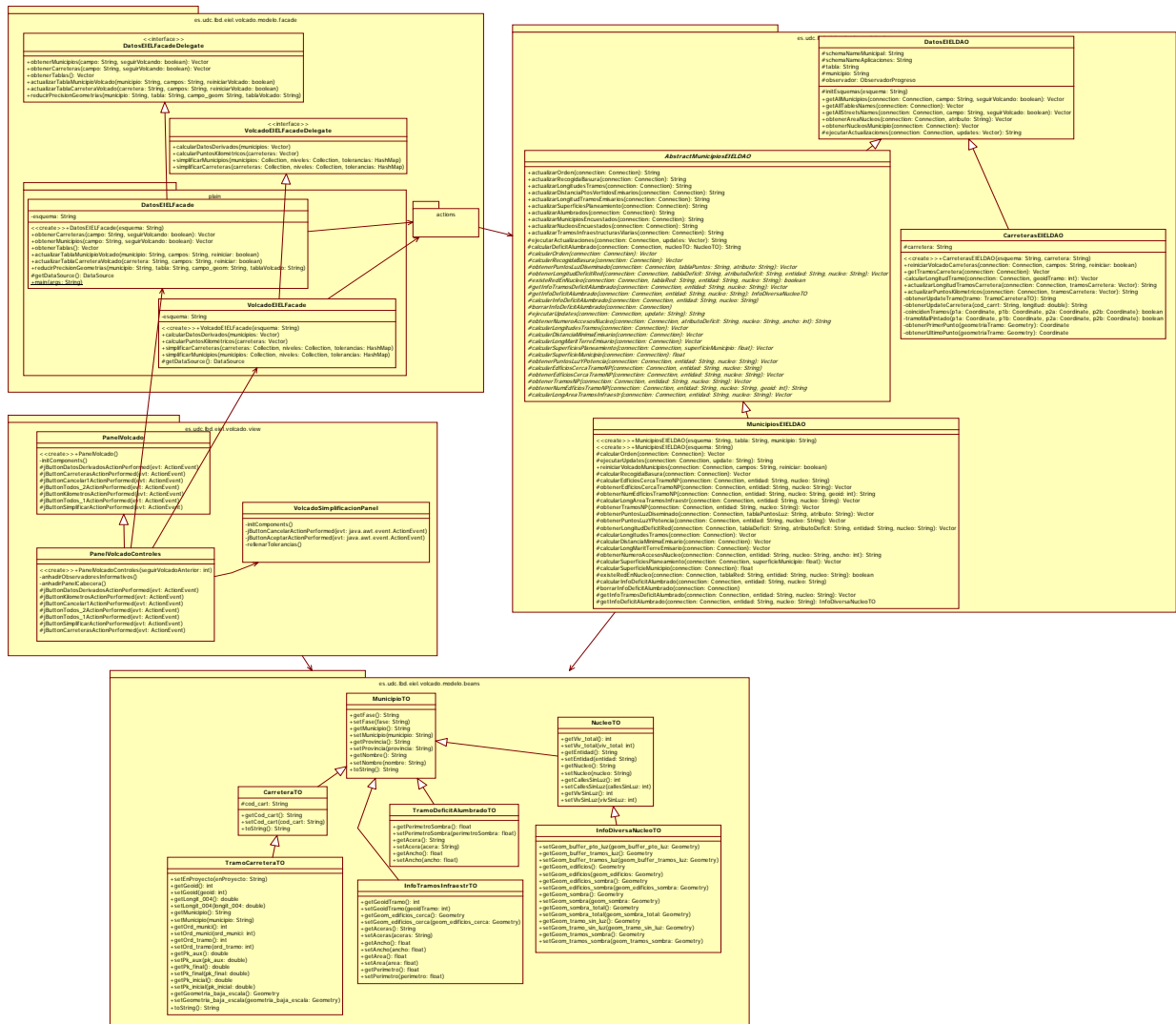
* niveles: Vector niveles de geometría que se van a simplificar.

tolerancias: HashMap con las tolerancias de simplificación para cada nivel.

Acciones: simplificación de las geometrías a nivel de carretera.

```
es.udc.lbd.eiel.util.exceptions.InternalErrorException
```

2. Arquitectura global del sistema



3. Descripción de datos Calculados automáticamente para cada tabla

A continuación se presentará un resumen de los datos calculados por el subsistema:

Cálculo de datos derivados

a. Cálculo del campo orden de las tablas:

Tablas:

- captaciones_agua
- colectores
- conducciones_agua
- depositos_de_agua
- depuradoras
- emisarios
- tra_potabilizacion
- casa_consistorial
- centro_asistencial
- cementerios
- centros_ensenanza
- centros_sanitarios
- centros_culturales
- edific_pub_sin_uso
- instal_deportivas
- lonja_merc_feria
- mataderos
- parques
- proteccion_civil
- tanatorios
- vertederos

b. Cálculo de los campos de la tabla *recogida_basura*:

- **produ_basu**
- **calidad**

Estos campos se calculan con una función que se encuentra en la siguiente archivo del subsistema: Scripts/volcado_recogida_basuras.sql.

c. Cálculo de los campos de *mun_encuestados*:

- **longi_011**: longitud de déficit de la red de distribución en diseminado.
- **ramal_014**: longitud de déficit de la red de saneamiento en diseminado.
- **puntos_luz**: número de puntos de luz en diseminado.

d. Cálculo de los campos de *nucl_encuestados*:

Para cada núcleo se calcula:

- **superficie**: superficie del núcleo.
- **syd_l_defi**: longitud de red con déficit de alcantarillado.
- **aag_l_defi**: longitud de la red deficitaria de distribución de agua.
- **alu_v_sin**: número de viviendas sin luz.
- **alu_l_sin**: longitud de las calles sin luz.
- **numero_accesos**: número de carreteras por las que se puede acceder al núcleo.
- **numero_accesos_mayores5**: número de carreteras con una ancho mayor que 5 por las que se puede acceder al núcleo.

e. Cálculo de la longitud de los tramos para las tablas:

- tramos_conduccion
- tra_red_distribucion
- tramos_colector
- tra_red_sanea

f. Cálculo de la distancia desde los puntos de vertido de los emisarios al núcleo. Tabla *emisarios_enc*.

- **distancia**: distancia desde el punto de vertido del emisario al núcleo.

g. Cálculo de la longitud de los tramos de los emisarios. Se calculan los campos de la tabla *tramos_emisarios*:

- **longit_marit**: longitud del emisario bajo el agua.
- **longit_terre**: longitud del emisario en tierra firme.

h. De la tabla *planeamiento* se calcula la superficie para cada tipo de planeamiento.

La superficie del municipio se obtiene de la tabla *mun_encuestados*, los demás de *isla_planeamiento*. Los tipos de planeamiento existentes son:

- superficie
- no_urbable
- habitats
- natura2000
- rustico_urb
- urbano_res numeric
- urbano_ind numeric
- urbaniz_res numeric
- urbaniz_ind numeric
- urbano_rus numeric
- espec_protec numeric

- urbano_no_cons
- urbaniz_delim
- rustico_prot_ord
- rustico_prot_agro
- rustico_prot_inf
- rustico_prot_aguas
- rustico_prot_costas
- rustico_prot_esp_nat
- rustico_prot_paisaj
- rustico_prot_patri
- rustico_prot_flores

i. Cálculo del alumbrado. Se actualizan los campos de la tabla *alumbrados*:

- **puntos_luz**: número de puntos de luz del núcleo.
- **pot_instal**: potencia de los puntos de luz.

j. Cálculo de los campos de la tabla *tramos_infraestr_viarías*:

- **longit_006**: longitud del tramo.
- **superf_006**: superficie del tramo.
- **viv_afecta**: número de viviendas que tienen el acceso principal sin pavimentar.

Cálculo de los puntos kilométricos.

a. Cálculo de los campos de la tabla *tramos_carreteras*:

- **longit_004**: longitud del tramo de carreteras.
- **pk_inicial**: punto kilométrico inicial.
- **pk_final**: punto kilométrico final.

b. Cálculo del campo *longitud* de cada carretera en la tabla *carreteras*.

Simplificación.

Tanto a nivel de tablas de municipios como de carreteras, se realizan una simplificación de las geometrías de las entidades.

4. Descripción de datos del MAP Calculados automáticamente

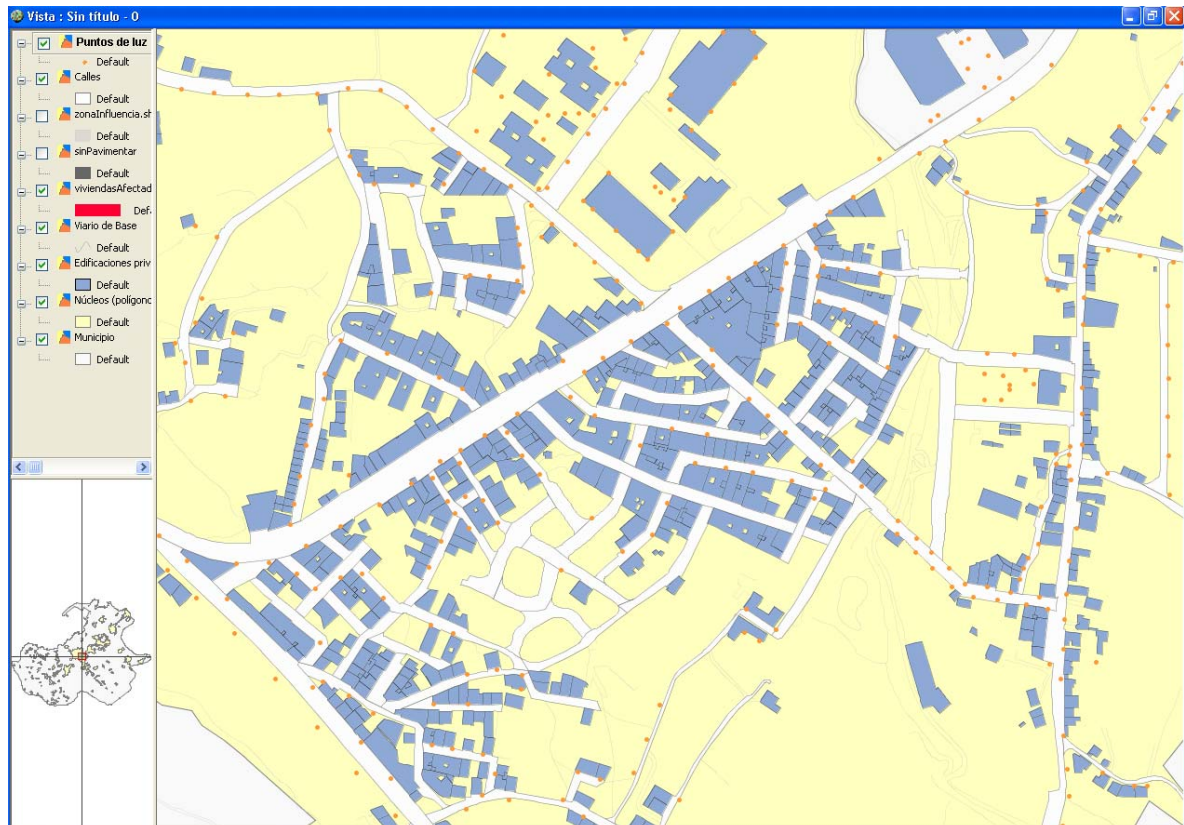
A continuación se presenta un resumen de los datos que se calculan automáticamente para el MAP.

1. ALUMBRADO

Se calculan:

- **puntos_luz** : a partir de la capa puntos_luz y nucl_encuestados se calculan el número de puntos de luz de cada núcleo.
- **pot_instal**: a partir del campo anterior puntos_luz y una constante para la potencia, se obtiene la potencia total instalada para el servicio de alumbrado público.

Ejemplo gráfico de los puntos de luz en Arteixo:



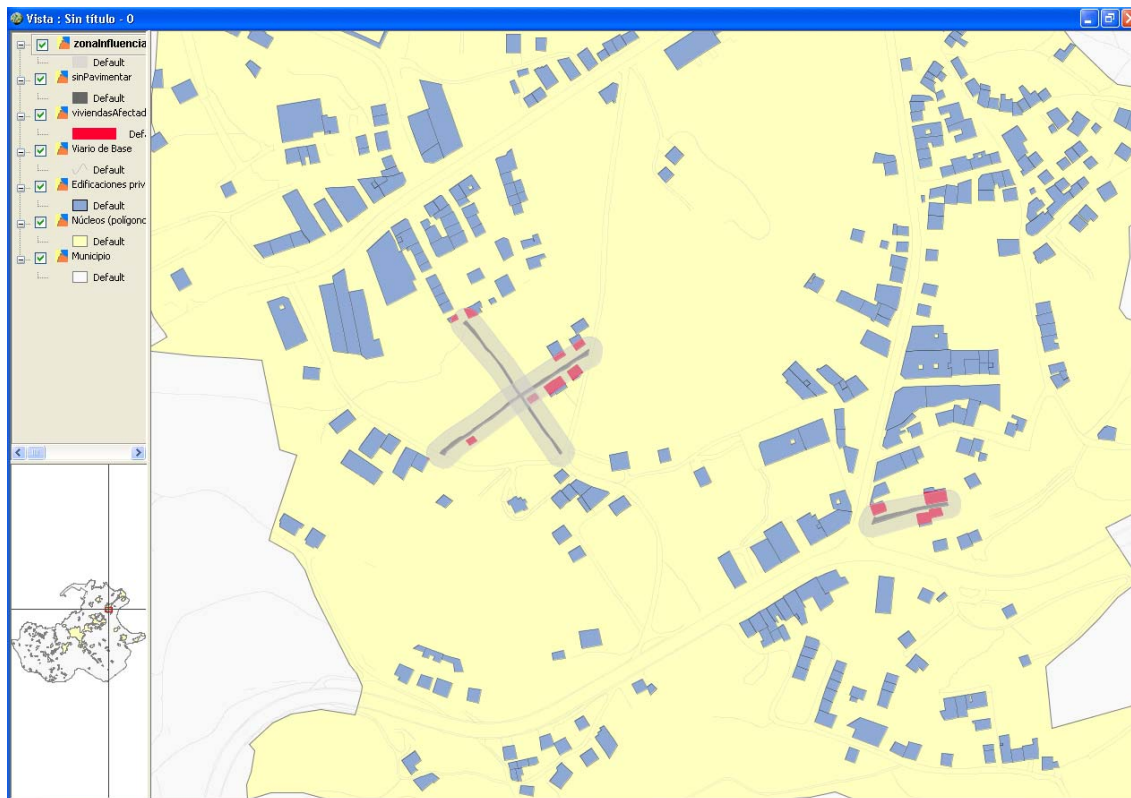
2. INFRAESTR_VIARIA

Se calcula:

- **longitud**: longitud de la infraestructura viaria.
- **superficie**: superficie de la infraestructura viaria.

- **viv_afecta:** se calcula el número de viviendas que tienen el acceso principal sin pavimentar a partir de los tramos con estado “no pavimentado”.

En este ejemplo gráfico podemos ver las viviendas afectadas (en rojo) por tener el acceso principal sin pavimentar (en gris):



3. MUN_ENC_DIS

Se calculan:

- **longitud:** a partir de la capa *tra_deficit_red_distr* se calcula la longitud de déficit de la red de distribución de agua.
- **longit_ramal:** a partir de la capa *tra_deficit_red_sanea* se calcula la longitud de déficit de la red de saneamiento.
- **puntos_luz:** a partir de la capa *puntos_luz* se calculan el número de puntos de luz en diseminado.

4. NUCL_ENCUESTADOS_3

Se calculan:

- **aag_1_defi :** la longitud de la red deficitaria de distribución de agua se calcula para cada núcleo a partir de la capa *tra_deficit_red_distr*.

5. NUCL_ENCUESTADOS_5

Se calculan:

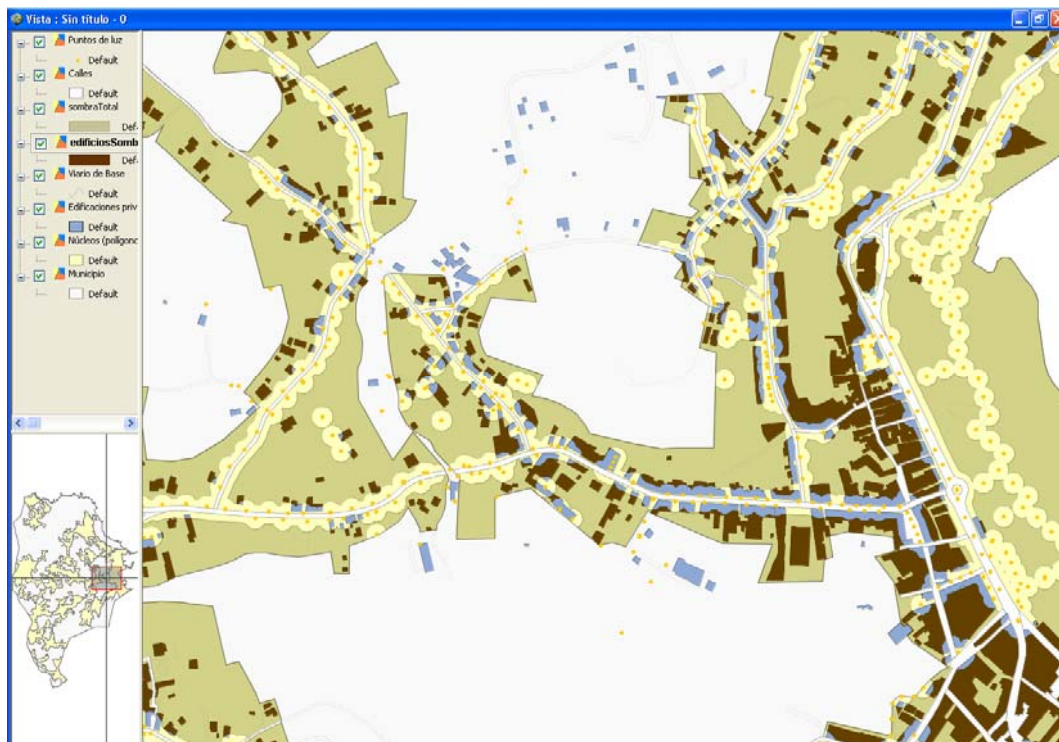
- **syd_1_defi** : la longitud de red con déficit de alcantarillado se calcula a partir de la capa *tra_deficit_red_distr*.

6. NUCL_ENCUESTADOS_7

Se calculan:

- **alu_v_sin**: el número de viviendas sin luz se calcula a partir de una serie de cálculos haciendo uso de capas como *puntos_luz* e *infraestr_viaria* para calcular las zonas iluminadas y, así, derivar las zonas sombreadas.
- **alu_l_sin**: de la misma forma, se calcula la longitud de las calles sin luz.

Ejemplo gráfico del déficit de alumbrado en Sada:



7. PLAN_URBANISTICO

Para cada tipo de figura de planeamiento se calcula la superficie. Los campos afectados son:

- **urban**: superficie del suelo urbano.
- **urbanizable**: superficie que puede ser urbanizada.
- **nourbable_esp**: superficies protegidas.
- **no_urbable**: superficie que no puede ser objeto de urbanización. Se calcula a partir de las anteriores.

8. RECOGIDA_BASURA

Se calculan: **produ_basura** y **calidad**.

Ambos campos se derivan a partir de una serie de operaciones que tienen en cuenta:

- tipo_rbas (tipo de recogida de la basura).
- periodicid (periodicidad de la recogida de basura).
- Padron (población de los núcleos).

9. EMISARIO_ENC

Se calculan:

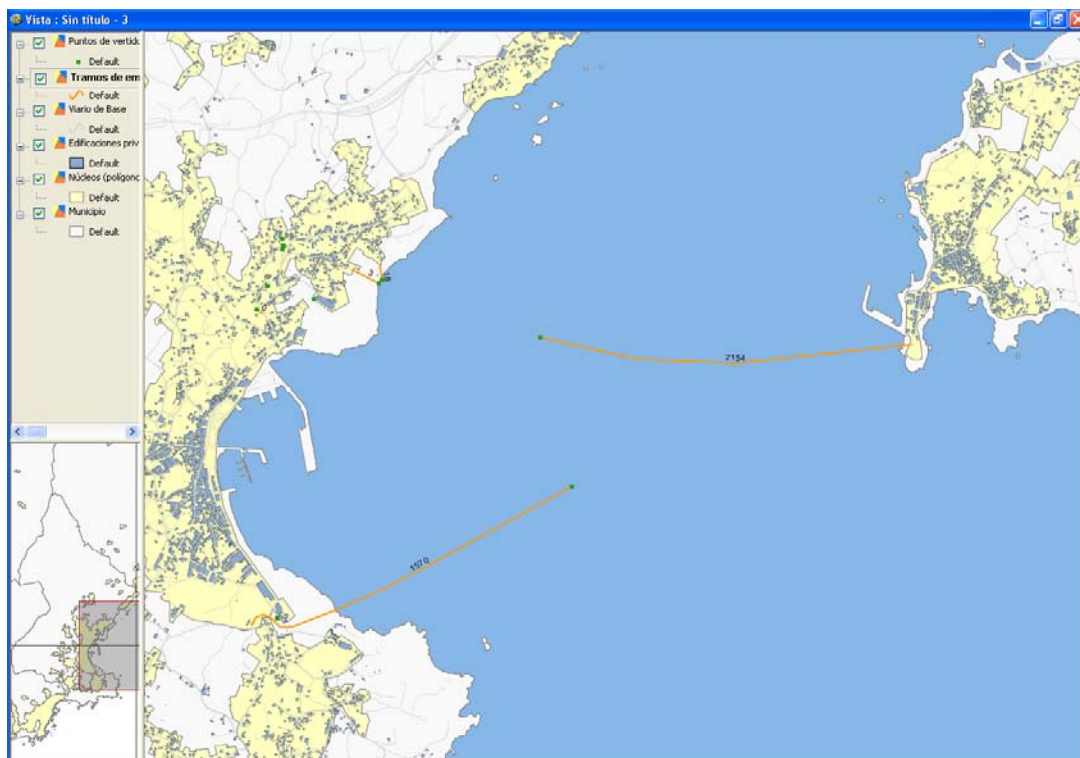
- o **distancia**: se calcula la distancia entre el punto de vertido del emisario y el núcleo más cercano.

10. TRAMO_EMITARIO

Se calculan:

- o **long_terre**: la longitud terrestre del tramo se calcula a partir de la longitud total del tramo y la superficie terrestre.
- o **long_marit**: la longitud marítima del tramo, será la diferencia entre la longitud total y la longitud terrestre.

Ejemplo gráfico donde vemos la longitud marítima de los emisarios (naranja):



11. RED_DISTRIBUCION

Se calcula:

- **longitud:** longitud del tramo de red de distribución.

12. TRAMO_CONDUCCION

Se calcula:

- **longitud:** longitud del tramo de conducción.

13. TRAMO_COLECTOR

Se calcula:

- **long_tramo:** longitud del tramo de colector.

14. RAMAL_SANEAMIENTO

Se calcula:

- **longit_ramal:** longitud de los ramales de la red de distribución.

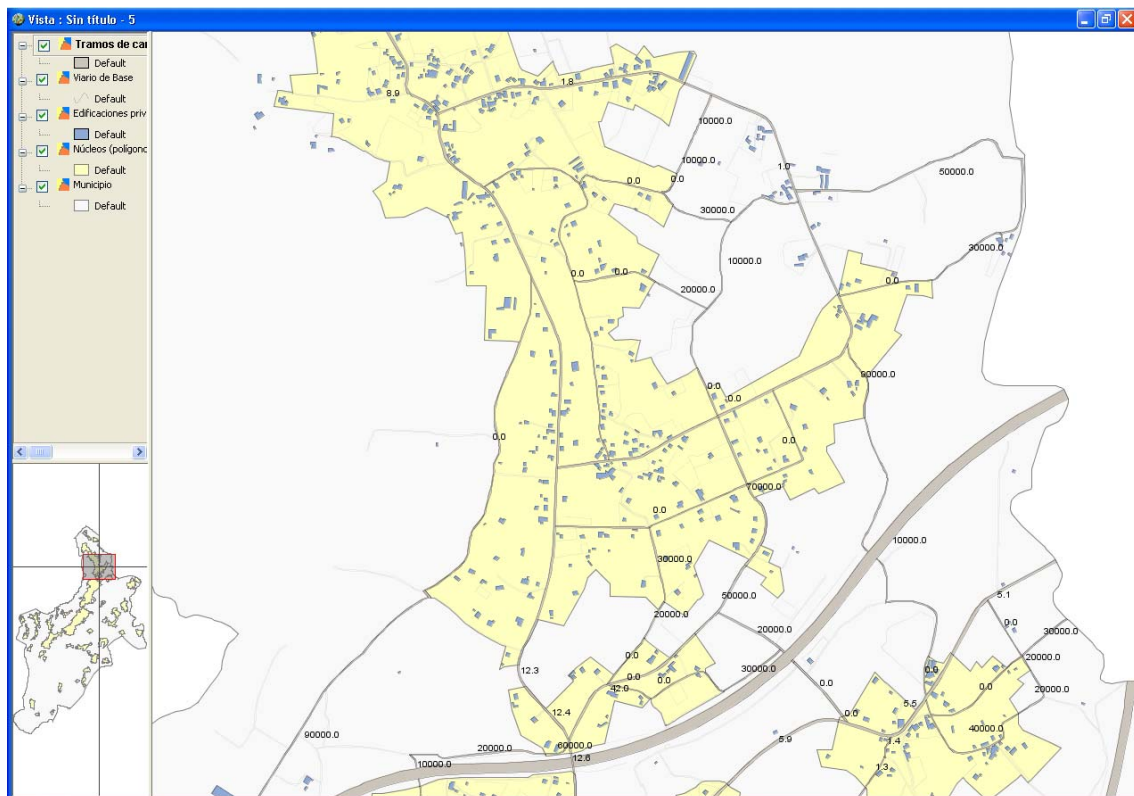
15. TRAMO_CARRETERA

Se calcula:

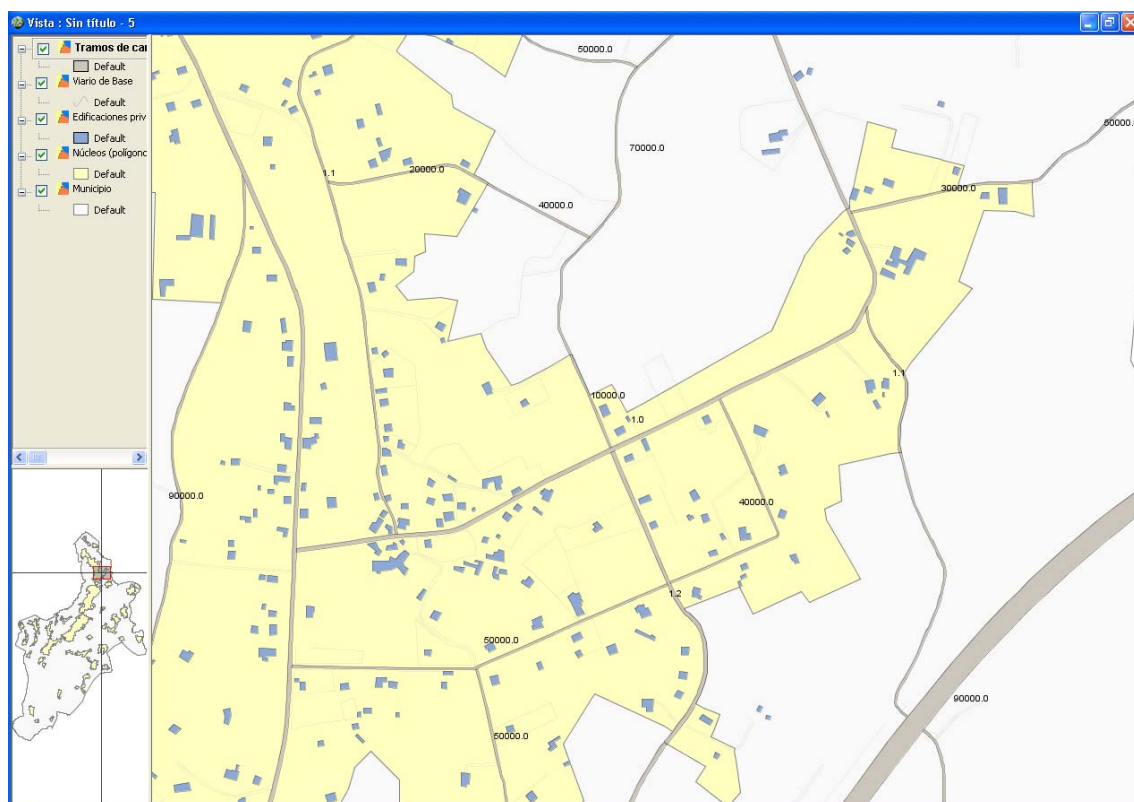
- **pk_inicial:** punto kilométrico inicial del tramo.
- **pk_final:** punto kilométrico final del tramo.

Para el cálculo de estos campos se trabaja de forma secuencial (el punto inicial de un tramo depende del final del tramo anterior) con los tramos para ir sumando la longitud de cada uno y así determinar el punto kilométrico inicial y final.

Ejemplo donde vemos los puntos kilométricos iniciales:



Ejemplo donde vemos los puntos kilométricos finales:



Historial de Cambios

Versión	Fecha	Cambios
0.1	01/04/2008	Verónica Fariña Iglesias: Versión Inicial