SAFE SOFTWARE

Technology Brief

GeoMedia and FME - The "Universal Information Integrator" Meets the "Universal Translator"

Summary

GeoMedia's ability to access and integrate data from a number of diverse systems and formats is well known and respected. Recent efforts to integrate FME technology into the GeoMedia framework have extended GeoMedia's reach into many new territories. By giving GeoMedia access to the FME's semantic data translation processor and the vast library of formats that FME reads and writes, powerful integrations and customer solutions can be realized. The paper will discuss the technical approaches used for the integration, and survey some of the solutions this integration is already providing to customers.

Introduction

Although the base GeoMedia product is well served with a number of interfaces to a variety of formats, customers have requested that GeoMedia be integrated with the FME Universal Spatial Data Translator and its interfaces to a myriad of systems. This integration provides GeoMedia users with access to a wealth of complementary functionality, including the ability to read and write to a large number of otherwise inaccessible formats.

From its very inception, GeoMedia's architecture provided for extensibility. The "Geographic Data Object" (GDO) design allows new formats to be "plugged into" GeoMedia. Through the GDO Software Development Kit (SDK) supplied by Intergraph, building such plug-ins involves implementing against a set of predefined interfaces. Then, once the GDO is completed and installed into a GeoMedia user's environment, it becomes available as an accessible "warehouse" of spatial data. A very important aspect of the FME integration with GeoMedia was to create a GDO that allowed direct access from GeoMedia to any of the more than 70 formats that FME supports.

Many customers wanted to be able to export data out of GeoMedia into a number of different formats, but the GDO SDK does not currently allow for "write" GDOs to be built. However, user requirements did not stipulate that the data had to be exported from within a GeoMedia session. Users also expressed an interest in being able to bring data into GeoMedia Access warehouses from outside of a GeoMedia session. To accommodate these needs, FME's ability to accept reader/writer "plug-ins" was exercised. Using the FME Plug-in Builder SDK, both a reader and a writer were built for GeoMedia Access warehouses . The result is that all of the FME translation suite capabilities - including translation in and out of a wide variety of formats, geometric and attribute manipulations, coordinate transformations, and visualization with the FME Universal Viewer - can be applied to GeoMedia datasets.

GeoMedia Meets FME Objects

In order to provide direct access from within GeoMedia to any of the formats and systems that FME reads from, a GDO interfacing was created to the FME component library called FME Objects. FME Objects provides programmatic access to all of the FME readers, writers, and processing facilities. FME Objects presents a common data model to its clients that is independent of format. This characteristic allowed a single GDO to be written on top of FME Objects and still access all of the FME formats. For example, the GDO would not care to distinguish between data read from an SDE database versus data read from an SDTS dataset. In all cases, only the attributes and feature types would differ.

FME Objects can be used through a variety of interfaces, including C, C++, Java, COM, and Delphi. For the FME Objects GDO, the C++ interface proved to be the most straightforward and efficient method because the GDO SDK is also in C++.

Building an FME Objects GDO for GeoMedia was not without its challenges. Besides understanding the two sets of interfaces involved, and building the glue between them, the largest hurdle was the user interface. The GeoMedia framework provides for the extension to the "connection wizard" used to specify data sources. This approach to obtaining these parameters did not match the approach assumed by the FME Objects dialog package, which assumed a modal dialog would be used.

In order to provide access to all the FME Objects formats without creating a custom wizard panel for each one, a hybrid approach was chosen. Custom FME Objects wizard panels were created to ask for the format type, and for the dataset location. For commonly used formats, such as SDE, additional panels were created to ask for further connection information. For less commonly used formats, a single wizard panel was created that provided a means to invoke the FME Objects "settings dialog" for the format. This pragmatic solution minimized the development work needed, while at the same time providing an optimal user experience for frequently used formats.

Related to the user interface issue was the specification of an area of interest when the data source being accessed was a large spatial database. Presently, the user must key in the bounds of the area of interest. This is an awkward and error-prone solution, and so work is underway to explore alternate ways of indicating a region of interest that could adapt to the area of interest currently shown in GeoMedia.

Resolving the interface differences between the GeoMedia GDO object model and those provided by FME Objects was more challenging in some areas than in others. In some cases, no straightforward mapping between the two models was apparent, and so considerable effort was needed in order to present the correct information from FME Objects through the GDO to GeoMedia. In other cases, the mapping was fairly simple and straightforward. For example, the "schema features" of FME Objects were easy to map into the corresponding record definitions required of the GDO.

FME Meets GeoMedia

When building a GDO for GeoMedia, the GeoMedia application was in control. When FME itself reads and writes GeoMedia Access warehouses, the FME application does the driving. Several hurdles had to be overcome when building the interface from FME to GeoMedia Access warehouses, but the end result provides much value to the end users.

The GeoMedia Access warehouse reader for FME was built by directly accessing the Access database from C++ through ODBC. In this module, the documented geometry structures in the geometry BLOB were decoded on the fly and presented to the FME as proper FME features. All other attributes were read using traditional SQL calls.

Because the geometry model of GeoMedia does not completely match that of the FME, some of the compound geometries required some effort to transform into a supported representation. For example, area boundary geometries in GeoMedia may consist of several linear pieces, which taken together form the area. FME requires formed shapes, and so processing is required during reading to turn the linear pieces into polygons.

Another mismatch occurred when reading text. In FME, it is necessary for annotation entities to carry with them a height in ground units. In GeoMedia, there is no such restriction - in fact, the text entities have no size information stored with them. This presents challenges when moving the annotation to another system. The workaround

solution is for FME to pick an arbitrary height to apply to all annotations, and allow end users to adjust this height.

The GeoMedia Access warehouse writer for FME was built using the GeoMedia COM objects for writing to Access warehouses. Calling COM from C++ presented significant issues. In general, using COM from C++ is not pleasant, as much type conversion and coercion is necessary. In specific, it was even more difficult because the example GeoMedia applications tended to be written in Visual Basic, and the equivalent calls to make in a C++ environment were not always obvious. Ultimately, victory was had and the results speak for themselves.

Two final issues that presented difficulties in the GeoMedia/FME connection concerned arc and coordinate system conversion. GeoMedia represents only circular arcs, not elliptical ones, and does so by storing three points and an indication of the direction the arc takes through these points. FME supports elliptical arcs, and stores a major and minor axis, rotation, start and sweep angle. When writing a non-circular arc to GeoMedia, the arc is stroked into a line and stored as a linear geometry. However, circular arcs are stored as arcs, after the math is done to convert the representation. When reading arcs from GeoMedia, the representation is converted and an elliptical arc (whose major and minor axes are identical) is returned.

Converting coordinate system representations was the single most difficult aspect of the entire project. Both FME and GeoMedia have extensive support for coordinate systems, including a large number of projection types, ellipsoids, and datums. GeoMedia has an additional transformation matrix that stores a conversion from the storage units into metres, whereas FME allows arbitrary units of measure for its coordinate systems. Finally, GeoMedia considers the state plane and UTM zones as predefined coordinate systems, rather than storing them as a parameterized projection, as FME does.

Getting the conversion between FME coordinate systems and GeoMedia right took patience and time, and some very large configuration files. Ultimately, mappings between the projections and their parameter names, ellipsoids, datums, and units of measure were derived and stored in ASCII configuration files that ship with FME. The best news is that it works!

User Applications

The most gratifying aspect of the GeoMedia/FME integration is the way that it simplifies and enhances the users' workflows.

One organization is using the FME Objects GDO to directly read data stored in ESRI's SDE into the GeoMedia environment, using it as a backdrop for other data layers from other environments. This application highlights GeoMedia's role as the "Universal Information Integrator." Prior to the availability of the GeoMedia/FME integration, the only other way to accomplish this task was to export the data from SDE into Shape files, and then use the Shape files as a data source. This was time-consuming and often impractical if the data volumes were large.

For another organization, the ability to read and write GeoMedia Access warehouses directly from FME has provided several shortcuts in their workflows. For example, data from the Access warehouse can be exported directly into binary MapInfo TAB files, to be sent to MapInfo users in the organization. As well, batch imports of thousands of parcels from CAD files can be run unattended, and the semantic manipulation possible in FME can be applied during the translation to create a result that is immediately meaningful to the end users.

Yet another application involved using FME to perform a GeoMedia Access warehouse-to-Access warehouse translation. In this situation, an Australian GDA datum shift was performed on the data, moving it to the latest GDA94 datum system to be used with other data in the organization. Prior to the GeoMedia/FME integration, it was

very difficult to perform this specialized datum shift.

Conclusion

Extending GeoMedia with FME Objects technology through the creation of a new GDO and extending FME with a GeoMedia Access reader and writer were both technically challenging and interesting projects. Although it has just recently become available, these two integrations have already simplified many workflows for GeoMedia users. The remaining barriers to certain types of data integration with GeoMedia have been overcome!