

Generating Alignment Sheets in ArcInfo

by Christopher Eykamp
July 1998

The Camisea region of Peru contains vast energy reserves, estimated at 11 trillion cubic feet of natural gas, and 600 million barrels of gas liquids. Getting this gas to market will require utmost sensitivity to the area's remarkable ecosystem and indigenous peoples — as well as building the world's most ambitious natural gas transportation system.

Shell and Mobil joined forces to attempt this project. To design and build the pipeline, Shell created a consortium with Bechtel (US), Cosapi (Peru) and Odebrecht (Brazil), called BCO. Although ultimately Shell, Mobil, and the Peruvian government could not come to an agreement on crucial issues related to the establishment of a natural gas market, and the project was cancelled, the six months of design work presented a unique opportunity to push forward the state-of-the-art in GIS techniques.

Alignment Sheets

As with any large construction project, the design and construction of the Camisea pipeline required alignment sheets. These are plans that show, in great detail, construction details of the pipeline, the conditions expected along the route, and important features in the surrounding area. Generation of these drawings has traditionally been done in CAD, which works well under ordinary circumstances. But Camisea was far from ordinary.

The construction plan called for twin pipelines extending 700 kilometers from the Camisea gas fields in southeastern Peru to the Pacific coast south of Lima. The route ran through some of South America's thickest jungles, cut by rivers, ridges, and canyons, and climbed almost 5,000 meters over the Andes, higher than any pipeline ever attempted. Complicating matters, the region demanded extraordinary environmental sensitivity to its ecosystem, remaining indigenous cultures, and archaeological heritage.

Due to the remoteness and difficulty of the terrain that the pipeline would cross, the route was constantly in flux during the design phase as new information from the field became available. Furthermore, one of the critical requirements of this project was that it give greater weight to environmental and social factors than has traditionally been the case, and this involved maintaining several dynamic databases. Because of the complexity of using ever-changing data sources with CAD programs, project managers decided to try something new and use GIS to generate the alignment sheets. While some commercial GIS alignment sheet generation packages are available, they work best with a static pipeline route. For this and other reasons, such as the need to modify the program in-house, the GIS team decided to develop their own alignment sheet generation system using ArcInfo as the base GIS, running on an NT network.

Alignment Sheets can take many forms, but the style preferred by the engineers and planners at BCO generally divides a landscape A3 page into two sections. The top half is a fairly standard GIS map — a plan view of a section of pipeline, typically between one and ten kilometers long, with other features of

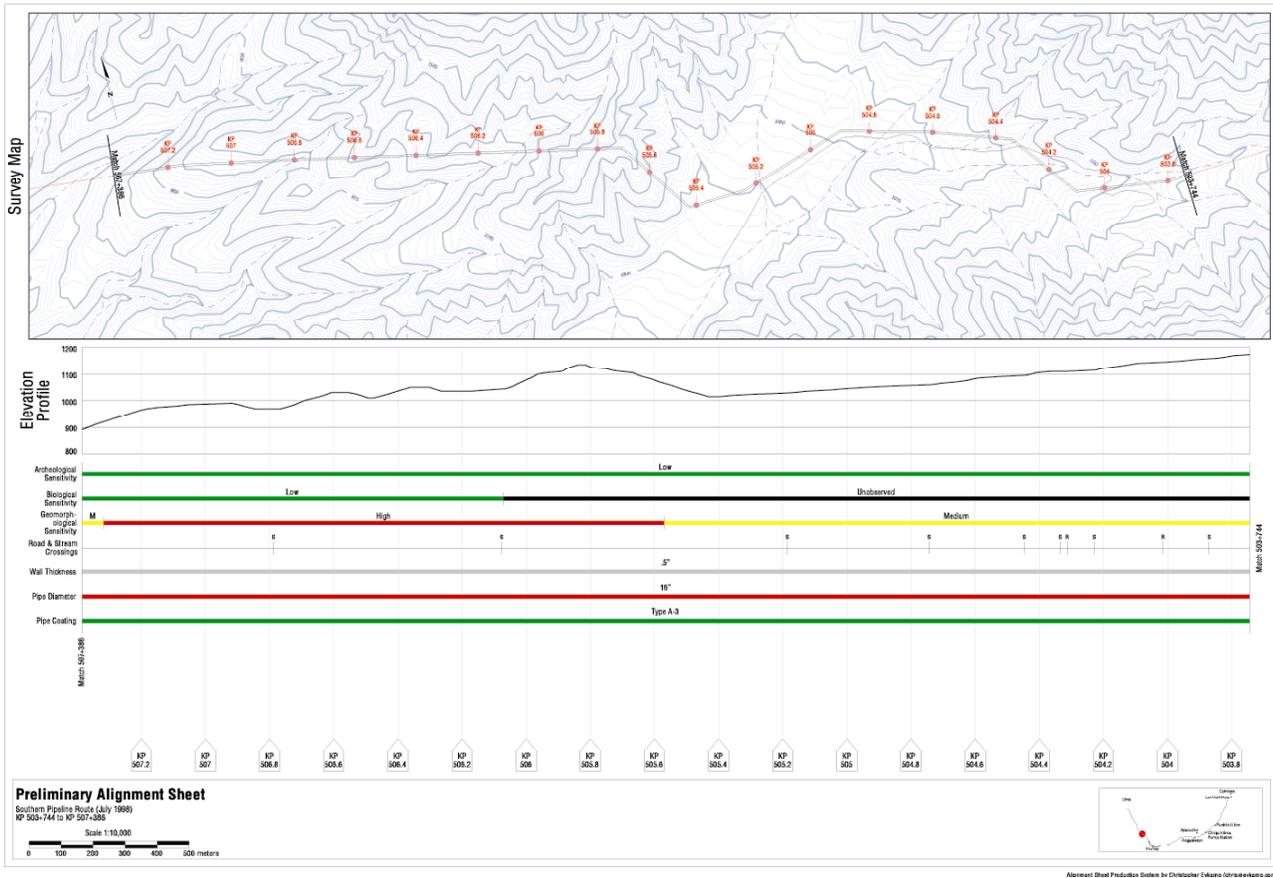
interest such as villages and rivers added to provide spatial context. The bottom half of the sheet is subdivided into several smaller “panels”, each showing a particular type of information using a schematic view. There is usually an elevation profile, showing the altitude and slope of each segment of the pipeline. This presents a simple graphic illustration of the terrain over which the pipeline passes which, considering the extremely rugged landscape, was critical information. Each sheet also shows several stripmaps, displaying data ranging from construction details such as pipe wall thickness, coating material, and valve locations, to environmental data such as areas of environmental and ecological sensitivity, or proximity to important archaeological features. These sections show the pipeline “stretched out” as if it were a straight line, and is a compact and efficient device to display the characteristics of a linear feature.

Flexibility and Automation

There were two important considerations involved in the generation of the alignment sheets: flexibility and automation. The design of the sheets had to be flexible — given the variety of

Data Collection

Typically, building a database for a project the size and complexity of Camisea is a huge challenge, especially when working in areas that have never before been mapped. Fortunately, BCO was prepared to invest in basic data collection. They contracted a firm to take thousands of aerial photographs of the proposed pipeline route, from which they created a series of base datasets, such as contours, roads, rivers, and buildings. Other data proved more problematic — survey teams were not provided with any training about how to organize the information they collected, and as a result, incoming data was difficult to manage. It took several visits to the field to agree on a system that would work both in the lab and in the difficult conditions outside. (In one camp, at an altitude of over 4,000 meters, surveyors faced daytime snow gales and nighttime temperatures that dropped well below freezing, while less than 50 miles away, *topografos* in isolated camps battled malarial mosquitoes and fire ants in the sweltering tropical jungle, with only a static filled radio to connect them with the outside world.)



Sample Alignment Sheet

data to be presented and the range of potential audiences, it had to be easy to rearrange and change the data displayed on a particular alignment sheet. For example, an ecologist might not need information regarding pipe coating materials, but would be very interested in knowing where the pipeline entered a macaw nesting area. A hydraulic engineer, on the other hand, would want detailed information about slope and pipe material, but not about the sensitivity of the local fauna. The system had to be flexible enough to encompass a range of different display types, each with its own sizing, positioning, and data requirements. Therefore, the program was designed to be completely modular, allowing the plan map, pipeline profile, stripmaps, and other displays to be reconfigured, resized, or omitted altogether through a simple configuration AML. Thus each client's preferred map layout could be retained as data was updated and sheets had to be

regenerated, and the various "rendering engines" could be upgraded individually.

Because each sheet showed only a small section of pipeline, dozens or hundreds of sheets were required to show conditions along the entire length of the project, depending on the scale. Therefore, the process of generating the sheets had to be fully automated. Because each "customer" required different types of data to be displayed, and the data itself was constantly in flux, hundreds of runs were envisioned over the life of the project.

Automation had one additional benefit – it cut costs dramatically. Though there was an initial development cost to get the system up and running, a single operator could produce a set of alignment sheets "in the background" while continuing to perform other tasks. Contrast that with the ongoing expense of maintaining a team of CAD operators to make changes

to large numbers of alignment sheets manually.

Technical Challenges

Perhaps the greatest technical challenge of the project involved the generation of the elevation profile and stripmaps. ArcInfo's route/event model provides an easy way to generate stripmaps, but because of the constantly changing pipeline route, it was decided early on not to use this feature, and instead all rendering routines were written from scratch. Given the sources and structure of the data coming in to the system, constant recalibration of the route systems and event databases could easily lead to human error – if the recalibration involved small changes, it would not always be obvious if a particular data layer had been updated or not. This mirrored another problem encountered in the early stages of data collection; the project received data in two similar

projections which differed from one another only by about 200 meters, resulting in considerable confusion.

The profile generation system posed similar challenges. Complex algorithms had to be developed to calculate the exact position of thousands of elevation points in an efficient manner. The key was interfacing AML with an embedded Perl program, which was able to carry out many calculations quickly “in bulk” rather than individually using ArcInfo’s cursors (see sidebar).

Despite the challenges, the alignment sheet project was a success. A considerable amount of data was collected during the course of the project, and the programs have proven themselves. Although the termination of Camisea has halted work for now, it is likely that the system will be developed further and deployed on other projects in the future. This will be possible because its modular design creates a flexible, evolving toolkit for generating customized alignment sheets for any project.

Christopher Eykamp is a GIS consultant currently working with Bechtel. He recently completed a map generation system using a wizard-style interface that is being used to create materials for the Sydney 2000 Olympics. He can be reached by email at chris@eykamp.com, or through his website at <http://www.eykamp.com>.

Using Perl with AML

It is possible to embed Perl code directly in an AML file. AML and Perl are not directly compatible — that is, if ArcInfo encounters a Perl command, it will return an error, and vice-versa. Luckily, due to the natures of the two languages, it is possible to keep them isolated despite being stored in the same file.

AML is an interpreted language. This means that the AML processor executes a program line-by-line, and does not look at any lines the thread of execution does not encounter. Therefore, if we keep ArcInfo away from the Perl code, by using a `&return` or `&stop` statement, the AML interpreter will never know that there is alien code embedded in the file.

Perl, on the other hand, is a compiled language, which ordinarily means that the entire file is checked for syntax errors before being compiled, and is then stored in its compiled binary state before being run by the user. Both of these factors would seem to work against us.

Perl, however, has several interesting features which allow us to overcome these problems. First, Perl is only compiled at run time, which means that, unlike other compiled languages such as C or Pascal, the user handles only the source code. Perl code, like AML, is simply text. Second, Perl has ability to extract its source code from another file, using the `-x` option. This feature was designed to make it easy to run Perl programs sent by email, and directs the Perl compiler to ignore anything before the first instance of `#!/perl`. This allows us to put our AML code at the top of the file, where the AML interpreter will find it, with Perl code at the bottom, where the Perl compiler will extract it.

The magic happens when the AML interpreter encounters the following line:

```
&sys perl -x %aml$fullfile% %tmpfile1% %tmpfile2%
```

The `&sys perl` bit tells ArcInfo to fire up a system shell and run the Perl compiler, passing the remainder of the line as arguments. Perl compiles and runs the program file stored in the special AML variable `%aml$fullfile%`, in which ArcInfo stores the name of the currently executing AML (thus relieving us of the need to code the name and physical location of the AML into itself). The `-x` option tells Perl to ignore everything until it encounters an instance of `#!/perl`, and the two temporary files set up earlier in the AML are passed as program parameters. The embedded Perl program needs to be set up to get its input from the file passed as the first parameter, and write its output to the file passed as the second, which can then be further processed by the AML.

(This text extracted from <http://www.eykamp.com/gistools/perlaml>, which contains more information on using this technique.)