**Example Project: Documentation**

> *This document is designed to help FME users submit a project for consideration as part of the FME Certified Professional accreditation process.*

One of the key components of a project is a project description. Without it, it is very difficult to understand what a project is designed to do and why.

The following is a description to go with the included example project.

Remember, that this is just an example of how a project might be documented. Your own work may be very much different and include different details and information.

---

**Project Name** Bradgate Park Legacy Data

**Project Description** Translation and transformation of a GENIO format dataset

**Project Date** June 2011

**Included Files** The project files include a source dataset, FME workspace, and example output

**Detailed Description**
A topographic land survey of Bradgate Park, Leicestershire, was undertaken in 1990. The data was processed using an application called MOSS, and the data stored in a text-based format called GENIO.

Because the data now needs to be used in ArcGIS, and because neither MOSS nor any other application capable of reading GENIO data now exists, FME was chosen to read and translate the data into a suitable format such as File Geodatabase.

**Source Data**
GENIO (GENeralized Input Output) is a text-based format defined by a series of “major options” and “minor options”.

Major options are keywords that define the actions to be carried out on incoming data (for example DELETE, CREATE, COPY). These can be largely ignored as the intention is just to extract data.

Minor options can be considered as a header to each feature. They define the structure of the data within the file, the feature type of the data, and other aspects such as dimension and end-of-feature markers.

For example, minor option 001 defines the data structure, and option 080 defines the feature type.

In the snippet below, the 001 option tells us that the following data is composed of two sets of coordinates, each of which has an X and Y coordinate (Floating Point, 15 digits long, 3 decimal places) plus a Z coordinate (9 digits, 3 decimal places).

080 tells us the feature type (WS) and – importantly – that the end-of-feature markers are -1.0

```
001FORMAT(2(2F15.3,F9.3))
080WS                    0.0   0.0   3.0   -1.0  -1.0
      170.036      1750.610 -999.000       173.044      1750.325 -999.000
      176.054      1750.089 -999.000       179.065      1749.928 -999.000
      -1.000       -1.000
```

**Proposed Solution**

The ideal solution would be a full GENIO reader. However, since the aim is a one-off extraction of data, it is only necessary to do enough work to read the subset of format contained in the dataset.

In this case there are only two different data “formats”:

```
001FORMAT(2(2F15.3,F9.3))
001FORMAT(2F15.3,2F9.3,/,11A4
)
```

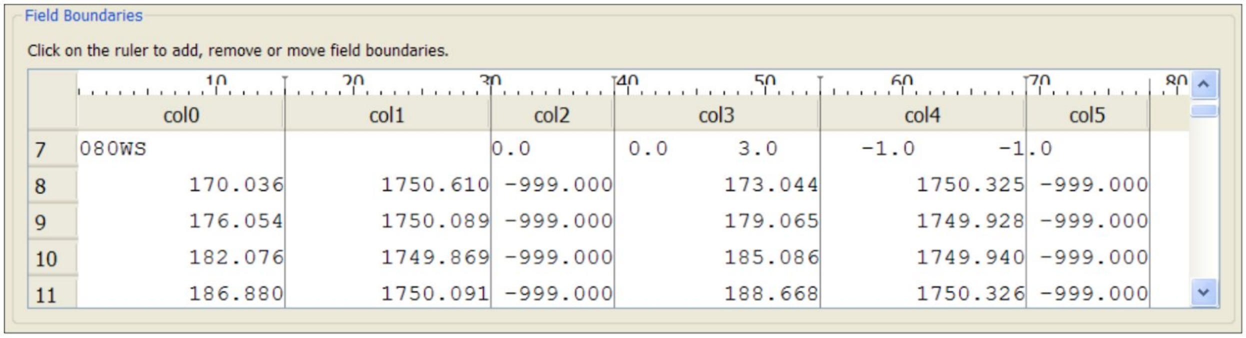
The first of these denotes features as described beforehand. Of the second, the “/” character denotes a new line, and 11A4 denotes a text string (presumably 11x4=44 characters in length)

So, all data can be treated as the same structure and there is no need to be able to parse the 001 option. All that needs to happen is that text strings are treated as a special case.

Because the source data is a text-based format, it is capable of being read by several FME readers such as the Text File reader, CSV reader, or – new for FME2012 – the CAT (Column Aligned Text) format reader.

Since the data structure is the same throughout the file, it can be thought of as Column-Aligned text and read with the new FME2012 CAT reader.

**FME Solution** The solution consists of a single workspace. Data is read with the CAT reader. The parameters dialog lets me define each coordinate value as a separate column.



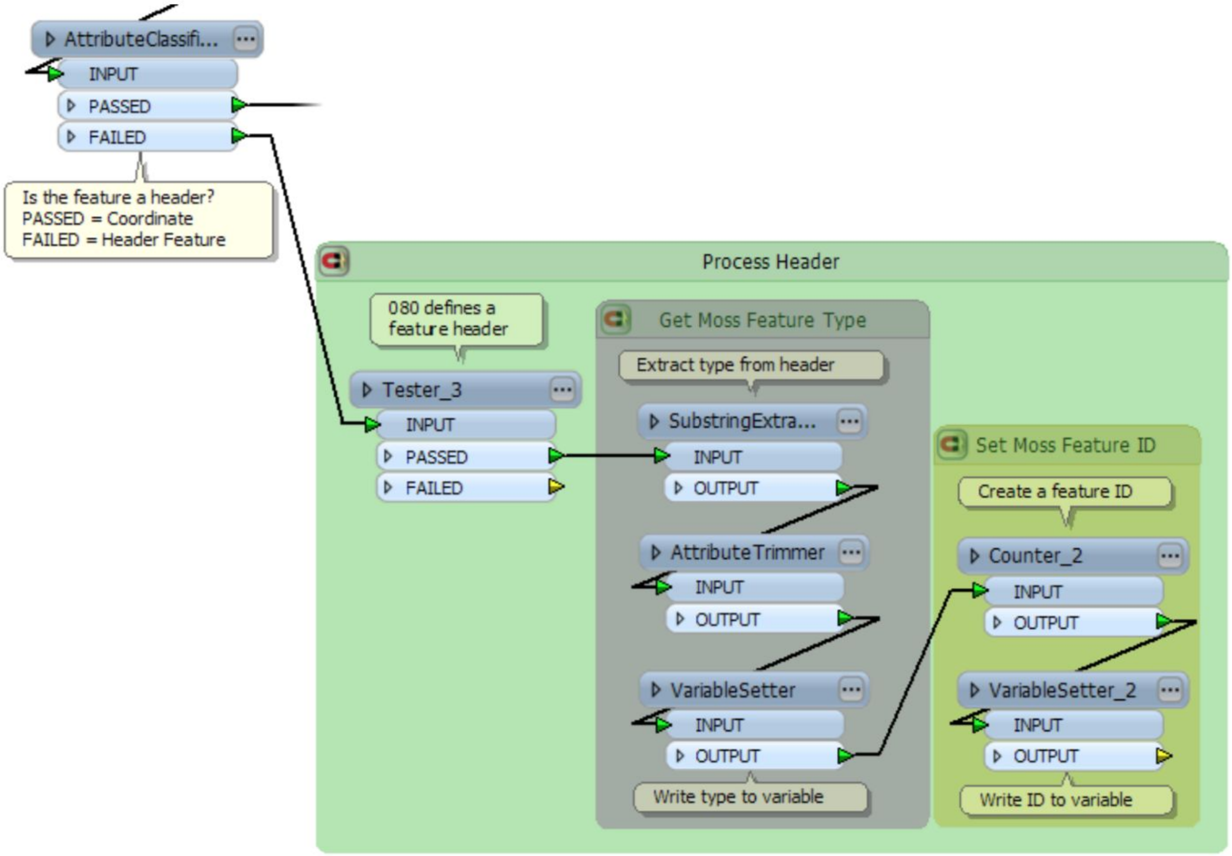
Field Boundaries

Click on the ruler to add, remove or move field boundaries.

|  | col0 | col1 | col2 | col3 | col4 | col5 |
|---|---|---|---|---|---|---|
| 7 | 080WS |  | 0.0 | 0.0      3.0 | -1.0     -1 | .0 |
| 8 | 170.036 | 1750.610 | -999.000 | 173.044 | 1750.325 | -999.000 |
| 9 | 176.054 | 1750.089 | -999.000 | 179.065 | 1749.928 | -999.000 |
| 10 | 182.076 | 1749.869 | -999.000 | 185.086 | 1749.940 | -999.000 |
| 11 | 186.880 | 1750.091 | -999.000 | 188.668 | 1750.326 | -999.000 |

Because each feature is made up of a series of lines, it is necessary to store the current feature type in an FME variable. Subsequent lines of the file that make up the same feature will receive the same feature type by retrieving the value of that variable.

So, the first part of the workspace tests for a header record then – having found one – extracts the feature type, sets a feature ID, and records these to FME variables.



Is the feature a header?
PASSED = Coordinate
FAILED = Header Feature

080 defines a feature header

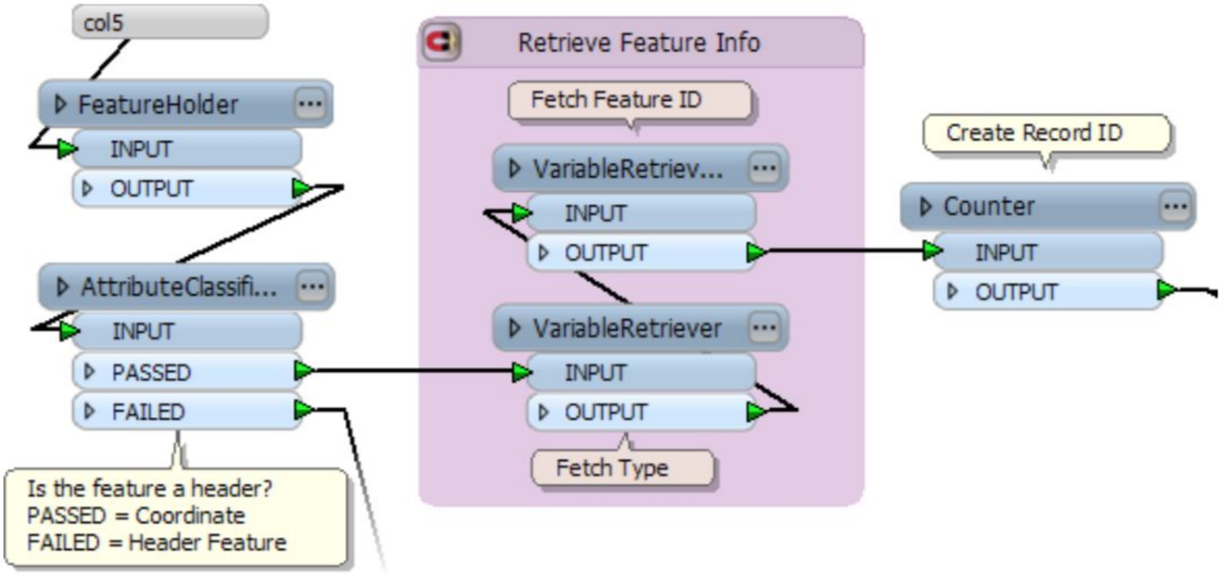
Extract type from header

Write type to variable

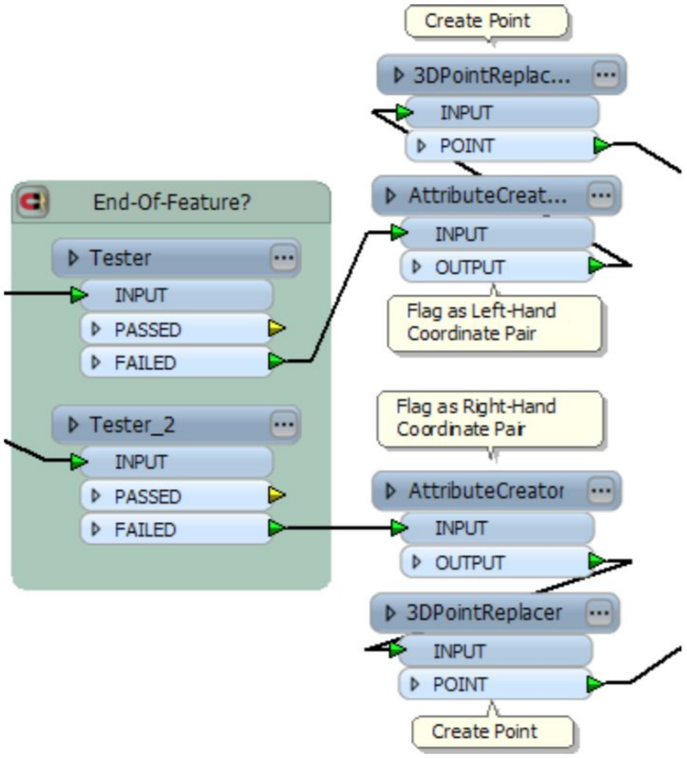
Create a feature ID

Write ID to variable

When the next line is a coordinate pair (i.e. not a header) then they retrieve the previously set variables, and create a Record ID (e.g. Feature 22, Record 4):

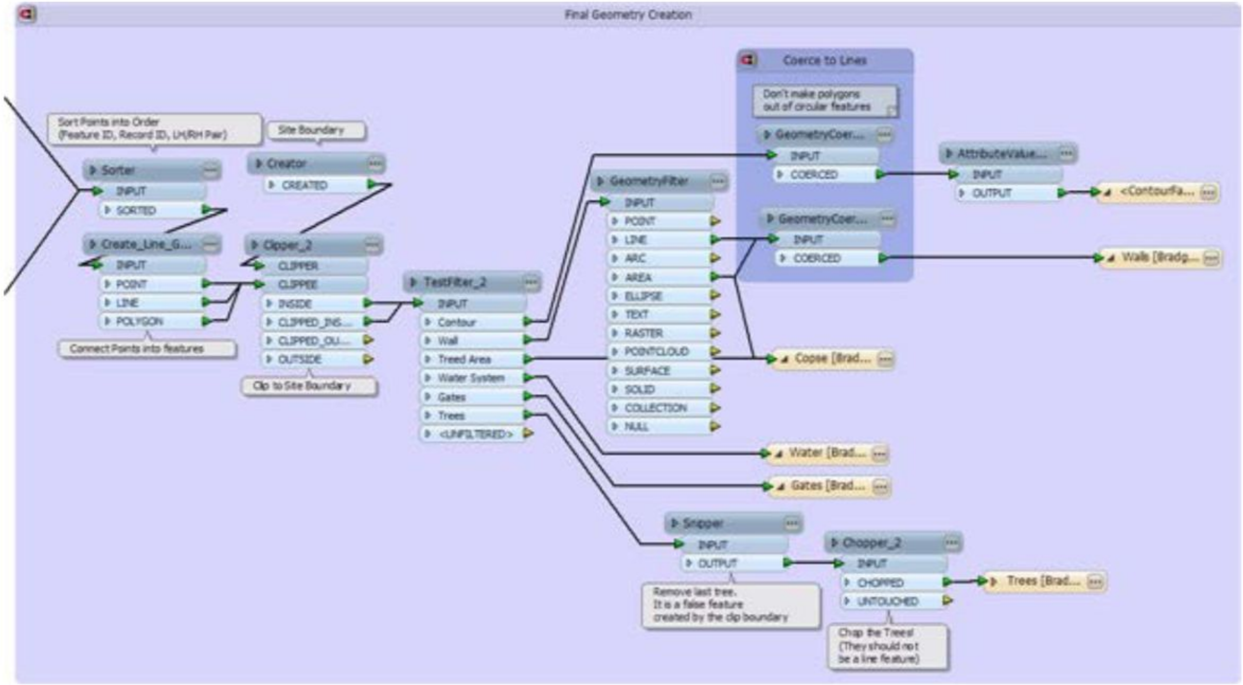


The next section of workspace checks for the end-of-feature marker, then creates a point for each coordinate pair. Remember, each line has two coordinate pairs on it – one is flagged A and the other



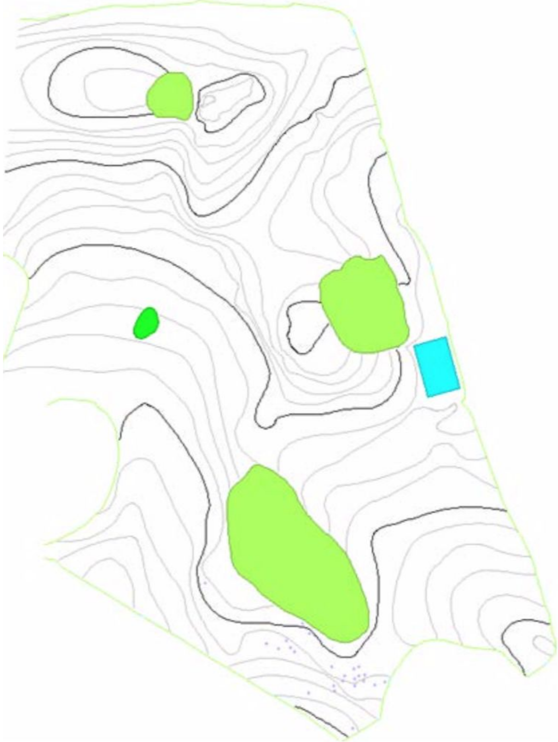
B.

The final part of the workspace sorts the point features into order, turns them into the final geometry, and finally writes the output to Geodatabase:



## Results

The result of this workspace is a set of data that – while not perfect – can certainly be written to a suitable format and edited until it meets the required standard.



The only part of the data that is not handled properly is text. The features are there, but do not include text strings, and would require manual editing.