

Reading Map 3D Object Data with FME

Overview

A key challenge for users of spatial data is interoperability. Even for a committed user of AutoCAD Map 3D, there are likely to be occasional needs to move data into and out of its native data format.

FME is a spatial ETL (extract, transform and load) platform designed specifically for moving data between different data formats and structures. It is particularly useful for moving data between applications of different types, for example CAD to GIS.

This paper shows how FME meets the challenge of translating and transforming complex AutoCAD Map datasets. The first section provides brief background information on FME and the concept of spatial ETL. The second section focuses specifically on FME's ability to read AutoCAD Map 3D object data.

FME and Spatial Data Interoperability

The key to *transparent* interoperability – i.e. with minimum user involvement – is the mapping of data models between systems. At Safe Software we refer to this term as *Schema Mapping*.

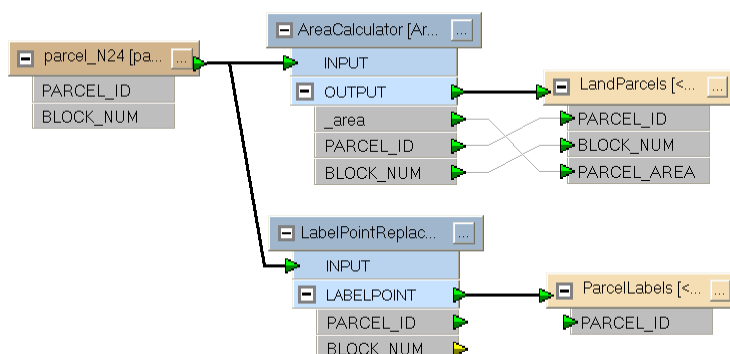
For any given data format, the combination of rules defining its internal data structure are often unique. For example format A might permit upper or lower case attribute names, while B only permits upper case; format C might support many types of geometry, while format D only permits simple lines and points.

Therefore, to convert data for use between systems, an application needs to not only translate the format of data, but also transform the structure of the data so that it seamlessly integrates with the destination format's specifications.

It is this ability to restructure (transform) data between schemas that elevates an application above a simple translation tool into one that is classified as spatial ETL.

Spatial ETL Tools

Spatial ETL tools such as FME typically handle data models in a visual way. Rather than setting up a lookup table or script, the user is presented with a graphic representation of the translation, in much the same layout as a flow diagram. Additional objects – called transformers in FME – may be dropped into the workflow to restructure data mid-translation.



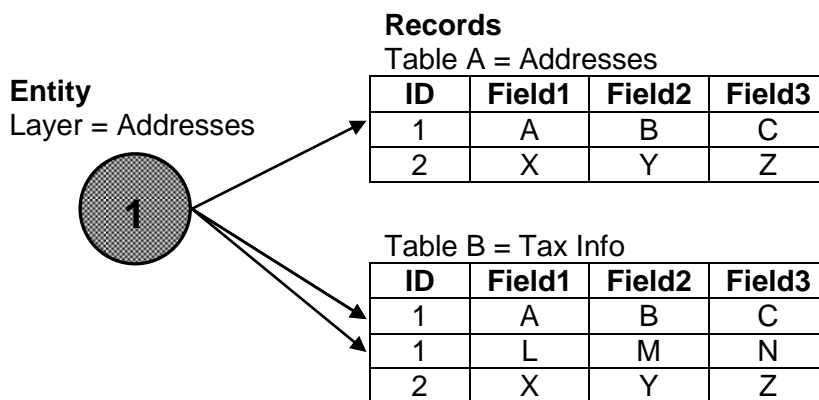
A simple example where polygons in format X are being converted to format Y. The brown objects (feature types) represent data layers. The blue objects (transformers) here measure the area of each polygon and generates a label based on parcel ID.

Although handling data visually is very intuitive, there is a balancing act to follow between an overly simple representation of the data model, and a comprehensive representation that is true to the data model but excessively complex.

The ideal visualization is closely matched to the tasks a user undertakes. Obviously this is difficult to automate when the range of potential tasks is wide, or when the format complex but with a very lenient data structure. AutoCAD Map 3D object data is such a format.

AutoCAD Map 3D Object Data

The AutoCAD Map 3D object data format is typical in that it consists of spatial features (entities) stored on separate *layers*, with links to associated attribute records in data *tables*. However, as the following example illustrates, what is unusual is that any particular entity can possess any number of attribute records in any number of data tables (or none at all), even to the extent of one entity having multiple records in the same table.



Moreover, different tables – even those referenced by the same entity – can have the same field names (notice how both tables have Field1, Field2, Field3), and tables can also carry the same name as a layer within the DWG dataset (Addresses).

While this flexibility maximizes the potential uses for such data, it does make it incredibly difficult to create a standard and predictable view of the data schema. While some of the more extreme structures are not likely to find many real-world applications, a spatial ETL view would, ideally, encompass all possible variations and all possible user perspectives.

Using FME to Read AutoCAD Map 3D Object Data

When FME support for AutoCAD Map 3D object data was added to FME2008, Safe Software developers faced three key challenges:

- 1) To visualize all the potential data structures with maximum usefulness for a user.
- 2) To structure reading in such a way that the data could automatically be translated back to the same format, without loss of information
- 3) To integrate all of the format's unique requirements into the existing product framework and interface



Whereas a single visualization method is applicable for many formats of data, this concept was soon discounted here because of the high number of potential entity-layer-table permutations, and because there are a number of obvious user perspectives that might require different views of a dataset.

Also, a single visualization – if designed for writing back to the same format – might be of limited use to a user with different intentions.

The solution, therefore, was to provide three different modes by which to depict source datasets. This lets a user choose the view of the data that most closely corresponds to their output requirements.

The three different reading modes are "**Group by Entity**", "**Raw Relational**", and "**Group by Object Data**".

GROUP BY ENTITY

Group by Entity simply means that each layer in the source dataset is represented by a corresponding object in the spatial ETL schema. Attribute data is automatically attached to the spatial entities, making this a very good mode for users wishing to translate CAD data in this format, into a GIS application.

Automatically matching records to entities when the format specification is this lenient does lead to special cases, but for the most part these can be handled automatically without the user needing to be aware of the issues.

RAW RELATIONAL

This mode again provides an object in the spatial ETL schema for each source layer. However, in this case, attribute data is not attached to the spatial entities, but is handled by their own set of objects in the schema.

In effect the user is provided with a schema representing the raw data and permitted to manipulate it however he or she chooses. This is the preferred mode for writing data back to a CAD format, even back to the AutoCAD object data format; for example the need to reproject data or modify the schema without translating to a different format.

Because no record matching is taking place, there are fewer special cases to be handled.

GROUP BY OBJECT DATA

Group by Object Data is the opposite of Group by Entity, in that each object in the spatial ETL schema represents an attribute table in the source data. The data – when read – is structured as one feature per record in a data table.

When one entity owns multiple records – for example a line feature that is both state and county boundary – the output is a spatial feature for each record. In effect the data is duplicated once per record, so this mode is particularly useful when wanting to specifically divide up such data, but inefficient when used for any other purpose.

The other consideration in this mode is how to handle entities that do not have associated attribute records. In this case an object in the view is created for each layer that contains such features; a comprehensive solution, but one that can be confusing.



EXAMPLES

It's easier to understand the three modes of reading object data when illustrated with examples.

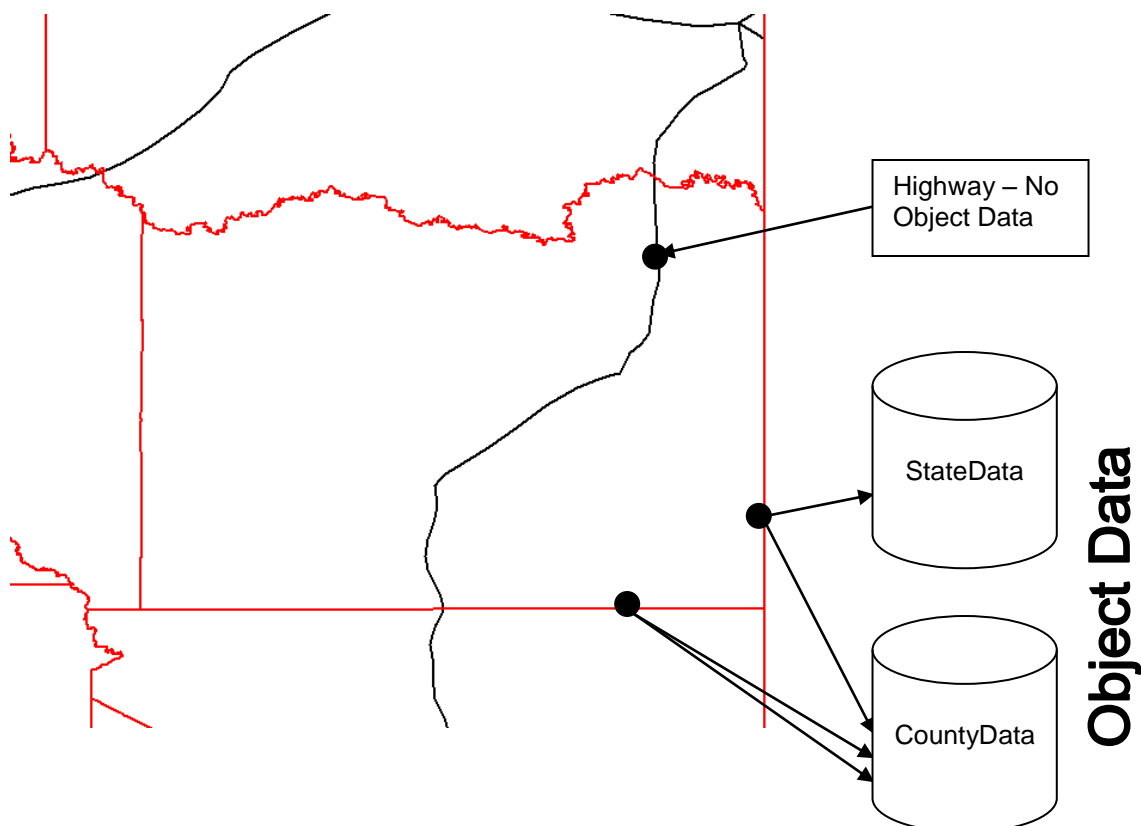
In this example we wish to translate a dataset containing government boundaries for the state of Texas. Each boundary divides either two different counties within the state, or a single county and the adjacent state.

The data itself consists of a set of topologically connected lines with object data identifying which counties (or state) the lines are dividing.

Thus each boundary is a single line feature with two records; rather than two features with identical geometry but different records.

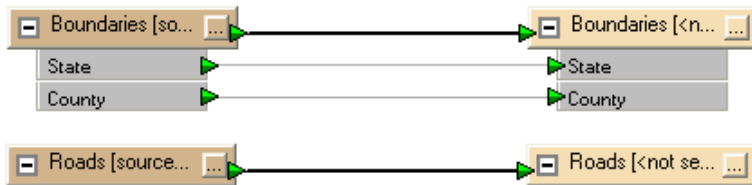
These line features are held on a layer called "Boundaries" while the object data is held in two tables called "CountyData" and "StateData".

The dataset also contains a set of line features representing state highways. These features have no associated object data.



Group-By Entity Mode

When setting up a translation for this example data, in Group-By Entity mode we get a separate object in the spatial ETL workspace for each layer in the source AutoCAD file.



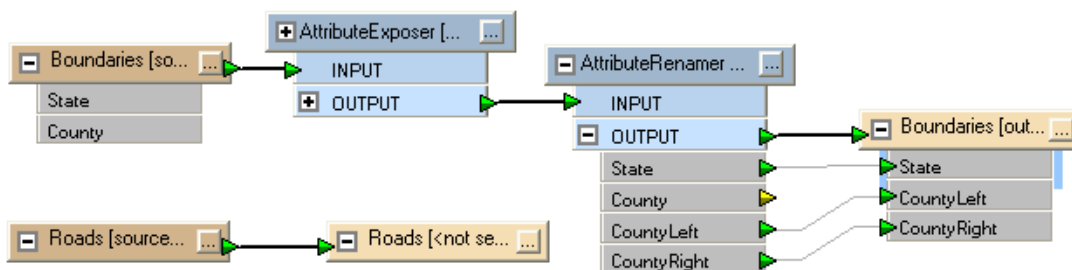
Notice how the Boundaries feature have attributes for both State and County; even for internal boundaries, because both internal and external are being routed through the same workspace object.

The Roads object has no attributes because it has no object data table.

Where there is more than a single record attached to an entity (as in the case of an internal boundary) there will be multiple values for each attribute. These are handled in FME as a "list" type attribute; `CountyData{0}.County` and `CountyData{1}.County`

County	Gaines County
CountyData{0}.County	Gaines County
CountyData{1}.County	Terry County

With a couple of transformers thrown in, we are now set up to write the data to a MapInfo TAB output dataset for use in that GIS application.

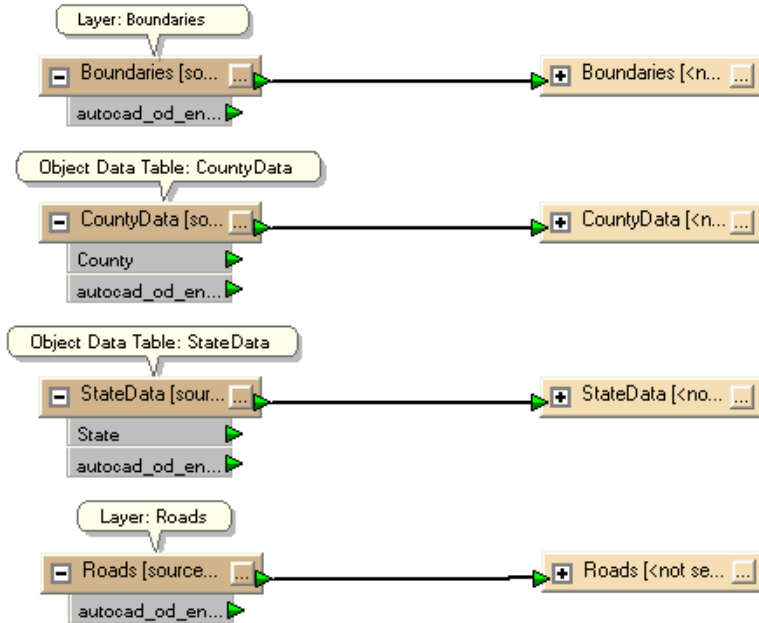


Raw Relational Mode

When setting up the same translation in Raw Relational mode we get a separate object in the spatial ETL



workspace for each layer, plus a separate object for each table, the output of which is non-spatial features.



Notice how both features and object data have an attribute – autocad_od_entity_key – which acts as a lookup between entity and table data. There is also an (unexposed) attribute to show which data table the key refers to.



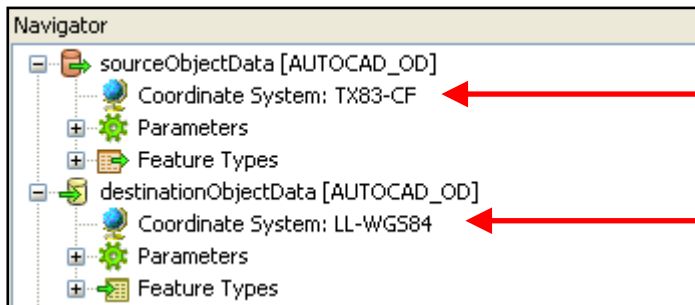
This feature has an entry (entries) in the CountyData table under the key DCD

autocad_ineweight	-1
autocad_map_odtable{0}	CountyData
autocad_od_entity_key	DCD
autocad_original_entity_type	autocad_linnoline

...and browsing the table we see two entries: Van Zandt and Henderson County.

County	autocad_od_entity_key
Van Zandt County	DCD
Henderson County	DCD

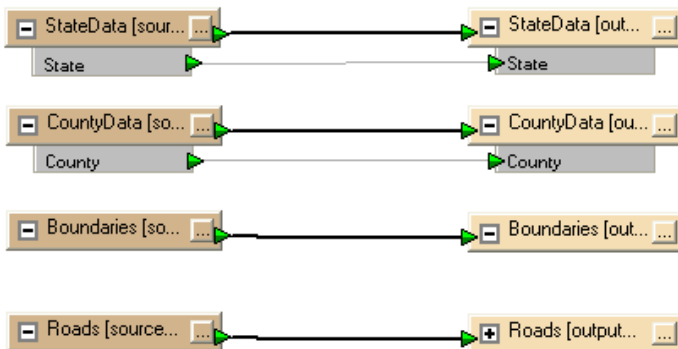
By adjusting a couple of source and destination settings, we can simply translate the data back to AutoCAD Map 3D object data, but in a new coordinate system.



Because autocad_od_entity_key is already defined, writing back to the same format automatically recreates the source schema and matches the correct entities to the correct object data.

Group by Object Data Mode

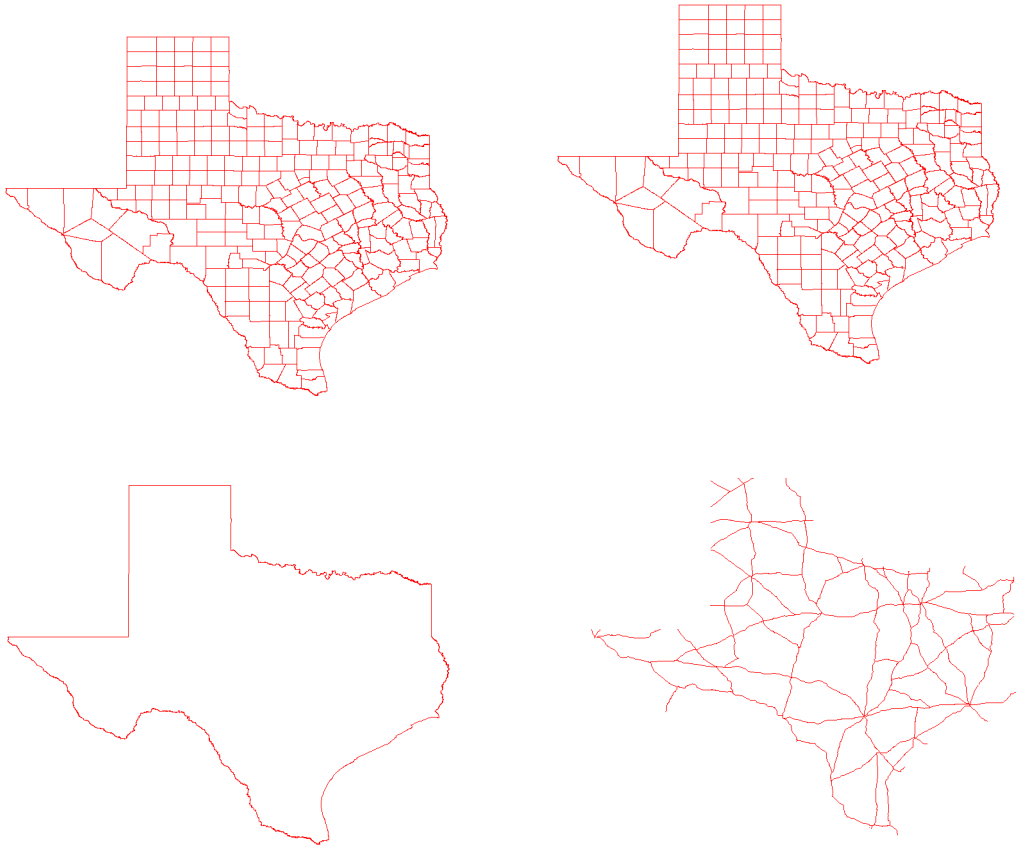
In Group-by Object Data mode the translation is set up as a separate object in the spatial ETL workspace for each table, the output of which is spatial data with attributes. There is also an object for each layer, which consists solely of geometry (ie no attributes).



Notice how the uppermost two objects – representing the data tables – have attributes, while the layer objects do not.

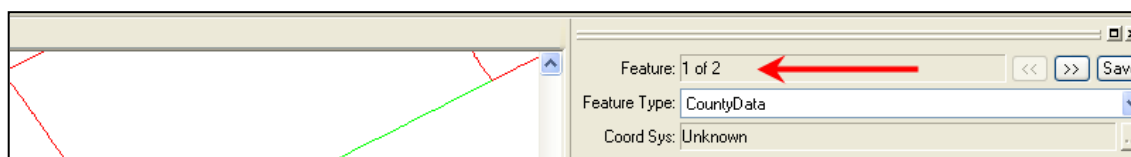
Having objects for both layers and tables causes a duplication of data in the output:

Clockwise from Top-Left: CountyData, Boundaries, Roads, StateData

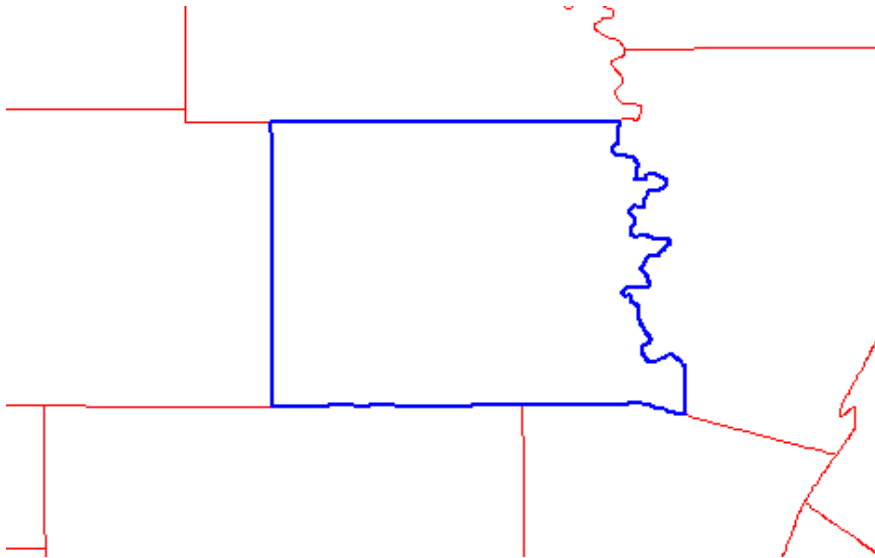
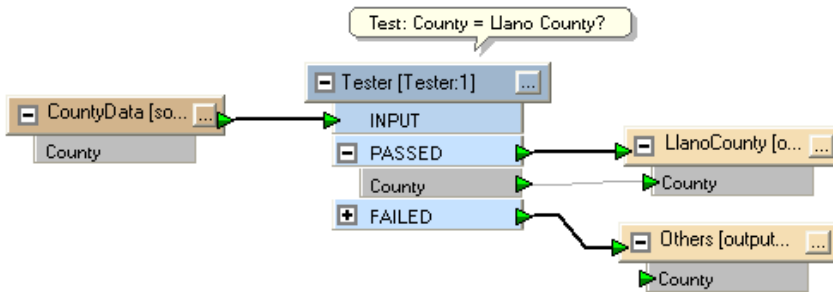


Another feature of this mode is that entities in the same category, with multiple records, will result in multiple output features.

Query an internal boundary, and you find there are two features (one for each county).



Although multiple geometries per boundary seems wasteful, it does allow us to extract a single county boundary, and still leave the remaining boundaries intact.



Find Out More

More examples for each of these three modes are available on the FME wiki, fmepedia: www.fmepedia.com/index.php?title+AutoCAD_Map_3D_Object_Data_Reading. To learn more about FME's object data reading and writing capabilities, additional resources are available at: <http://www.safe.com/fme2008/unlock-object-data.php>.

Summary

AutoCAD Map 3D object data is very flexible, but this very flexibility can hinder interoperability with other systems. FME's ability to read and write this format was specifically designed to visually represent AutoCAD data models in a number of different ways, each equivalent to a particular user perspective.

If there is one ongoing problem it is that the conflicts between source and destination formats are not always recognized by users, and that FME's attempts to automatically reconcile these conflicts are often misinterpreted as flaws in the software. For that reason, the development of FME's object data reader and writer – as with any other new function – has a parallel process of customer education to inform users as to how best to use the new capabilities.

