

Canadian Council on Geomatics Interchange Format (CCOGIF) Reader/Writer

FORMAT NOTES:

This format is not supported by FME Base Edition.

The Canadian Council on Geomatics Interchange Format (CCOGIF) ASCII reader and writer module provides the Feature Manipulation Engine (FME) with access to the contents of a CCOGIF dataset stored in ASCII format in a single disk file. The structure of this file is discussed in Canadian Council on Geomatics' document, *Standard File Exchange Format for Digital Spatial Data* version #2.3, published October 1994.

The CCOGIF format, a data exchange format, provides a very general medium in which to represent a data model. FME accesses the individual records of a CCOGIF file at a very low level, involving only minimal interpretation of the contents of those records. This allows FME to handle virtually any data encoded with the CCOGIF standard, but requires a somewhat more sophisticated mapping file to make full use of the data.

Overview

The CCOGIF disk file consists of a series of logical records. Each of these records either describes *metadata* which is information about the data contents or structuring, or *entity data* which are geometric features.

The CCOGIF file describes a single data volume, that groups spatial data into *datasets*, *data groups*, and *data themes*. A CCOGIF volume contains one or more datasets. A single CCOGIF dataset contains one or more data groups, and a single data group contains one or more data themes.

At the highest level of grouping, the CCOGIF dataset – not to be confused with FME's concept of a dataset which is referred to as an FME dataset for the remainder of this chapter – groups the entity data by geographic region, such as a map sheet. In other words, all geographic data contained in a single CCOGIF dataset are somehow geographically related. All entity data within a CCOGIF dataset are measured in a single coordinate system.

Each data group provides some conceptual grouping of geographic entities. The criteria of this grouping are entirely data-dependent and are not constrained by the CCOGIF standard. This grouping is somewhat analogous to FME's notion of a feature type. For example, a CCOGIF dataset might contain the data groups *Highway*, *Bridge*, and *Intersection*.

The data within a single CCOGIF data group is divided into data themes. Each data theme represents a certain entity type: point, line, or area. The definition of a theme includes a list of data attributes. All attributes are defined on every entity record within the theme. A single data group may contain more than one theme of a given type – for example, two point themes. The themes are always ordered so that point themes come first, then line themes, and finally area themes.

CCOGIF ASCII Quick Facts

Format Type Identifier	CCOGIF
Reader/Writer	Both
Licensing Level	Professional
Dependencies	Writer requires extra-cost plug-in from Safe Software
Dataset Type	File
Feature Type	Group base name
Typical File Extensions	.asc
Automated Translation Support	Yes for Reader No for Writer
User-Defined Attributes	Yes
Coordinate System Support	No
Generic Color Support	Never
Spatial Index	Yes
Schema Required	No
Transaction Support	Yes
Geometry Type	ccogif_entity_type

Geometry Support			
Geometry	Supported?	Geometry	Supported?
aggregate	no	point	yes
circles	no	polygon	yes
circular arc	no	raster	no
donut polygon	yes	solid	no
elliptical arc	no	surface	no
ellipses	no	text	no
line	yes	z values	no
none	no		

Reader Overview

For the most part, the CCOGIF reader simply returns a feature to represent each record it encounters in the CCOGIF file. The reader does not have any requirement for definition statements.

The feature type of a feature returned from the CCOGIF reader depends on whether the feature represents metadata or entity data. Features that represent metadata records are returned with a feature type of `CCOGIF_METADATA`, whereas features that represent entity records are returned with a feature type dependent on the CCOGIF data group and theme within that data group. The feature type will have the format `<GroupName>_<ThemeIndex>`, where `<GroupName>` is the name of the group extracted from the Data Group Header Record (DGHR), and `<ThemeIndex>` is the position of the data theme within the group.

There are different ways to generate mapping files to read CCOGIF data. The generated mapping files run the features through a number of factories, so the actual names of the feature types used in an automatically generated mapping file will depend on which method is used and may not correspond to the feature types returned from the reader itself. The different methods are discussed later in this section, under the heading *Generated Mapping Files*.

Reader Directives

The suffixes shown are prefixed by the current `<ReaderKeyword>` in a mapping file. By default, the `<ReaderKeyword>` for the CCOGIF reader is `CCOGIF`.

DATASET

Required/Optional: *Required*

The value for this keyword is the name of the file containing the CCOGIF volume to be read. A typical mapping file fragment specifying an input CCOGIF volume looks like this:

```
CCOGIF_DATASET /usr/data/ntdb/021g01.asc
```

Note: Notice that this refers to the CCOGIF volume and not the CCOGIF dataset. There may be several datasets in a single CCOGIF volume.

Workbench Parameter: `<WorkbenchParameter>`

Writer Overview

The CCOGIF writer provides the ability to write FME feature data to a single CCOGIF file, the name of which is specified by the `DATASET` keyword. The contents of this file forms a single CCOGIF volume, consisting of exactly one CCOGIF dataset.

Metadata records may be inserted into the output data stream (for example, with the `CreationFactory`) to define the precise contents of all metadata records in the output file. This technique is described later in this chapter, under the heading *Defining Volume Structure*.

Unlike the CCOGIF reader, the writer requires `DEF` lines to define the attributes of the output CCOGIF file. The writer provides a mechanism in the `DEF` lines to precisely specify the attributes and order of every theme within each data group being written.

FME ships with sample mapping files that output CCOGIF from a `NULL` data source, as well as from Oracle. These are found in the `gallery/ccogif` subdirectory of the FME installation.

Writer Directives

The following directives are processed by the CCOGIF writer.

- `DATASET`
- `DEF`
- `AREA_RELPOSN_ATTR`

The suffixes shown are prefixed by the current <WriterKeyword> in a mapping file. By default, the <WriterKeyword> for the CCOGIF writer is CCOGIF.

DATASET

The file name of the output CCOGIF data file.

DEF

The CCOGIF DEF line is required to specify the contents of a CCOGIF data theme before any geometric entities may be written to that theme. All entities in a given data theme have the same geometric entity type – point, line, or area – and have the same set of attributes defined on them. The DEF line for the theme provides this information.

The syntax of a CCOGIF DEF line is:

```
<WriterKeyword>_DEF <themeName> \
  [CCOGIF_GROUP_NAME <groupName>] \
  [CCOGIF_THEME_ENTITY_TYPE <entityType>] \
  [CCOGIF_THEME_ORDERING <orderIndex>] \
  [<attrName> <attrType>]+
```

The <themeName> is simply the identifier used within the mapping file to refer to the theme. Data themes do not have identifiers within the CCOGIF file, so the chosen <themeName> is not actually reflected in the CCOGIF file.

The CCOGIF_GROUP_NAME keyword specifies the name of the group containing the data theme. The value <groupName> is placed into the ccogif_data_group_name attribute for the theme's group's header record. All themes given a common <groupName> value belong to the same group.

In general, the DEF lines require each theme to be explicitly specified. Exceptions to this are noted later in this section, under the heading *Defining Volume Structure*.

The CCOGIF_THEME_ENTITY_TYPE keyword specifies the geometric type of the entity records to be written to the theme and is required in most cases. The value of <entityType> must be one of the values ccogif_point, ccogif_line, or ccogif_area.

The optional CCOGIF_THEME_ORDERING keyword allows each theme to be assigned a numeric ordering value. When the themes are written out to the CCOGIF file, they are ordered so that:

- All point themes belonging to a given data group are written first, followed by line themes, and finally area themes.
- All of a group's themes of a given entity type – line, point, or area – are written with a numerically increasing <orderIndex> value.

Themes for which no ordering index was specified are written with an arbitrary relative ordering after all themes of the same entity type for which a theme ordering was specified.

All attribute names must contain no more than 40 characters. They may be composed of nearly any printable characters including alphanumeric, colons, periods, commas,

apostrophes, and accented characters. The following table shows the attribute types that are supported:

Attribute Name	Description
INT	Integers are represented with 16 ASCII characters. They are stored as base 10 numbers, right-justified in the field, with leading zeroes to fill the remaining space. The first character denotes the sign of the integer and is either + if positive or - if negative.
REAL	Real numbers are stored in base 10 exponential form as 16 ASCII characters. The format for the real number is: ±d.ddddddddE±dd.
DMS	Degree Minute Second (DMS) fields are used to store angular values in terms of degrees, minutes, and seconds. The degrees and seconds are represented with base 10 integers, and the seconds value is represented with a fixed point number with five digits of precision. The format of a DMS value is ±ddd mm ss.sssss. The integer part of each of the three numbers – degrees, minutes, seconds – is padded on the left with zeroes to fill its allotted space.
CHAR(<width>)	Character fields are stored as fixed-length strings. Their values are padded on the right with spaces to fill the allotted space.
DATE	Date fields are stored as 8-character strings, with the format YYYYMM-MDD.

AREA_RELPOSN_ATTR

Some data encoded in CCOGIF requires each line which defines the boundary or a hole of an area to include an attribute specifying whether it is a part of a boundary or a specific hole. (For example, NTDB v3.1 uses the "ATE" attribute for this purpose.)

All line entities which form the outer boundary of an area will contain a value of 0 for this attribute. Those line entities which form the first hole will have a value of 1; those which form the second hole will have a value of 2; and so on. All other entities will have a value of -1 in this attribute.

AREA_RELPOSN_ATTR names the attribute which is to hold this information. If it is not specified, this information will not be stored on the entities. Otherwise, the named attribute will contain the information. (This attribute must be defined as a numeric attribute in the data theme's DEF line for the information to actually appear in the output CCOGIF file.)

Feature Representation

In addition to the generic FME feature attributes that FME Workbench adds to all features (see *About Feature Attributes* on page 7), this format adds the format-specific attributes described in this section.

A CCOGIF feature can represent either a metadata feature or an entity feature. Metadata features describe the structure of the CCOGIF data whereas entity features form the geometry of the volume.

Metadata Features

Each metadata feature describes the contents of one of the following records from the CCOGIF file:

Record Name	Description
VDR	Volume Descriptor Record: contains information about the entire CCOGIF volume, such as its creation date and generating agency.
UFLR	User Fixed Length Record: contains any user data associated with the CCOGIF volume or dataset. It simply contains free-form ASCII data.
DSHR	Data Set Header Record: describes each dataset contained in a CCOGIF volume. It contains some extraneous information about the data (name, creation date, geographic location, etc.), as well as information pertaining to the interpretation of the data itself (X, Y, and Z data types, topology information, map projection information, etc.).
EMDR	Entity Metadata Record: describes the source and quality of the entity data contained in a dataset. There are one or more of these records following each DSHR record.
DGHR	Data Group Header Record: is the first record of each data group within a CCOGIF dataset. It provides a name for the group, as well as provides counts of how many point, line, and area themes are contained in the group.
DTHR	Data Theme Header Record: describes the contents of a single data theme. It provides information about the entity data itself (entity type, number of entities), as well as a count of user attributes defined on each entity and a calculation of the length of the fixed-length part of each entity in the theme.
ADR	Attribute Descriptor Record: There is one attribute descriptor record in each data theme. It is a variable-length record that describes the names and types of attributes defined on each entity contained in the theme.
EOVR	End Of Volume Record: marks the logical and physical end of the CCOGIF volume. It is always the last record in a CCOGIF file.

All metadata features have a feature type of `CCOGIF_METADATA`. The CCOGIF record described by a metadata feature is reflected by the value of the attribute `ccogif_record_code`. This attribute has one of three or four character record names listed in the above table.

Each metadata feature has a particular set of attributes, depending on the CCOGIF record it represents. The following sections list the attributes for each record type.

Volume Descriptor Record

The Volume Descriptor Record (VDR) is the first record in a CCOGIF file. The FME represents this record as a CCOGIF_METADATA feature with the following attributes:

Attribute Name	Description	Type
ccogif_record_code	Record code (constant VDR).	char(4)
ccogif_log_vol_id	Logical volume identifier.	int(16)
ccogif_phys_vol_num	Physical volume number in this logical volume (numbered sequentially from 1).	int(16)
ccogif_vol_cre_date	Volume creation date.	date
ccogif_vol_data_desc	Logical volume data description.	char(128)
ccogif_vol_gen_cntry	Volume generating country.	char(64)
ccogif_vol_gen_agncy	Volume generating agency.	char(64)
ccogif_vol_gen_fclty	Volume generating facility.	char(64)
ccogif_fmt_ctrl_doc_id	Format control document identifier.	char(64)
ccogif_sw_release_id	Software release identifier.	char(64)
ccogif_feat_code_rev	Feature code revision level.	char(64)
ccogif_num_ufl_recs	Number of user fixed length records immediately following this volume descriptor record.	int(16)
ccogif_bytes_prev_rec	Number of bytes left from last logical record of previous physical volume.	int(16)

User Fixed-Length Record

User Fixed-Length Records (UFLRs) provide a place for the user's software to place any ASCII information for its own purpose. Each UFLR contains up to 2044 bytes of user-defined data. There are zero or more UFLRs immediately following each VDR or DSHR in the CCOGIF file. The ccogif_num_ufl_recs attribute of the VDR or DSHR feature tell us how many UFLR records to expect.

A single FME feature is used to represent a sequence of user fixed length records. The data from the entire sequence of UFLRs appears concatenated together in a single attribute on the single FME feature.

FME represents the sequence of UFLRs with a CCOGIF_METADATA feature with the following attributes:

Attribute Name	Description	Type
ccogif_record_code	Record code, constant UFLR.	char(4)
ccogif_user_def_data	User-defined data, concatenated from an entire sequence of UFLR records.	char(n)

Data Set Header Record

The Data Set Header Record (DSHR) defines all information common to all entities contained in a single CCOGIF dataset. A CCOGIF volume may contain more than one dataset.

FME represents a DSHR record with a `CCOGIF_METADATA` feature with the following attributes:

Attribute Name	Description	Type
<code>ccogif_record_code</code>	Record code (constant DSHR).	<code>char(4)</code>
<code>ccogif_data_set_name</code>	Dataset name.	<code>char(64)</code>
<code>ccogif_ds_cre_date</code>	Dataset creation date.	<code>date</code>
<code>ccogif_ds_loc_text</code>	Dataset geographic location text.	<code>char(64)</code>
<code>ccogif_related_ds</code>	Reference to other related datasets.	<code>char(64)</code>
<code>ccogif_data_three_dim</code>	Specifies whether data is three-dimensional (3D). If false, Z coordinate is always zero. Legal values are T (true), F (false) and U (unknown).	<code>char(1)</code>
<code>ccogif_pt_to_ln_topo</code>	Specifies whether point-to-line topology exists in a dataset. Legal values are T (true), F (false) and U (unknown).	<code>char(1)</code>
<code>ccogif_ln_to_pt_topo</code>	Specifies whether line-to-point topology exists in a dataset. Legal values are T (true), F (false) and U (unknown).	<code>char(1)</code>
<code>ccogif_colloc_exists</code>	Specifies whether dataset employs line collocation. Legal values are T (true), F (false) and U (unknown).	<code>char(1)</code>
<code>ccogif_ln_to_area_topo</code>	Specifies whether line-to-area topology exists in a dataset. Legal values are T (true), F (false) and U (unknown).	<code>char(1)</code>
<code>ccogif_area_to_ln_topo</code>	Specifies whether area-to-line topology exists in a dataset. Legal values are T (true), F (false) and U (unknown).	<code>char(1)</code>
<code>ccogif_known_pt_in_area</code>	Specifies whether there is a known point in each area. Legal values are T (true), F (false) and U (unknown).	<code>char(1)</code>
<code>ccogif_attrs_in_entity</code>	Specifies whether attributes are present in entity records. Legal values are T (true), F (false) and U (unknown).	<code>char(1)</code>

Attribute Name	Description	Type
ccogif_feat_classes	Ordered list of feature classes (A to K) for the dataset. Blanks are placed for classes not present; e.g., A, D, GH, J.	char(32)
ccogif_num_data_grp	Number of data groups contained in this CCOGIF dataset (n>=1).	int(16)
ccogif_num_ufl_recs	Number of user fixed length records immediately following this dataset header record (n>=0).	int(16)
ccogif_num_emd_recs	Number of entity metadata records that describe the contents of this CCOGIF dataset (n>=1).	int(16)
ccogif_x_data_type, ccogif_y_data_type, ccogif_z_data_type	Data type of x, y, or z coordinate values. Legal values are INT, REAL or DMS.	char(4)
ccogif_x_data_units, ccogif_y_data_units, ccogif_z_data_units	Units in which x, y, or z coordinate values are measured – for example, METRES, METRES ASL.	char(16)
ccogif_z_min_value, ccogif_z_max_value	Z coordinate minimum and maximum values. Interpretation of this attribute depends on the values of ccogif_z_data_type and ccogif_z_data_units.	variable ^a
ccogif_proj_id	Map projection identifier that describes the map projection in which the entity data in the dataset is encoded. See the discussion below this table for more details.	char(4)
ccogif_geod_datum	Name of geodetic datum.	char(16)
ccogif_adj_name	Name of adjustment.	char(16)
ccogif_vert_datum	Name of vertical datum.	char(16)

a. Fields marked with a type of variable are either REAL, INTEGER, or DMS depending on the DSHRs x, y, or z data type corresponding to that field.

The `ccogif_x_data_type`, `ccogif_y_data_type`, and `ccogif_z_data_type` tell what numeric format is used to represent x, y, and z coordinate values. `INTEGER` and `REAL` are obvious representations. `DMS` values for x and y coordinate values only are stored as `+ddd mm ss.sss`, where `ddd` is the degrees portion of the number, `mm` is the number of minutes, and `ss.sss` is the number of seconds. `DMS` values are converted by FME to their corresponding numeric or decimal values.

The `DSHR` feature also contains a number of attributes specific to the map projection in which the dataset's entities are expressed. The selection of map projection is made by the `ccogif_proj_id` attribute. It has one of the following values:

Map Projection ID	Projection Name
0100	Latitude/Longitude

Map Projection ID	Projection Name
0200	Transverse Mercator—Universal Transverse Mercator (UTM) projections are stored with this ID as well
0203	Mercator
0300	Lambert Conformal
0400	Stereographic
0500	Polyconic

The attributes for each map projection type are listed in the following sections.

Latitude/Longitude Project Parameters

Datasets in the Latitude/Longitude projection have the following attributes defined on their DSHR feature:

Attribute Name	Description	Type
ccogif_proj_id	Map projection identifier, constant 0100.	char(4)
ccogif_proj_name	Map projection name, constant LATITUDE/LONGITUDE.	char(32)
ccogif_proj_origin_x, ccogif_proj_origin_y	Longitude/latitude origin for x,y coordinates.	DMS
ccogif_proj_num_bnd_crd	Number of coordinate pairs that form a bounding polygon for this dataset (0=>n=>12).	int(16)
ccogif_proj_bnd_crd{n}.x, ccogif_proj_bnd_crd{n}.y	Coordinate #n of bounding polygon (0=>n>config_num_bnd_crd).	DMS

Transverse Mercator Projection Parameters

Datasets in the Transverse Mercator projection have the following attributes defined on their DSHR feature:

Attribute Name	Description	Type
ccogif_proj_id	Map projection identifier, constant 0200.	char(4)
ccogif_proj_name	Map projection name, constant TRANSVERSE MERCATOR.	char(32)
ccogif_proj_cent_merid	Central meridian.	DMS
ccogif_proj_zone_width	Zone width.	DMS
ccogif_proj_sphd_name	Spheroid name.	char(20)
ccogif_proj_semi_major	Semi-major axis.	real(16)
ccogif_proj_semi_minor	Semi-minor axis.	real(16)

Attribute Name	Description	Type
ccogif_proj_eccent	Eccentricity.	real(16)
ccogif_proj_scl_fact	Scale factor.	real(16)
ccogif_proj_false_east, ccogif_proj_fals_north	False Easting/Northing.	real(16)
ccogif_proj_zone	Zone number.	int(16)
ccogif_proj_orig_east, ccogif_proj_orig_north	Origin (easting, northing).	variable ^a
ccogif_proj_num_bnd_crd	Number of coordinate pairs that form a bounding polygon for this dataset (0>=n>=12).	int(16)
ccogif_proj_bnd_crd{n}.x, ccogif_proj_bnd_crd{n}.y	Coordinate #n of bounding polygon (0>=n>config_num_bnd_crd).	variable ^a

a. Fields marked with a type of variable are either REAL, INTEGER, or DMS depending on the DSHR's x, y, or z data type corresponding to that field.

Mercator Projection Parameters

Datasets in the Mercator projection have the following attributes defined on their DSHR feature:

Attribute Name	Description	Type
ccogif_proj_id	Map projection identifier, constant 0203.	char(4)
ccogif_proj_name	Map projection name, constant MERCA-TOR.	char(32)
ccogif_proj_mid_lat	Mid latitude.	DMS
ccogif_proj_sphd_name	Spheroid name.	char(20)
ccogif_proj_semi_major	Semi-major axis.	real(16)
ccogif_proj_semi_minor	Semi-minor axis.	real(16)
ccogif_proj_eccent	Eccentricity.	real(16)
ccogif_proj_orig_east, ccogif_proj_orig_north	Origin (easting, northing).	variable ^a
ccogif_proj_num_bnd_crd	Number of coordinate pairs that form a bounding polygon for this dataset (0>=n>=12).	int(16)
ccogif_proj_bnd_crd{n}.x, ccogif_proj_bnd_crd{n}.y	Coordinate #n of bounding polygon (0>=n>config_num_bnd_crd).	variable ^a

a. Fields marked with a type of variable are either REAL, INTEGER, or DMS depending on the DSHR's x, y, or z data type corresponding to that field.

Lambert Conformal Projection Parameters

Datasets in the Mercator projection have the following attributes defined on their DSHR feature:

Attribute Name	Description	Type
ccogif_proj_id	Map projection identifier, constant 0300.	char(4)
ccogif_proj_name	Map projection name, constant LAMBERT CONFORMAL.	char(32)
ccogif_proj_frst_scl_par	First scaling parallel.	DMS
ccogif_proj_secnd_scl_par	Second scaling parallel.	DMS
ccogif_proj_sphd_name	Spheroid name.	char(20)
ccogif_proj_semi_major	Semi-major axis.	real(16)
ccogif_proj_semi_minor	Semi-minor axis.	real(16)
ccogif_proj_eccent	Eccentricity.	real(16)
ccogif_proj_orig_east, ccogif_proj_orig_north	Origin (easting, northing).	variable ^a
ccogif_proj_num_bnd_crd	Number of coordinate pairs that form a bounding polygon for this dataset (0>=n>=12).	int(16)
ccogif_proj_bnd_crd{n}.x, ccogif_proj_bnd_crd{n}.y	Coordinate #n of bounding polygon (0>=n>config_num_bnd_crd).	variable ^a

a. Fields marked with a type of variable are either REAL, INTEGER, or DMS depending on the DSHR's x, y, or z data type corresponding to that field.

Stereographic Projection Parameters

Datasets in the Mercator projection will have the following attributes defined on their DSHR feature:

Attribute Name	Description	Type
ccogif_proj_id	Map projection identifier, constant 0400	char(4)
ccogif_proj_name	Map projection name, constant STEREOGRAPHIC	char(32)
ccogif_proj_scale_lat	Scaling latitude	DMS
ccogif_proj_sphd_name	Spheroid name	char(20)
ccogif_proj_semi_major	Semi-major axis	real(16)
ccogif_proj_semi_minor	Semi-minor axis	real(16)
ccogif_proj_eccent	Eccentricity	real(16)
ccogif_proj_orig_east, ccogif_proj_orig_north	Origin (easting, northing)	variable ^a

Attribute Name	Description	Type
ccogif_proj_num_bnd_crd	Number of coordinate pairs that form a bounding polygon for this dataset (0>=n>=12)	int(16)
ccogif_proj_bnd_crd{n}.x, ccogif_proj_bnd_crd{n}.y	Coordinate #n of bounding polygon (0>=n>config_num_bnd_crd)	variable ^a

- a. Fields marked with a type of variable are either REAL, INTEGER, or DMS depending on the DSHRs x, y, or z data type corresponding to that field.

Polyconic Projection Parameters

Datasets in the Mercator projection have the following attributes defined on their DSHR feature:

Attribute Name	Description	Type
ccogif_proj_id	Map projection identifier, constant 0500.	char(4)
ccogif_proj_name	Map projection name, constant POLY-CONIC.	char(32)
ccogif_proj_cent_merid	Central meridian.	DMS
ccogif_proj_sphd_name	Spheroid name.	char(20)
ccogif_proj_semi_major	Semi-major axis.	real(16)
ccogif_proj_semi_minor	Semi-minor axis.	real(16)
ccogif_proj_eccent	Eccentricity.	real(16)
ccogif_proj_orig_east, ccogif_proj_orig_north	Origin (easting, northing).	variable ^a
ccogif_proj_num_bnd_crd	Number of coordinate pairs that form a bounding polygon for this dataset (0>=n>=12).	int(16)
ccogif_proj_bnd_crd{n}.x, ccogif_proj_bnd_crd{n}.y	Coordinate #n of bounding polygon (0>=n>config_num_bnd_crd).	variable ^a

- a. Fields marked with a type of variable are either REAL, INTEGER, or DMS depending on the DSHR's x, y, or z data type corresponding to that field.

Entity Metadata Record

The Entity Metadata Record (EMDR) describes the source and quality of the data contained in the CCOGIF dataset. There may be multiple EMDRs in a single CCOGIF dataset if there are varying sources and quality of data in the dataset. The entity data records in the CCOGIF dataset refer back to these entity metadata records by ID.

The FME represents an EMDR with a CCOGIF_METADATA feature with the following attributes:

Attribute Name	Description	Type
ccogif_record_code	Record code, constant EMDR.	char(4)
ccogif_meta_data_id	Metadata ID number.	int(16)
ccogif_data_gen_agncy	Data generating agency.	char(64)
ccogif_capture_method	Method of data capture or revision.	char(64)
ccogif_col_instrmt	Type of collecting instrument.	char(64)
ccogif_src_mat_type	Type of source material.	char(64)
ccogif_src_mat_scale	Scale of source material.	char(64)
ccogif_src_mat_date	Date of source material.	date
ccogif_fld_comp_date	Date of field completion.	date
ccogif_data_captr_date	Date of data capture or revision.	date
ccogif_src_mat_spec	Reference to specification document on source material and collection or revision methods.	char(192)
ccogif_feat_code_spec	Reference to specification document on feature coding and attribute assigning procedures.	char(192)
ccogif_data_struc_spec	Reference to specification document on data structuring process.	char(192)
ccogif_qual_ctrl_spec	Reference to specification document on quality control procedures.	char(192)
ccogif_trans_gen_spec	Reference to specification document on transformations and generalization procedures.	char(192)
ccogif_field_cpltn_spec	Reference to specification document on field completion procedures.	char(192)
ccogif_acc_det_proc_spec	Reference to specification document on accuracy determination procedures.	char(192)
ccogif_data_resolution	Resolution of the data.	char(64)
ccogif_x_accuracy, ccogif_y_accuracy, ccogif_z_accuracy	X, Y, Z positional accuracy of the data.	real(16)

Data Group Header Record

Each data group within a CCOGIF dataset starts with a Data Group Header Record (DGHR). This record defines the name of the data group and tells how many point, line, and area themes are contained in the group.

The FME represents a DGHR with a CCOGIF_METADATA feature with the following attributes:

Attribute Name	Description	Type
ccogif_record_code	Record code, constant DGHR.	char(4)
ccogif_data_group_name	Name of data group.	char(64)
ccogif_num_point_themes, ccogif_num_line_themes, ccogif_num_area_themes	Number of (point, line, area) themes contained in this data group.	int(16)

Data Theme Header Record

Each theme in a data group starts with a Data Theme Header Record (DTHR). This record defines the entity type (point, line, area) contained in the theme, the number of attributes defined on each entity in the theme, and the length of the fixed length entity records within the theme.

All entities within a particular theme must be of the same entity type and must have the same set of attributes.

FME represents a DTHR with a CCOGIF_METADATA feature with the following attributes:

Attribute Name	Description	Type
ccogif_record_code	Record code, constant DTHR.	char(4)
ccogif_entity_type	Type of entity in theme (POINT, LINE, AREA).	char(8)
ccogif_num_entities	Number of entities in this data theme.	int(16)
ccogif_num_attr_desc	Number of attributes defined on each entity of the theme.	int(16)
ccogif_fixed_len_bytes	Length of the fixed length portion of each entity in the theme.	int(16)

Attribute Descriptor Record

All entities within a given data theme have the same set of attributes defined on them. The Attribute Descriptor Record (ADR) lists the name and type of each attribute defined on the entities of the current data theme. The number of attributes defined on a theme is specified in the ccogif_num_attr_desc attribute of the DTHR.

The FME represents an ADR with a CCOGIF_METADATA feature that has the following attributes:

Attribute Name	Description	Type
ccogif_record_code	Record code, constant ADR.	char(4)
ccogif_attr{n}_name	Name of attribute #n.	char(40)

Attribute Name	Description	Type
ccogif_attr{n}_type	Type of attribute #n (INT, REAL, DMS, CHAR, or DATA).	char(4)
ccogif_attr{n}_len	String length if attribute #n is a CHAR type, otherwise 0.	int(16)

Entity Features

The entity features are the features that represent the entities of the CCOGIF dataset. When reading CCOGIF data, an entity feature will have a feature type of `<data_group>_<theme_index>`, where:

`<data_group>` is the name of the data group, taken from the most recent DGHR meta-data record. All special characters—such as spaces, colons, etc.—are replaced with underscores.

`<theme_index>` is the index within the data group of the theme containing the entity. The themes are numbered sequentially within their data group. The first theme in each data group is number 1.

Entities may be described in a CCOGIF file as a pair of records. Each entity has a fixed length record and, optionally, a variable-length record. The fixed length portion contains information, such as an ID number, a point's location, a line's topological information, a feature code, and attribute values. In other words, the data present for all entities in the theme.

The variable-length portion contains the data that varies in size between entities within a given theme and may or may not be present for a given entity. It contains, for example, a list of lines attached to a point entity, the coordinates defining a line, or the list of identifiers of lines defining an area's boundaries.

FME's entity features combine the contents of the fixed length and variable length records for a particular entity into a single FME feature.

All entity features have the following attributes defined on them:

Attribute Name	Description	Type
ccogif_record_code	Record code, constant PFLR for point entities, LFLR for line entities, AFLR for area entities.	char(4)
ccogif_data_coll_md_ptr	Data collection metadata pointer (reference number of EMDR corresponding to data collection).	int(16)
ccogif_data_rev_md_ptr	Data collection metadata pointer (reference of EMDR corresponding to data revision or validation).	int(16)
ccogif_prim_feat_code	Primary feature code for the entity.	char(12)
ccogif_data_group_name	Name of the data group containing the entity.	char(40)

Attribute Name	Description	Type
ccogif_data_theme_id	Index of the data theme within the data group.	int(16)

Each specific type of entity has additional attributes to describe the entity and are listed in the sections that follow.

In addition to the standard attributes, each entity feature also has values for all attributes listed in its data theme's `ADR`.

Point Entity Features

Point entities are represented in the CCOGIF file as a Point Fixed-Length Record (PFLR) and, optionally, a Point Variable-Length Record (PVLRL). The FME combines the contents of these two records into a single feature – the point entity feature.

The geometry of FME's point entity feature is the point's coordinates from the `PFLR`. A Z-value of `-9999` represents an undefined value, so any `PFLRL`s that have a Z-value of `-9999` are translated as an (x,y) coordinate instead of an (x,y,z) coordinate.

In addition to the geometry and the attributes common to all entity features (listed in the previous section), point entity features have the following attributes defined:

Attribute Name	Description	Type
ccogif_record_code	Record code, constant PFLR.	char(4)
ccogif_point_id	Point ID number.	int(16)
ccogif_num_lines	Number of lines attached to this point (n>=0).	int(16)
ccogif_orientation	Orientation of point, measured in degrees counterclockwise from the x-axis.	real(16)
ccogif_line_id{n}	Line identifier of nth line attached to this point.	int(16)

Line Entity Features

Line entities are represented in the CCOGIF file as a Line Fixed-Length Record (LFLR) and optionally, a Line Variable-Length Record (LVLRL). The FME combines the contents of these two records into a single feature—the line entity feature.

The geometry of FME's line entity feature is the line's coordinates from the `LVLRL`. A Z-value of `-9999` represents an undefined value, so any `LVLRL`s that have Z-values of `-9999` are translated as (x,y) coordinates instead of (x,y,z) coordinates.

In addition to the geometry and attributes common to all entity features, which were listed in the previous section, line entity features have the following attributes defined:

Attribute Name	Description	Type
ccogif_record_code	Record code, constant LFLR.	char(4)

Attribute Name	Description	Type
ccogif_line_id	Line ID number.	int(16)
ccogif_num_lines	Number of lines attached to this point (n>=0).	int(16)
ccogif_coll_line_id	ID number of collocated line (0 if not collocated).	int(16)
ccogif_start_node_id	Start node ID number (0 if not defined).	int(16)
ccogif_end_node_id	End node ID number (0 if not defined).	int(16)
ccogif_left_area_id	Left area ID number (0 if not defined).	int(16)
ccogif_right_area_id	Right area ID number (0 if not defined).	int(16)

Area Entity Features

Area entities are represented in the CCOGIF file as an Area Fixed-Length Record (AFLR) and optionally, as an Area Variable-Length Record (AVLR). The FME combines the contents of these two records into a single feature – the area entity feature.

The geometry of FME's area entity feature is the coordinates of a point inside the area, as defined in the AFLR. A Z-value of -9999 represents an undefined values so any AFLRS that have a Z-value of -9999 are translated as an (x,y) coordinate instead of an (x,y,z) coordinate. Note that the area entity feature itself does not contain any polygonal geometry. Instead, it contains a list of attributes pointing to the identifiers of the lines that make up the boundary of the area. To form a polygon from CCOGIF data it is necessary to use the *ReferenceFactory*, or a similar factory, in the mapping file to associate the area feature with its polygonal boundaries.

In addition to the geometry and the attributes common to all entity features (listed in the previous section), area entity features have the following attributes defined:

Attribute Name	Description	Type
ccogif_record_code	Record code, constant AFLR.	char(4)
ccogif_area_id	Area ID number.	int(16)
ccogif_num_bnd_lines	Number of lines that form the boundaries of this polygon (n>=0).	int(16)
ccogif_line_id{n}	Line identifier of nth boundary line for this area.	int(16)

Defining Volume Structure

The contents of a CCOGIF volume follow a firm structure, provided through the use of metadata records. The sequence and contents of these metadata records is crucial to the correctness of a CCOGIF volume. The CCOGIF writer provides a mechanism for the mapping file to explicitly define the required records and their contents.

There are six areas where the CCOGIF writer provides direct control of the generated CCOGIF records:

- Definition and order of data themes within a data group
- Specification of metadata records, such as:
 - the contents of Volume Descriptor Record (VDR)
 - the contents of User Fixed-Length Record (UFLR)
 - the contents of Data Set Header Record (DSHR)
 - the contents of Entity Metadata Record (EMDR)
- Specification of the contents of entity records

The following sections cover each of these in more detail.

Theme Definition

Geometric entity records are grouped into themes, where all entities within a given theme have the same geometric type – point, line, or area – and the same set of attributes. Each theme written to a CCOGIF file is defined by a CCOGIF `DEF` line. The `DEF` line specifies the name of the group to which the theme belongs, and allows specification of the relative ordering of the themes within the groups. The header records for data groups (DGHR) are written to the output file for each group mentioned on a `DEF` line.

The group name and theme ordering index may be specified explicitly on the `DEF` line with the `CCOGIF_GROUP_NAME` and `CCOGIF_THEME_ORDERING` keywords, or they may be implied by the theme identifier. If the `DEF` line does not have a `CCOGIF_GROUP_NAME`, the theme identifier, or FME feature type, becomes the implied group name. In this case, the `DEF` line actually defines attributes for the group itself and not a particular theme, and therefore must not include any theme ordering or entity type information. The reasons and implications of assigning attributes to a group instead of a theme within the group are discussed below.

If the `DEF` line contains neither an explicit group name or an explicit theme ordering, and the theme identifier is of the form `<groupName>_<number>`, where `<number>` is any integer, then the group name and theme ordering are implied as `<groupName>` and `<number>` respectively.

The geometric entity type for each theme must be provided on the theme's `DEF` line by using the `CCOGIF_THEME_ENTITY_TYPE` keyword. However, it is not necessary for every `DEF` line to have an entity type provided. If neither entity type nor theme ordering information is specified, then the `DEF` line is considered to define a set of attributes for a data group rather than for a data theme.

Strictly speaking, it makes no sense to talk of attributes being assigned to a CCOGIF data group as attributes are assigned to themes within the CCOGIF file. This mechanism, however, provides the ability to essentially define a point, line, and area theme within a single group, with identical sets of attributes. Any features with the specified feature type are written to the appropriate theme—point, line, or area—depending on the geometry type of the feature. Features with aggregate geometry or no geometry at all will not be written.

The themes implied with this mechanism, called generic themes from this point on, may coexist with other themes in a group, making it possible to define a number of themes for a data group as well as to define generic themes for “the other stuff that we want to write but that doesn't fit into our predefined themes”. If the generic theme

mechanism is used, its `DEF` line must appear in the mapping file before the `DEF` lines for any non-generic themes within that group.

Specifying Metadata Records

The metadata records in a CCOGIF file contain many pieces of information that must be correctly defined for the file to be accurate. Much of this information is for the benefit of human users of the end product and may vary not only from site to site, but from one dataset to another. This information cannot be coded into FME itself, therefore it must be supplied into the mapping file.

The CCOGIF writer expects this information to be passed in the same metadata features that the CCOGIF reader creates. If the writer receives a feature with a feature type of `CCOGIF_METADATA`, it will look at the `ccogif_record_code` attribute to see which of the metadata features, described in *Metadata Features* on page 332, the feature represents. It extracts from this feature whatever `ccogif_XXX` attributes apply to that particular type of metadata feature and eventually writes the information to the metadata records in the output CCOGIF file.

The order of the incoming metadata features is significant, as they are written out in a similar order to which they are received. Any volume-specific metadata **must** come before the Data Set Header Record (DSHR). Metadata features received after the DSHR appear in the output CCOGIF file as a part of the dataset.

In addition, all metadata features must be presented to the writer before any entity features are presented. The CCOGIF writer needs information from the DSHR in order to write entity data, so it creates a default DSHR if none has been given. Any metadata that comes after the default DSHR has been generated may contradict the default values placed into the DSHR, resulting in an invalid output CCOGIF file.

Several ways to generate the metadata features for the CCOGIF writer are discussed below:

- Use the Multi-Reader as the input reader and use a template CCOGIF file as a source for `CCOGIF_METADATA` features. In other words, specific metadata features may be chosen from this reader, then redirected to the writer to provide attribute values for the corresponding metadata records in the output.
- Store the template's `CCOGIF_METADATA` features in a feature store and use the `RecordingFactory` to inject them into the feature stream. If this method is chosen, it's important to choose the playback mode of `PLAYBACK` instead of `PLAYBACK_AT_END`.
- Use the `CreationFactory` to create the metadata features with all of the required attributes.

Once the features have been generated, they have to be handed to the writer with a feature type of `CCOGIF_METADATA`. This can be accomplished by creating a false theme in any of the groups in the output file such as:

```
CCOGIF_DEF CCOGIF_METADATA                                     \  
    CCOGIF_GROUP_NAME           "FEATURES"
```

With this definition in place, you can correlate the metadata features to the `CCOGIF_METADATA` feature type of the output format. This correlation must equal all

ccogif_XXX attributes on the source and target sides for all metadata feature types being correlated. For example:

```
CCOGIF CCOGIF_METADATA \
  ccogif_adj_name          %ccogif_adj_name\
  ccogif_area_to_ln_topo   %ccogif_area_to_ln_topo\
  ccogif_attrs_in_entity   %ccogif_attrs_in_entity\
  ccogif_bytes_prev_rec    %ccogif_bytes_prev_rec\
  ccogif_colloc_exists     %ccogif_colloc_exists \
  ...
  ccogif_z_min_value       %ccogif_z_min_value

SHAPE CCOGIF_METADATA \
  ccogif_adj_name          %ccogif_adj_name\
  ccogif_area_to_ln_topo   %ccogif_area_to_ln_topo\
  ccogif_attrs_in_entity   %ccogif_attrs_in_entity\
  ccogif_bytes_prev_rec    %ccogif_bytes_prev_rec \
  ccogif_colloc_exists     %ccogif_colloc_exists\
  ...
  ccogif_z_min_value       %ccogif_z_min_value
```

The following sections describe ways in which some of the types of metadata features are treated specially by the CCOGIF writer. This special treatment simply ensures that the record exists and has some legal, if not meaningful, default values in place for the ccogif_XXX attributes.

Volume Descriptor Record Contents

The Volume Descriptor Record (VDR) must be present in every CCOGIF file. If it is not given to the CCOGIF writer, it will be created with the default attribute values listed in the following table. If a VDR metadata feature is given to the CCOGIF writer and it defines only some of the ccogif_XXX attributes that appear in the table below, the default value will be “taken” for those not specified.

Attribute Name	Contents – Default Values
ccogif_log_vol_id	FME-generated CCOGIF dataset
ccogif_phys_vol_num	0
ccogif_vol_cre_date	Current date
ccogif_vol_data_desc	Empty string (“ ”)
ccogif_vol_gen_centry	Empty string (“ ”)
ccogif_vol_gen_agncy	Empty string (“ ”)
ccogif_vol_gen_fcilty	Empty string (“ ”)
ccogif_fmt_ctrl_doc_id	Empty string (“ ”)
ccogif_sw_release_id	Empty string (“ ”)
ccogif_feat_code_rev	Empty string (“ ”)

Attribute Name	Contents – Default Values
<code>ccogif_num_ufl_recs</code>	Computed from the amount of user data specified in the volume's UFLR metadata features
<code>ccogif_bytes_prev_rec</code>	0

The value for the attribute `ccogif_num_ufl_recs` is always filled in automatically by the CCOGIF writer. Its value will be based on the length of the user data passed to the writer via the User Fixed-Length Records' (UFLR) metadata features.

User Fixed-Length Record Contents

Each VDR and DSHR record in a CCOGIF file may be immediately followed by zero or more UFLRS. When given a sequence of UFLR metadata features, the CCOGIF writer will read the `ccogif_user_def_data` attribute of each feature and concatenate their values into one large character string. When it comes time to write out the metadata records, the writer creates as many UFLRS as are required to hold the accumulated data. Each UFLR can contain up to 2044 bytes of user data.

The placement of the UFLRS in the output CCOGIF file depends on where they occur in the sequence of metadata features. If the UFLR appears before the DSHR feature, or the first entity feature, if no DSHR feature is explicitly given, they will be written out immediately following the volume's VDR. In this case, the VDR's `ccogif_num_ufl_recs` are set to specify how many UFLRS follow.

If the UFLR metadata features appear after the DSHR metadata feature, then the UFLRS will be written immediately following the DSHR record and the DSHR's `ccogif_num_ufl_recs` will be set to specify how many UFLRS follow.

Data Set Header Record Contents

Every CCOGIF file may contain one or more datasets.¹ The first record of each dataset in a CCOGIF volume is called the Data Set Header Record (DSHR). It provides information particular to the dataset not only for the human user—it also directs computer applications on how to process the data.

The portions of the DSHR that relate to the processing of the data are of specific interest to the CCOGIF writer. When it writes out data within a dataset, it must remain consistent with the dataset characteristics set out in the DSHR. The following information is particularly interesting to the writing process.

- **Coordinate data type:** The DSHR specifies the data type, `INT`, `REAL` or `DMS`, for each of the `x`, `y`, and `z` coordinate values within the dataset. The CCOGIF writer uses the values of the DSHR metadata feature's `ccogif_x_data_type`, `ccogif_y_data_type`, and `ccogif_z_data_type` attributes to format the numerical coordinates correctly.
- **Data set content indicator:** The Data Set Content Indicator (DSCI) is a sub-record of the DSHR that tells whether data is 3D, whether there is a known point in each area, and whether various topology information such as, point to line topology, line to point topology, line collocation, line to area topology, and area to line topology, is present in the entity attributes. FME's CCOGIF writer currently does not

1. The FME's CCOGIF writer currently supports only a single dataset.

generate any topology information, however it passes along any that has been added to geometric entity features which have been given to it.

- **Coordinate system and map projection:** The correct coordinate system for the output CCOGIF dataset must be specified in a `DSHR` metadata feature, as this information is not yet tied in to FME's coordinate system manager. If no coordinate system information is provided to the CCOGIF writer, it will arbitrarily choose a Universal Transverse Mercator (UTM) zone 18 projection, which is most likely not what is wanted.

The following table lists the default values for all attributes in the `DSHR`. These default values will be written out for any attributes not mentioned in "any" `DSHR` metadata feature given to the writer.

Attribute Name	Contents – Default Value
<code>ccogif_data_set_name</code>	FME-generated CCOGIF dataset
<code>ccogif_ds_cre_date</code>	Current date
<code>ccogif_ds_loc_text</code>	Empty string (" ")
<code>ccogif_related_ds</code>	Empty string (" ")
<code>ccogif_data_three_dim</code>	Depends on whether first entity feature written is two- or three-dimensional.
<code>ccogif_pt_to_ln_topo</code>	F for False
<code>ccogif_ln_to_pt_topo</code>	F for False
<code>ccogif_colloc_exists</code>	F for False
<code>ccogif_ln_to_area_topo</code>	F for False
<code>ccogif_area_to_ln_topo</code>	T for True
<code>ccogif_known_pt_in_area</code>	T for True
<code>ccogif_attrs_in_entity</code>	T for True
<code>ccogif_feat_classes</code>	Empty string (" ")
<code>ccogif_num_data_grp</code>	Number of data groups written to this CCOGIF dataset.
<code>ccogif_num_ufl_recs</code>	Computed from the amount of user data specified on the UFLR metadata records.
<code>ccogif_num_emd_recs</code>	Number of EMDRs to write to the dataset.
<code>ccogif_x_data_type,</code> <code>ccogif_y_data_type,</code> <code>ccogif_z_data_type</code>	REAL
<code>ccogif_x_data_units,</code> <code>ccogif_y_data_units,</code> <code>ccogif_z_data_units</code>	METRES for x and y, METRES ASL for z
<code>ccogif_z_min_value,</code> <code>ccogif_z_max_value</code>	If the entity data is 3D, these are the actual minimum and maximum elevations. Otherwise, they are left blank.

Attribute Name	Contents – Default Value
ccogif_proj_id	Defaults to UTM zone 18, so the projection ID will be Transverse Mercator (0200)
ccogif_geod_datum	Name of geodetic datum
ccogif_adj_name	Name of adjustment
ccogif_vert_datum	Name of vertical datum

The `ccogif_num_ufl_recs` attribute on the `DSHR` is completely dependent on the amount of user data passed to the CCOGIF writer in `UFLR` metadata features and is always overwritten by the writer.

The FME defaults to a map projection of UTM zone 18. The following table presents the default values for the projection-related attributes for the `DSHR`.

Attribute Name	Content – Default Value
ccogif_proj_id	Constant: 0200
ccogif_proj_name	Constant: TRANSVERSE MERCATOR
ccogif_proj_cent_merid	Constant: 75
ccogif_proj_zone_width	Constant: 6
ccogif_proj_sphd_name	GRS 80
ccogif_proj_semi_major	Constant: 6.378137E+06
ccogif_proj_semi_minor	Constant: 6.35675231E+06
ccogif_proj_eccent	Constant: 6.694380070E-03
ccogif_proj_scl_fact	Constant: 0.9996
ccogif_proj_false_east, ccogif_proj_fals_north	Constant: 500000 east, 0 north
ccogif_proj_zone	Constant: 18
ccogif_proj_orig_east, ccogif_proj_orig_north	Constant: (0,0)
ccogif_proj_num_bnd_crd	Constant: 0
ccogif_proj_bnd_crd{n}.x, ccogif_proj_bnd_crd{n}.y	Empty string (" ")

It is important to note that the CCOGIF writer requires the `DSHR` information (especially the `x,y`, and `z` data types) before it starts to write entity data to the CCOGIF file.

Entity Metadata Record Contents

Each dataset requires at least one Entity Metadata Record (EMDR). The entity records may reference two or three `EMDRS`.

The CCOGIF writer requires EMDR metadata features to define meaningful contents for the EMDRs. If no EMDR metadata features are given to the writer, a single EMDR will be written to the output CCOGIF file with all attributes given default values from the table below. The default attribute values are also used to fill in any values of attributes not present in the EMDR metadata features passed in.

Attribute Name	Description
ccogif_meta_data_id	Constant: 0
ccogif_data_gen_agncy	Empty string (" ")
ccogif_capture_method	Empty string (" ")
ccogif_col_instrmt	Empty string (" ")
ccogif_src_mat_type	Empty string (" ")
ccogif_src_mat_scale	Empty string (" ")
ccogif_src_mat_date	Current date
ccogif_fld_comp_date	Current date
ccogif_data_captr_date	Current date
ccogif_src_mat_spec	Empty string (" ")
ccogif_feat_code_spec	Empty string (" ")
ccogif_data_struct_spec	Empty string (" ")
ccogif_qual_ctrl_spec	Empty string (" ")
ccogif_trans_gen_spec	Empty string (" ")
ccogif_field_cpltn_spec	Empty string (" ")
ccogif_acc_det_proc_spec	Empty string (" ")
ccogif_data_resolution	Empty string (" ")
ccogif_x_accuracy, ccogif_y_accuracy, ccogif_z_accuracy	Constant: 0

The `ccogif_data_coll_md_ptr` and `ccogif_data_rev_md_ptr` attributes on all output entity features must be given an explicit value in the mapping file to ensure they are meaningful.

Entity Record Contents

The entity records written out by the CCOGIF mirror those obtained from the reader. The `ccogif_XXX` attributes must be defined to be meaningful before the feature is written to the output file. If a feature is passed into the writer with an attribute named `ccogif_record_code`, the value of that attribute is inspected to see if it contains one of the values: `ccogif_point`, `ccogif_line`, or `ccogif_area`. If this attribute does not exist, the geometry type of the feature is used to determine which type of entity is to represent the feature.

No topology is normally created by the CCOGIF writer when writing entities to the CCOGIF file, except when polygon data is written to an area theme. In this case, a coverage is computed by intersecting the boundaries and holes of all polygon and donut features written to the group. The coordinates for the lines delineating the resulting areas are written out as LFLR records to the group's "generic" linear data theme, and the AFLR records are written to whatever theme the original polygons were directed at.

If some other topology is required, it can be defined through other means (that is, elsewhere in the mapping file or by an external tool) and presented to the writer as `ccogif_XXX` attributes on the entity features. This would require, however, that every aspect of the CCOGIF entity, including the entity ID and any other internal references, be correctly defined on the feature before it makes its way to the CCOGIF writer.

Generated Mapping Files

In CCOGIF files, geometric entities are grouped by geographic area, then further grouped according to attributes of the data itself such as, data themes with common geometric entity types and sets of attributes. This is very different from the conceptual divisions between data entities that typically must rely on the content of the primary feature code to provide notion similar to FME's feature types. The interpretation of a feature type requires knowledge of the conventions by which the data was encoded in the CCOGIF file.

Tip

Geomatics Canada's document titled "*Conversion of NTDB Data into CCOGIF Format*" provides an example of such a set of convention.

Without knowledge of the underlying conventions, it is very difficult to automatically generate a single mapping file that works with more than one input file. The definitions of the themes within the groups just isn't consistent enough.

To overcome this obstacle, FME can generate two different kinds of mapping files:

- The first is a *generic* mapping file that extracts all of the information it can from the features, then groups them into FME feature types based on the data theme and data group. When run, this mapping file provides a very simple representation of the data in the output format without regard to any specific set of conventions.
- The second type of mapping file which may be generated takes into account the representation of the National Topographic Data Base (NTDB) data in the CCOGIF file and is referred to as a *profile-specific* mapping file.

The term *profile* is used to refer to a set of file, feature, and attribute naming conventions used to store NTDB in another, that is non-CCOGIF, format. Geomatics Canada has three such profiles, each designed to embody NTDB data within the characteristics and limitations of a particular file format. The three profiles are for ASCII Ungenerate (ARCGEN), MIF/MID, and DXF. The FME provides a way to generate mapping files to write CCOGIF data in another format, generally following the conventions of any of these profiles.

Note: The target format does not have to be the same as the format for which the profile was defined.

The resulting files do not exactly conform to the profile, due to differences in data format and to the way in which mapping file generation works within FME. However, the resulting data files are generally much closer to what you would want than those that a generic mapping file would yield. Manual editing of the mapping files can, of course, bring it much closer.

The advantage of the profile-specific mapping files is that the knowledge of the conventions for storing the NTDB in the source and destination formats is stored in the mapping file. Therefore a single mapping file may be used for a whole series of mapsheets, whereas a generic mapping file would only be applicable to a single CCOGIF file.

The disadvantage to the profile-specific mapping file is that the actual generation process needs a few parameters about the input mapsheets. This requires some knowledge of the data in order to generate the mapping file. In addition, the generated mapping file must be used with NTDB data that is consistent with the parameters with which the mapping file was generated.

The following sections describe the process and application of the two kinds of mapping files in greater detail.

Generic Mapping Files

The generic mapping files are useful for a “one-off” translation of a CCOGIF file to another format. It will translate all of the geometric entities in the file, along with their attributes, and perform simple polygon construction with the area entities. This sort of translation is useful to quickly determine what kind of data was stored in the CCOGIF file or to create a starting point for a hand-coded mapping file.

When a generic mapping file is used, an FME feature is generated for each geometric entity. The features generated have feature types of `<groupName>_<themeIndex>`, where `<groupName>` is the name of the data group that contains the entity and `<themeIndex>` is the position of the entity’s data theme within all of the group’s themes. Because the nature of CCOGIF makes it unlikely that two CCOGIF files could have the same group and theme structure, a mapping file generated from the contents of a given CCOGIF file should only be used to translate that file.

Profile-Specific Mapping Files

Geomatics Canada has defined conventions for storing NTDB data in four different formats:

- CCOGIF
- ASCII Ungenerate, also known as ArcInfo Generate or ARCGEN
- MID/MIF
- DXF

We refer to each of these as a profile.

The published profiles define conventions for attribute naming, file naming, file composition—for example, organized by NTDB theme versus entity name—and rules for defining the specific attributes’ values. The FME has facilities for generating mapping files that translate CCOGIF into any FME-supported format, closely adhering to one of these three profiles:

- ARCGEN
- MID/MIF
- DXF

These profiles are addressed in greater detail following this discussion on profile-specific mapping files.

Aside from choice of profile, the generated mapping file depends on the following three parameters:

- **Choice of Language (English or French):** NTDB data encoded into CCOGIF contains both French and English group names and attribute names. A mapping file is configured to choose which language is used on the output file to name output feature types and attribute names.
- **Choice of NTDB Revision (2 or 3):** NTDB data in CCOGIF follows either the revision 2 or 3 standard. Since this information is not made available to the mapping file generation process, the user must specify it at the time of mapping file generation. A mapping file generated to process mapsheets from one revision cannot be used to process mapsheets from another revision.
- **Choice of Scale (50k or 250k):** NTDB data can contain information for a 50k or a 250k mapsheet. The user must select which scale of data a mapping file is to work with at the time of mapping file generation. A mapping file generated to process one scale of data may not be used to process a mapsheet from another scale because the set of groups and attributes differ slightly.

These parameters are supplied to FME mapping file generation process using the macros `NTDB_Language`, `NTDB_Version`, and `NTDB_Scale`. When generating a mapping file from the command line, the parameters would be specified something similar to the following command. As no CCOGIF input file is required for a profile-specific mapping file, the word `unused` is given.

```
fme generate ntdbcg shape unused mymapping.fme --NTDB_Language French --
NTDB_Version 3 --NTDB_Scale 50k
```

Once a profile-specific mapping file has been generated, it may be stored (for example, in the FME gallery) to be used later. It is not necessary to generate a profile-specific mapping file each time a translation is performed.

The following sections describe the specifics of each of the three profiles in more detail.

ASCII Ungenerate (ARCGEN) Profile

The ASCII Ungenerate profile is specified in the Geomatics Canada document titled "*Conversion of NTDB Edition 3 Data into ASCII Ungenerate Format*". This profile has the following properties:

- A separate output file is generated for each entity and geometric representation such as, point, line, area.
- File names have a maximum of eight characters. The first seven are the seven-character identifier for the theme—for example, `BATIMEN`, `BUILDIN`, `CHEMINE`, `CHIMNEY`—followed by a single character for the entity type—`P`, `L` or `A`.
- Point data is stored in a file with the extension `.pts`, lines in a file with the extension `.lin`, and areas in a pair of files—a `.lin` file for the boundary and a `.pts` file for the centroid.

- The National Topographic System (NTS) mapsheet number; for example, 031h01 is used to name a directory that contains the subdirectories `points`, `lines`, and `areas`.
- Attributes are stored in a comma-separated value (CSV) file in the same directory as the corresponding geometry data.
- Each attribute file contains a minimum set of attributes: `identifier`, `entity_name`, `code_gener`, `code_expli`, `ATG`, `ATZ`, `ATE`, `accuracy` (precision in French), and `angle` is used for point entities only.

Some of these conventions are difficult to follow with an automatically generated mapping file, especially considering the variety of output formats available. Even for ARCGEN output, however, FME cannot completely adhere to these rules without involving manual editing of the generated mapping file.

The FME's approximation to the above conventions are as follows:

- Target dataset is specified by the user at run-time to be the NTS map number. For many formats, this is a directory that contains a separate file for each feature type, or entity file name. Other formats are written to a single file, with different layers or levels, or whatever the target format's terminology is, for the entity files.
- Feature type names are the same eight-character name mentioned in the specification. The seven-digit entity name is determined by looking up the generic code in a predefined tables. Some formats tack on a suffix, such as `_arc` or `_point` to the entity name. The way mapping file generation works in FME, this is unavoidable however, it can be removed by hand once the mapping file has been generated.
- No subdirectories are created in the target directory for `points`, `lines`, and `areas`.
- If the target format were ARCGEN, the file names will all have `.gen` extensions, instead of `.pts` and `.lin`, and no CSV files will be created.
- When possible with the choice of output formats, the attributes are defined as described above. Additional attributes take either the English or French name of the corresponding CCOGIF attributes, depending on the setting of `NTDB_Language`.

To generate a mapping file for the ARCGEN profile, a source format specification of `nt-dbcg`, which is an abbreviation of **NTDB CCOGIF to Generate**, is used. An example of how this is written is:

```
fme generate nt-dbcg ...
```

MID/MIF Profile

The MapInfo Data Interchange Format (MID/MIF) profile is specified in the Geomatics Canada document titled *Conversion of NTDB Edition 3 Data into MID/MIF Format*. This profile has the following properties:

- NTDB entities are written to output files organized by theme. The name of the output file combines the NTS mapsheet number with the theme abbreviation. The table below summarizes the list of themes and their abbreviations.

NTS Mapsheet Themes, Abbreviations, and Numbers

Theme Name	Abbreviation	Theme number
Designated areas	AD	0

Theme Name	Abbreviation	Theme number
Roads	CH	1
Man-made features	CO	2
Relief and landform	FO	3
General	GE	4
Hydrography	HD	5
Hypsography	HP	6
Power network	RE	7
Rail Network	RF	8
Road network	RR	9
Water saturated soils	SS	10
Toponymy	TO	11
Vegetation	VE	12

- Each dataset in a physical volume occupies a directory identified by the NTS number.
- The output coordinate system for NTDB is a UTM system with a NAD83 datum, coordinates in metres, and a scale factor of 0.9996.
- The output file contains at a minimum the following attributes:
 - CODE (explicit code)
 - ATTF
 - ELEVATION
 - ORIENTATION
 - ATV1_ACCURACY
- Other attributes are named `ATFn_<attribute_name>` and `ATVn_<attribute_name>`.
- A MapInfo symbol table is used to represent explicit codes.

Some of these conventions are difficult to follow with an automatically generated mapping file, especially considering the variety of output formats available. Even for MID/MIF output, however, FME cannot completely adhere to all conventions without involving manual editing of the generated mapping file.

The FME's approximation to the above conventions are as follows:

- The profile specification calls for output files that correspond to FME feature types to have names including the NTS map number. Unfortunately, the FME mapping file generation process cannot use a variable name for the output feature types, therefore it generates all mapping files with output feature types of `NTSNUM_<themeAbbrev>`, where `<themeAbbrev>` is a theme abbreviation from *NTS Mapsheet Themes, Abbreviations, and Numbers* on page 355. It is necessary to modify the generated mapping file to include the map number as a part of the output feature type names.

- Where possible with the choice of output formats, the attributes are defined as described above. Additional attributes take either the English or French name of the corresponding CCOGIF attributes, depending on the setting of `NTDB_Language`.

To generate a mapping file for the MID/MIF profile, a source format specification of `ntdbbcm`, which is an abbreviation of **NTDB CCOGIF** to **MID/MIF**, is used. An example of how this is written is:

```
fme generate ntdbcm ...
```

DXF Profile

The DXF profile is specified in the Geomatics Canada document titled "*Conversion of NTDB Edition 3 Data into DXF Format*". This profile has the following properties:

- The data is written to a DXF file for each theme. The themes are the same thirteen themes as those used for the MID/MIF profile.
- The file names for the theme files are `<nts><abbrev>.dxf` where `<nts>` is the NTS map number and `<abbrev>` is the lower-case equivalent of the theme abbreviation listed in the above-mentioned table.
- Entities are stored in layers named `<entityName>_<explicitCode>` where `<entityName>` is the first 11 or fewer characters of the NTDB entity name and `<explicitCode>` is the explicit code of the entity.
- Fixed attributes—`ATFn_<attrName>`—are not stored with the features. Their values are implied by the explicit code.
- Except for toponymy, variable attributes are not transferred to the DXF features.
- There is no area under DXF.

Some of these conventions are difficult to follow with an automatically generated mapping file, especially considering the variety of output formats available. Even for DXF output, however, FME cannot completely adhere to these rules without involving manual editing of the generated mapping file.

The FME's approximation to the above conventions are given here:

- FME considers the target dataset of a DXF file to be the file itself, so it is not possible to generate a number of themes' output files from a single run of FME. To accomplish this, you would have to run the same CCOGIF file through the mapping file for each desired themes. Refer to the discussion under the heading *Theme Selection* for more details.
- The profile spec calls for output files that correspond to FME feature types to have names including the NTS map number. Unfortunately, FME mapping file generation process cannot use a variable name for the output feature types, therefore it generates all mapping files with output feature types of `NTSNUM_<themeAbbrev>`, where `<themeAbbrev>` is a theme abbreviation from *NTS Mapsheet Themes, Abbreviations, and Numbers* on page 355. It will be necessary to modify the generated mapping file to include the map number as a part of the output feature type names.
- When possible with the choice of output formats, the attributes are defined as described above. Additional attributes will take either the English or French name of the corresponding CCOGIF attributes, depending on the setting of `NTDB_Language`.

To generate a mapping file for the DXF profile, a source format specification of `ntdbcd`, an abbreviation of **NTDB CCOGIF** to **DXF**, is used. An example of how this is written is:

```
fme generate ntdbcd ...
```

Run-Time Options

Language Selection (Generic)

Normally, a generic mapping file names groups and attributes by some combination of their French and English names. The generated generic mapping file contains a few lines that may be uncommented to specify that the output files should contain only the French or English data group and attribute names.

Generating Metadata Report (Revision 2)

When a profile-specific mapping file is generated for an NTDB revision 2 CCOGIF file, it generates a report of the metadata in the file. When running the mapping file, the macro `MetadataReportFilename` must be defined. It contains the name of a file where the report is written. If a file already exists with this name, it will be overwritten with the report.

Tip: The advanced user may be interested to know that the actual generation of the report is performed by including the file `$(FME_HOME)/metafile/ntdbv2Report.fmi` into the mapping file.

Profile-Specific Theme Selection

By default, the profile-specific mapping files export all entity data from the input CCOGIF file to the output file. Often you only want to extract a single theme or a set of themes.

This may be performed by specifying a value for the `NTDB_SelectedThemes` macro when running the mapping file. This macro contains a comma-separated list of themes to be exported. If the macro is empty as it is by default, all themes are exported. The themes are specified either by the number or abbreviation in *NTS Mapsheet Themes, Abbreviations, and Numbers* on page 355.

Known Mapping File Issues

When generating mapping files to write to some formats, FME automatically appends geometry type names to the output feature types. Generally, this is a necessary practice for mapping file generation and cannot be overridden. The only ways around this is one of these approaches:

- to modify the mapping file after it has been generated
- to rename the files after the translation has completed

The first approach is more prudent for profile-specific mapping files, as they are likely to be used several times. The modified mapping files can be stored in the FME gallery for future use.

