TRIM.FaTE USER'S GUIDE
MODULE 3: LIBRARY AND SCENARIO DATA FILES[1]

Three types of data files that can be used in TRIM.FaTE are discussed in this module:

- **Import files** (Section 1);
- **Input data files** (Section 2); and
- **Export files** (Section 3).

Import files are used to populate or augment a TRIM.FaTE scenario or library with various objects and information or properties describing them. Input data files provide a way to store values to be assigned to designated properties during a scenario run. Export files are used to document a TRIM.FaTE scenario or library, or may be used as a method to transfer information and settings from one library or scenario to another. This module describes the types of import, data, and export files that are used in TRIM.FaTE, the purpose of these files, how these files are used within the TRIM.FaTE modeling framework, and the required syntax for each file type. Example data files or excerpts are included within the text in Courier font and as attachments to this module.

1. IMPORT FILES

Import files are used to import objects, and information or properties describing objects, into a TRIM.FaTE scenario or library. Although almost all the information needed for a TRIM.FaTE simulation can be entered directly into TRIM.FaTE (i.e., by using the graphical user interface (GUI) to manually enter the information into the library or scenario), the use of import files automates part of the scenario set-up process. The exception – where information *cannot* be entered using the GUI – is the addition of parcel and volume element information to a scenario; use of the Volume Element Import file is required by TRIM.FaTE for this process. Use of import files for the remainder of the scenario set-up steps is desirable because this method provides significant advantages over GUI input in terms of speed, flexibility, facilitation of the development of multiple similar scenarios, and record keeping.

A TRIM.FaTE import file is constructed by the user outside of the TRIM.FaTE model using a text editor and saved as a text file with the extension *.txt*. A specific format and syntax are required for import files. The user imports these files into TRIM.FaTE using commands on the pull-down menus within the TRIM.FaTE project and scenario windows. The specific command used depends on the type of data and file being imported.

Import files can be used to provide information for either a TRIM.FaTE library or a specific TRIM.FaTE scenario. The different types of import files are summarized here and described in more detail in the sections that follow.

[1] Descriptions of library-specific algorithms and properties presented in this module pertain to the March 2003 versions of the TRIM.FaTE library.

Library Import files (Section 1.1) are used to load data into a TRIM.FaTE library. Two types of library import files can be used.

- An **Object Import file** (Section 1.1.1) is a text file that is used to load the following types of objects into a new or existing library: property types, chemicals, sources, compartment (or composite compartment) types, or algorithms.

- A **Property Type Spreadsheet Import file** (Section 1.1.2) can be used to import new property types into a new or existing library.

Scenario Import files (Section 1.2) are used to load data into an existing TRIM.FaTE scenario. Four types of scenario import files can be used.

- A **Volume Element Import file** (Section 1.2.1) *must* be used to load the spatial layout data for a scenario.

- A **Compartment Import file** (Section 1.2.2) can be used to add new compartments into the volume elements that have been defined for a scenario.

- A **Property Import file** (Section 1.2.3) can be used to set property values for any objects that have been defined in a scenario (e.g., compartment properties, link properties).

- A **Run Import file** (Section 1.2.4) allows the user to set varying property values and links for *multiple* simulations via a single file and then run these simulations sequentially.

Each of these files is described in more detail in the sections that follow.

USER TIP: INSERTING COMMENTS

In some cases, it may be useful to insert comments into an import file for reference (e.g., to explain the purpose of the subsequent lines of text, to insert the date that the file was last edited). User comments can be inserted into all types of import files using either of two methods:

- "//" indicates that all text until the end of the line is a comment. This type of comment does not necessarily have to be inserted on its own line (i.e., it can be inserted at the end of a non-comment line).

- "/*" indicates that all text until a "*/" is a comment. This type of comment can span multiple lines. However, it cannot be embedded in other values; it must start on its own line.

In addition, the user can insert blank lines throughout an import file.

1.1 Library Import Files

Library Import files can be used to import qualitative or quantitative data into a library. Two types of Library Import files can be used: an Object Import file (Section 1.1.1) or a Property Type Spreadsheet Import file (Section 1.1.2).

1.1.1 Object Import File

Purpose and Use

TRIM.FaTE libraries can be created or expanded by using the Object Importer to import an Object Import file. The Object Importer reads a textual description of property types, compartments, algorithms, chemicals, sources, and composite compartments in the Object Import file format and adds these objects to the specified library.

An Object Import file can be imported into TRIM.FaTE by carrying out the following steps.

- In the library window of an open TRIM.FaTE library, select the File pull-down menu and click on "Import." A Choose Importer window will appear.
- Highlight "Object Importer" in the Choose Importer window and click "OK." An Import TRIM Objects window will appear that allows the user to browse files on the computer.
- Find the Object Import file, highlight, and click the "Import TRIM Objects" button. After importing the data, TRIM.FaTE will provide a message window that either indicates a successful import or informs the user of any errors that were encountered.

Format and Syntax

The first non-comment line of the file specifies the file format version and must contain the keyword "Version:" followed by the current file format version number (currently "1"). Note that the file format version is different from the TRIM.FaTE model version number.

```
Version:1
```

The subsequent lines are used to specify object definitions. The Object Import File can be used to define the following types of objects in the library:

- Property types;
- Chemicals;
- Sources;

- Compartments;
- Composite compartments; and
- Algorithms.

Objects are defined by sets of keywords and values. The case of keywords is ignored, but the case of values is retained. Values can be specified in two ways.

(1) Values that do not span multiple lines can simply appear after the object name and colon. All white space between the colon and the value, and between the value and the end of the line, will be ignored by the Object Importer.

(2) For values that span multiple lines, the values must be preceded and followed by double square brackets (i.e., "[[" and "]]"). Embedded new line characters and adjacent white space are converted to one space. Carriage returns can be inserted into the value using "\n". A backslash can be inserted into a multiline value using "\\".

The first line describing a particular object must begin with the keyword specifying the type of object followed by a colon and the name of the object. For example,

```
Compartment: Air
```

This line would indicate the beginning of the description of a compartment with the name "Air." The keywords for each of the object types that can be included in an Object Import file are listed in Table 1.

Table 1
Object Type Keywords

Object Type	Keyword
Property Type	Ptype
Chemical	Chemical
Source	PointSource
Compartment	Compartment
Composite Compartment	CompCompartment
Algorithm	Algorithm

There are restrictions on the forms of names for property types. Property type names must not include any spaces and must consist of only letters, numbers, and underscores (i.e., "_"). There are no restrictions on names of the other object types.

The lines following the initial line of the object definition contain additional keywords and values describing the object. The available keywords depend on the type of object being

created. Property types are defined by the keywords described in Table 2. Each keyword should be followed by a colon and the value associated with the keyword. The keywords can be presented in any order, provided the "Ptype" keyword and the name of the property type are presented first. It is important to note that the keywords for property types cannot be repeated (i.e., each keyword can only occur once per property type). The following is an example of the property type definition.

```
Ptype: AirDensity
DataType: FloatingPoint
DefaultValue: 2.5
Description: Density of particles in air
Units: g/m3
Max: 5.0
Min: 0.0
```

Table 2
Property Type Object Keywords

Keyword	Required/ Optional	Applicable Data Types	Definition
DataType	Required	All	Specifies the type of information that a property provides. In many cases this will be the type of information stored in the property. Valid data type are: Boolean, Category, DateTime, FloatingPoint, Integer, and String.
DefaultValue	Optional	All	Specifies the value to which each new property of this type will be set. If no default value is provided, the values all of new properties of this type will be unset.
Description	Optional	All	Provides a textual description of the property type. Can be used to describe the purpose of a property type and/or any guidelines that should be used in setting the property values.
Max	Optional	FloatingPoint, Integer	If a maximum value is specified, the user is warned when a value for a property of this type exceeds this maximum.
Min	Optional	FloatingPoint, Integer	If a minimum value is specified, the user is warned when a value for a property of this type is below this minimum.
Units	Required	FloatingPoint, Integer	Specifies the units for each property of this type. If the property type describes a unitless value, the user should use "unitless" or "n/a" as the units.

All objects except property types are defined by properties. Required properties do not have to be set when the object is imported, but they must be set before a simulation can be performed. Following the first line of the object definition, which specifies the type of object and the object name (e.g., Source: Acme Facility - Ohio, Compartment: surface soil, Chemical: benzo(a)pyrene), the keyword "Property" should appear followed by a colon and the name of a property type that will be used to define the object. On subsequent lines, the property should be defined by one or more property keywords. Acceptable keywords are listed and described in Table 3. These keywords must be presented in the order they are listed in the table; however, only the "Value" keyword is required for a given object. Each object can have multiple properties associated with it, and therefore there is no limit to the number of times the "Property" keyword (with its associated keywords defining the property) can be repeated. An example of a non-property type object definition that could appear in the Object Import file is presented below.

Table 3
Property Keywords

Keyword	Required/ Optional	Definition
Form	Optional	Indicates how the value of the property value is represented. The available options for this property are Constant, Formula, and InputFromFile. If no form is entered for a given property, the last form entered for the object will be used by default. This allows the user to group values of the same type together without having to specify the form for each one.
Value	Required	Specifies the value of the property. The form of the value must match that specified using the "Form" keyword. If the "Form" keyword is not specified, the value must be a Constant. Boolean values, categories, and Date/Time values are all considered Constants. For chemical-specific properties, the value is preceded by the chemical name in curly braces (e.g., {Elemental Mercury}). For properties that apply to multiple chemicals, the "Value" keyword should be repeated for each chemical on separate lines.
Description	Optional	Provides a description of the property (e.g., source of a value).

USER TIP: AVOID DUPLICATE OBJECT NAMES

When importing objects into an existing library using the Object Importer, be careful that the names of the objects being imported do not match any of the objects already in the library. Likewise, when importing objects into a new or existing library, be careful that the Object Import file does not include two objects of the same type with the same name. In both cases, TRIM.FaTE will generate an error upon import of the Object Import file.

```
Source: Acme Facility
Property: elevation
Value: 61.0
Description: Set to main stack height
Property: emissionRate
Value: {Benzo(A)Pyrene} 0.0070
Value: {Divalent Mercury} 0.051
Value: {Elemental Mercury} 5.1E-4
Property: enabled
Value: true
Property: X
Value: 553472.0
Description: [[X-coordinate of the source; set to X coordinate of centroid
of parcel]]
Property: Y
Value: 4590052.0
Description: [[Y-coordinate of the source; set to Y coordinate of centroid
of parcel]]
```

Composite compartments require one additional keyword, "Component," in addition to the other property keywords. This keyword specifies the name of a compartment that should be included in the composite compartment. This keyword is specified once on a separate line for each compartment included in the composite compartment. It is important to note that the compartments referenced using the "Component" keyword must be already defined in the library or must have been defined previously in the Object Import file. An example of a composite compartment definition is provided below.

```
CompCompartment: Agriculture - General
Property: acceptableAbiotic
Value: Soil | Surface Soil | Surface Soil - Default
Property: concentrationOutputFactor
Value: 1.0
Property: isBiotic
Value: false
Component: Leaf - Agriculture - General
Component: Leaf Particle - Agriculture - General
Component: Root - Agriculture - General
Component: Stem - Agriculture - General
```

An example of an Object Import file is included in Appendix A.

1.1.2 Property Type Spreadsheet Import File

Purpose and Use

New property types can be added to a new or existing TRIM.FaTE library via a Property Type Spreadsheet Import file. As described in Section 1.1.1, property types can also be added via an Object Import file. However, using a Property Type Spreadsheet Import file can facilitate user input of multiple property types and related data by allowing the user to create this file as a spreadsheet.

A Property Type Spreadsheet Import file is imported into TRIM.FaTE by following the same steps outlined in Section 1.1.1 for the Object Importer, with one exception: the user should select "Property Type Spreadsheet Importer" in the Choose Importer window that appears.

Format and Syntax

The format of the import file is a tab-delimited text file comprised of columns of data. This type of file can be created most easily with a spreadsheet program by using the "Save As" function and selecting "Text (Tab delimited)." The file must contain the following columns:

- Property Type
- Data Type
- Units
- Default
- Min
- Max
- Description

The "Property Type" column indicates the name of the property type being imported. The remaining columns correspond to those property type object keywords described in Table 2. The "Property Type" and "Data Type" columns must be populated for each property type being imported. Likewise, the "Units" column must be populated for all property types with "FloatingPoint" or "Integer" data types. The remaining columns are optional. An sample of part of a Property Type Spreadsheet Import file is presented below.

Property Type	Data Type	Units	Default	Min	Max	Description
Accumulation	Integer	n/a	1.0005			Accumulation
CompartmentName	String		AirComp	0.0		CompartmentName
StartRunDate	DateTime			0.0		Start of Run
RunNumber	Integer	n/a				Air Schmidt No

1.2 Scenario Import Files

Scenario Import files can be used to import data into an existing TRIM.FaTE scenario. Four types of files can be used to import data into a scenario: a Volume Element Import file (Section 1.2.1), a Compartment Import file (Section 1.2.2), a Property Import file (Section 1.2.3), or a Run Import file (Section 1.2.4).

1.2.1 Volume Element Import File

The Volume Element Import file contains all information required by TRIM.FaTE to define the parcels and volume elements, with their associated abiotic compartments, for a scenario. Importing this file is the only way to add these data to a scenario (i.e., there is not an alternative "manual" method for entering these data by typing data values directly into the GUI). To import the Volume Element Import File for a scenario, the user should complete the following steps:

- In the scenario window, click on the *File* pull-down menu and select "Import Volume Elements." The Import Volume Elements window will appear that allows the user to browse files on the computer.

- Use this browser to find the Volume Element Import file and highlight the file. Click the "Import Volume Elements" button to import the file.

- TRIM.FaTE will then provide a message window that either indicates a successful import or informs the user of any errors that were encountered.

The user should refer to **Module 5, Developing the Spatial Layout**, for detailed information on the format of the Volume Element Import file and suggested methods for acquiring the data included in this file. An example Volume Element Import file is included as an appendix of Module 5.

1.2.2 Compartment Import file

Purpose and Use

The Compartment Import File allows the user to automate the selection of Compartments from a library into a scenario.[2] The user can create a Compartment Import File manually (i.e., by creating a text file according to the format/syntax rules below) or "automatically" by exporting the list of compartments used in a given scenario. This file is designed to help simplify the process of creating a scenario setup by allowing the user to easily import a list of compartments into a scenario. In addition, an exported list of compartments can be easily edited and then imported into a new scenario to allow changes to scenario setups to be implemented quickly.

Compartments can be added to a scenario via the Compartment Import file by following these steps:

- In the scenario window, click on the *File* pull-down menu and select "Read Compartments From File." A window entitled *Open* will appear that allows users to browse the files on the computer.

- Use this browser to find the Compartment Import file and highlight the file. Click the "Open" button to import the file.

- TRIM.FaTE will then provide a message window that either indicates the number of compartments that were imported or informs the user of any errors that were encountered.

[2]Note that any compartments added to a scenario (via either a compartment import file or by manually selecting and adding compartments to the scenario) must exist in the library that is associated with the scenario.

Format and Syntax

The first non-comment line of the file must contain the keyword "Version:" followed by the current file format version number (currently "1").

```
Version:1
```

The remainder of the file consists of sections which start with a volume element on one row, followed by compartment names on subsequent rows. To place compartments that are in the Library into a specific volume element, first use the keyword "VolumeElement:" followed by the name of the volume element in the Scenario.

```
VolumeElement:SW_Silver Lake
```

Following the specification of a volume element, the compartments to be included in the volume element are specified on the next row(s). To do this, use the "Compartment:" keyword followed by the names of the Compartments in the Library. There are no limitations on the number of compartments that can be included in a volume element; however, each compartment included in a particular volume element must be unique (e.g., the user cannot include two compartments named "Long-tailed Weasel" in the same volume element).

```
Compartment:Common Loon
Compartment:Mallard
Compartment:Macrophyte
```

To add compartments to another volume element, add another section that starts with the Volume Element name and is followed by the Compartment names. Comments can be inserted anywhere in the file using the same syntax as the Object Import file (described above). Note that spaces can be included in volume element and compartment names (the text must match the exact name specified in the original volume element file).

At the end of the Compartment Import file, ***there must be a hard return inserted for TRIM.FaTE to read the file correctly.*** Without the return after the last line, TRIM.FaTE will not read the import file and will return an error message. A sample Compartment Import file is presented in Appendix B.

Writing Compartments to File

A Compartment Import file can also be created automatically by TRIM.FaTE for a scenario that has been created by the user. This function allows the user to create a new scenario or edit an existing scenario manually via the GUI, save the compartment list to a file, and set up a new scenario using this file as the Compartment Import file. This file can be created by following these steps:

- In the Scenario window of a simulation that has been set up (i.e., all compartments have been assigned to volume elements), click on the *File* pull-down menu and select "Write Compartments..." A *Save* window will appear with a file browser.

- Use this file browser to select the location where the new Compartment Import file will be saved. Put the cursor in the small window labeled "File name:" and enter the desired name for the new import file (make sure the file is named with the *.txt* extension to facilitate any future editing of the file).

- Click the "Save" button to create the new import file. A new Compartment Import file will be created as specified.

If desired, the user can open this new Compartment Import file using a text editor and make additional changes.

1.2.3 Property Import Files

Purpose and Use

Property Import files can be used to set property values and create links for a TRIM.FaTE scenario. This import file is designed to simplify the process of creating a scenario setup by allowing the user to set the property value(s) for certain objects (i.e., scenarios, chemicals, compartments, volume elements, links, or algorithms) and create links without having to manually accomplish these tasks via the GUI. This feature is very useful because multiple iterations of a scenario are generally required to accommodate troubleshooting and changes to property values.

Many property values can (and should) be set in the library used to set up the scenario and therefore do not need to be set at the scenario level. However, any spatially-varying properties (i.e., properties specific to a certain compartment, such as pH for a specific water body) cannot be established as library properties because the property values for those compartments would not be the same for all compartments of that type in the scenario. In addition, links and link properties cannot be defined in the library because the links between individual compartments and sinks are created in a scenario, not a library. Therefore, a Property Import file is especially useful for providing an automated method for entering this information.

Note that *links* for a scenario and the corresponding *properties* of these links must be imported in *separate* Property Import files. This is required because the links have to be established before any properties can be assigned to them. In addition, multiple Property Import files containing properties for different objects can be imported to the same scenario if desired.

A Property Import File can be added to a scenario by following these steps:

- In the scenario window, click on the *File* pull-down menu and select "Load Properties From File." A window entitled *Load Properties for [scenario]* will appear (with the scenario name inserted in place of the brackets).

- Click the "Browse" button on this window and use the file browser in the window that appears to find the Property Import file. Highlight the file and click the "Open" button.

- The file name will appear in the small window at the top of the *Load Properties for [scenario]* window.

- Click the "Override" button at the bottom of this window to import the file. ***Note that property values imported via a Property Import file will overwrite any previously existing property values in the scenario (i.e., those values existing in the library or values entered manually by the user prior to the property import process).***

- After the property import process is complete, TRIM.FaTE will provide a message window that either indicates a successful import or informs the user any errors that were encountered.

Format and Syntax: General Guidelines

The first non-comment line of the Property Import file must contain the keyword "Version:" followed by the current version number (currently "1").

```
Version:1
```

The second non-comment line of the file must contain the keyword "Scenario:" followed by the name of the scenario for which the property values and links will be specified.

```
Scenario: <scenario name>
```

The third non-comment line of the file must contain the keyword "Run:" followed by the run name. When using the Property Importer, the run name should always be "BaseRun."

```
Run: BaseRun
```

The remainder of the file is used to specify the property values and links for the scenario, as described in the next two sections.

Format and Syntax for Property Values

The format for setting property values is very similar to that used in the Object Import file. To set the properties for a particular object, the keyword associated with the object type must be entered, followed by the name of the object. The available object types and the name formats for each type are presented in Table 4.

Table 4
Property File Object Types and Keywords

Object Type	Keyword	Object Name Format [a]
Scenario	Scenario:	[*Scenario Name*]
Chemical	Chemical:	[*Chemical Name*]
Volume Element	VolumeElement:	[*Volume Element Name*]
Compartment	Compartment:	[*Compartment Name*] in [*Volume Element Name*]
Link	Link:	[*Compartment Name*] in [*Volume Element Name*] to [*Compartment Name*] in [*Volume Element Name*]
Algorithm	Algorithm:	[*Algorithm Name*]

[a] Text in brackets would be replaced with actual names in the Property Import file.

Examples of object type keywords and object names are presented below.

```
Scenario: LeadSmelting
Chemical: Elemental Mercury
Volume Element: Air_Source
Compartment: Air in Air_Source
Link: Air in Air_Source to Soil - Surface in SurfSoil_Source
Algorithm: Wet Particle Deposition from Air to Surface Soil
```

On the line following the object type keyword and object name, the user must specify the property (or properties) associated with the object for which values will be specified. The properties are specified using the "Property:" keyword followed by the name of the property.

```
Property: averageResultsFiles
```

After specifying a property, the user must specify on the next line the form of the value of the property using the "Form:" keyword followed by one of the following:

- InputFromFile
- Formula
- Constant

If no form is specified, TRIM.FaTE defaults to the form of the previous property for the selected object. If none of the properties have specified the form for this object, TRIM.FaTE assumes that the form is "Constant."

Next, the user must specify a value in the appropriate form. Values are specified using the "Value:" keyword followed by the value in the appropriate form. Syntax requirements for each of the three forms are described separately below.

Input From File. Values in the "InputFromFile" format (i.e., data stored outside of TRIM.FaTE in a time-stepped data file, as described in Section 2) should be specified as follows:

Value: <Path of Input File>, <Column Name in Input File>, "<File delimiter>"

For example:

```
Value: C:\\TRIM\\data\\LeadSmelting\\MetData.csv, Mixing Height, ","
```

Formula. Values in the "Formula" format should be specified using the same formula syntax used for specifying formulas throughout TRIM.FaTE (formula syntax is described in Module 4, Adding New Components to a Library). For example:

```
Value: containingScenario.AirTemperature_K * 100
```

Constant. Values in the "Constant" format should be specified as follows.

Value: <Constant Value>

For example:

```
Value: true
```

For any chemical-specific properties, the values are specified by preceding the value with the chemical name in curly braces (e.g., {Elemental Mercury}). For properties that apply to multiple chemicals, the "Value" keyword should be repeated on separate lines for each chemical. For example:

```
Value: {Elemental Mercury} 150.6
Value: {Divalent Mercury} 148.9
```

To specify the same value for multiple objects, the user should list the object names on separate lines followed by the form and value lines, as demonstrated in the following example.

```
VolumeElement: Air_ESE2
VolumeElement: Air_SSE2
VolumeElement: Air_SSW2
VolumeElement: Air_WSW2
VolumeElement: Air_WNW2
VolumeElement: Air_NNW2
Property: top
Form: InputFromFile
Value: C:\\TRIM\\data\\LeadSmelting\\MetData.csv, Mixing Height (m), ","
```

After specifying the value (or values) for the property, the user can specify additional property values for the object or specify property values for another object. To specify additional property values for the current object, the user can simply specify another property name and value(s). There is no need to repeat the object name. To specify property values for another object, the user must specify another object followed by the associated properties and values.

Format and Syntax for Links

In addition to specifying property values, the user can create links using a Property Import file. To do this, the user must first specify the keyword "NewLink:". On the following line, the user should specify the sending compartment using the "SendingCompartment:" keyword followed by the name of the sending compartment. On the next line, the user must specify the receiving compartment using the "ReceivingCompartment" keyword followed by the name of the receiving compartment. After specifying the receiving compartment, the user can optionally create a link going in the opposite direction (i.e., from receiving compartment to sending compartment) using the "ReciprocalLink:" keyword. It is important to note that this option results in the creation of an additional link; that is, if a reciprocal link is specified, *two* links will be created between this pair of compartments (i.e., one in each direction).

The final step in creating the link is to specify which algorithms should be applied on this link. An algorithm is specified on a new line using the "Algorithm:" keyword followed by the name of the algorithm. This keyword can be repeated if the user would like to include more than one, but not all, of the available algorithms on the link. A value of "Default" can be entered if the user would like to include all available, pertinent algorithms on the link. The format of each of the keywords described above is presented in Table 5.

Table 5
Keywords for Creating Links

Keyword	Format [a]
SendingCompartment:	[*Sending Compartment Name*] in [*Volume Element Name*]
ReceivingCompartment:	[*Receiving Compartment Name*] in [*Volume Element Name*]
ReciprocalLink:	[*Receiving Compartment Name*] in [*Volume Element Name*] to [*Sending Compartment Name*] in [*Volume Element Name*]
Algorithm:	[*Algorithm Name*] -OR- Default [b]

[a] Text in brackets would be replaced with actual names in the Property Import file.
[b] If "Default" is added after the keyword "Algorithm," TRIM.FaTE will assign all appropriate default algorithms that apply to that combination of sending and receiving compartment types.

An example of text specifying a new link is presented below.

```
NewLink:
SendingCompartment: Air in Air_WSW1
ReceivingCompartment: Meadow Vole in SurfSoil_SW2
ReciprocalLink: Meadow Vole in SurfSoil_SW2 to Air in Air_WSW1
Algorithm: Default
```

An example Property Import file containing both properties and links is presented in Appendix C. Refer to Module 8, Links and Algorithms, for more information on links.

1.2.4 Run Import Files

Purpose and Use

The Run Import file allows property values and links for *multiple* simulations to be read into TRIM.FaTE from a file. By using this file in conjunction with the TRIM.FaTE Run Importer function as described below, several runs for a single TRIM.FaTE scenario can be set up (each with a different set of properties and links and a unique run name) and then executed in one sequential batch. The scenario for each run will differ from the base scenario named in the file (i.e., the original scenario) only by the property values and links specified for each new run in this file. After each run, the scenario properties are restored to their original, base values and any new links created by the by the Run Import file are deleted. The Run Import feature can be useful for testing and diagnostic purposes (e.g., running a scenario several times while changing one or more parameters).[3]

Conceptually, a Run Import file is similar to a Properties Import file in that it contains property values and links; however, the Run Import file contains multiple "sets" of properties and links (i.e., one set of properties/links per run). A set of runs (i.e., one for each set of properties/links) can be initiated via the Run Importer and executed with a single command. The scenario for each run will differ from the original scenario only by those properties specified in the Run Import file. The user should designate that results from each individual run be stored in a separate file location so that subsequent runs do not overwrite the results from the previous run (note that the results file location can be set as a run-specific scenario property in the Run Import file).

A Run Import file can be used with the Run Importer function to set up sequential runs with different properties by following these steps:

- In the scenario window, click on the *File* pull-down menu and select "Load Runs From File." A dialog window entitled *Load Runs for [scenario]* will appear.

[3] TRIM.FaTE sensitivity and Monte Carlo analyses operate in a similar manner, in that a base scenario and multiple runs are executed sequentially. However, sensitivity and Monte Carlo analyses do not require the use of Run Import files; the set of runs required for those analyses is executed directly by the TRIM.FaTE sensitivity and Monte Carlo features. Refer to Module 15, TRIM.FaTE Sensitivity and Monte Carlo Analyses, for a complete description of these features.

- Click the "Browse" button and use the file browser window *Open Run File* that appears to find the appropriate Run Import file. Select the appropriate file and click the "Open" button.

- At this point, the Run Import file name will appear in the smaller pane at the top of the *Load Runs for [scenario]* window, and each run included in this file will appear in the larger pane on the bottom half of the window. The user should select the runs to be completed by highlighting the run names; multiple runs can be selected by holding the control or shift keys while clicking on the run names.

- Once the runs are selected, the user can initiate the series of runs by clicking the "Run" button. As with the Property Import file, any property values imported via a Run Import file will overwrite the previous property value existing in the scenario. TRIM.FaTE will change the appropriate property values and execute the simulations according to the properties specified.

- After the set of runs has been completed, TRIM.FaTE will provide a message window that either indicates successful runs or informs the user of any errors that were encountered.

Format and Syntax

The format of the Run Import file is identical to that of the Property Importer described in Section 1.2.3, with the two exceptions noted below. An example of a Run Import file is presented in Appendix D.

- The user has the option of including an additional line following the "Version" and "Scenario" lines that specifies the number of runs specified in the Import file. This line uses the keyword "NumberOfRuns:". This line may be helpful to the user as a reference, but it is not required by TRIM.FaTE because the model can determine the number of runs included in the file without this line.

- The Run Import file will contain data for multiple runs. Therefore, the set of property values and links specific to a run must begin with a line with the keyword "Run:" followed by a run name defined by the user. That run will include all of the property values and links until the next "Run:" keyword is encountered by the model.

2. INPUT DATA FILES

Certain types of TRIM.FaTE data are stored in files *external* to the scenario file, either by necessity or for convenience. To use input data stored in an external file, the user prepares a file according to certain formatting conventions and then inserts a reference into TRIM.FaTE that identifies the location of the input data file. Three types of input data files can be used by TRIM.FaTE:

- **GIS Overlay Files** (Section 2.1), used to specify GIS information for TRIM.FaTE results visualization tool;

- **Statistics Files** (Section 2.2), used to specify information about property distributions for probabilistic analyses; and

- **Time-varying Data Files** (Section 2.3), used to store values that change over time.

The first two types of files, GIS Overlay and Statistics files, are required for TRIM.FaTE results visualization and probabilistic analyses, respectively. Time-varying Data files are not *required* for a TRIM.FaTE run, but are a very convenient way to gather and enter values for properties that change several times over the course of a simulation. Each of these three data file types is discussed in more detail in the sections that follow.

2.1 GIS Overlay Files

Currently, the results from a simulation run can be viewed using the TRIM.FaTE Graphical Results Viewer (see Module 14, Simulation Results and Analyses, for a complete description of the Graphical Results Viewer). Results are analyzed using the Viewer in the context of the volume element types included in the simulation, where the type of volume element is defined by the abiotic compartment associated with the volume element (e.g., surface water, sediment, groundwater). GIS overlay files serve as a supplementary tool that can be used when viewing TRIM.FaTE results in the Viewer. An overlay file is a text file that is used to add additional "layers" of information when viewing the results within TRIM.FaTE. Specifically, overlay files can be used to create:

- Polygon shapes;
- Symbols; and
- Points.

In addition, the user can specify text to accompany each of these items. The user can also specify formats for the shapes that are added, including details such as line thickness and fill color. For example, if the user would like to see the results of a TRIM.FaTE simulation as they relate to a particular body of water, they can set up an overlay file that will add information about the water body to the visualization. This information could be added in the form of a polygon approximating the shape of the water body, a point indicating a location of interest (e.g., the center of the water body), or text providing additional information (e.g., the name of the water body).

In order to set up an overlay file, the user must enter applicable coordinate values (e.g., latitude and longitude) and specifications (e.g., line colors, shapes, text orientation) for the objects that will be added to the overlay text file. Table 6 outlines the various parameters that can be specified in a GIS overlay file. Examples of overlay files containing labeled points and polygons are presented in Appendix E.

In addition to "manually" adding information to be viewed with TRIM.FaTE results via an overlay file, users can import existing overlay files that contain "standard" GIS files in *.shp format. These GIS files can contain information such as road locations, watershed boundaries, and county borders. To import *.shp files into the TRIM.FaTE Graphical Results Viewer, the user should follow these steps.

(1) Open the TRIM.FaTE Graphical Results Viewer with a TRIM.FaTE output file (refer to Module 14, Simulation Results and Analyses, for more details).

(2) Select "Edit Layers" from the "Layer" pull-down menu.

(3) Click the "Add" button in the "Edit Layers" pop-up window.

(4) Select a *.shp file using the File Browser and click "Open." The selected file will be loaded into the visualization of the TRIM.FaTE results within the Viewer.

Table 6
GIS Overlay File Formatting Conventions

GIS Overlay File Parameter	Format Conventions	Default Value
Projection Information	(1) The first line must contain the keyword PROJECTION followed by the type of Projection, which must be one of LatLon, Lambert or UTM. (2) The next line must contain the keyword ELLIPSOID followed by the ellipsoid to be used with the projection. The Ellipsoids must be one of: Clarke 1880, GRS 1967, GRS 1980, International 1909, International 1924, SGS 1985, Sphere, WGS 1960, WGS 1966, WGS 1972, WGS 1984. (3) After this, each projection has a required set of parameters that is different for each one. – For the LatLon projection, the CenterLongitude parameter is required. – For the Lambert projection, the following parameters are required: NorthLatitude, SouthLatitude, CenterLongitude, and CenterLatitude. – For the UTM projection, the UTMZone parameter is required.	NA
Labeled Points	The user may specify sets of labeled points in a points section which must begin with BEGIN_POINTS and end with END_POINTS. The user must specify the type, color, and size of the symbol to be used to represent the point. In addition, the user must provide a text label for each point specified in this section. The user can specify the text font, color, and orientation for each point.	NA
Symbol Type	To set the symbol type, the user should use the SHAPE keyword followed by one of the following shapes: circle, square, triangle, diamond, cross, star, or none. – Example: SHAPE: circle	circle

GIS Overlay File Parameter	Format Conventions	Default Value
Symbol Color	To set the symbol color, the user should use the SYMBOLCOLOR keyword and one of the available colors: black, blue, cyan, darkGray, gray, green, lightGray, magenta, orange, pink, red, white, or yellow.	black
Symbol Size	To set the symbol size, the user should use the SYMBOLSIZE keyword followed by an integer that is the size of the symbol in pixels.	5
Font	To set the font type, the user should use the FONT keyword followed by one of the fonts available on your system.	Arial
Font Style	To set the font style, the user should use the FONTSTYLE keyword followed by plain, bold, italic or bolditalic.	plain
Font Size	To set the font size, the user should use the FONTSIZE keyword followed by an integer that is the font size in points (e.g., 10, 11, 14).	12
Font Color	To set the font color, the user should use the FONTCOLOR keyword followed by one of the available colors: black, blue, cyan, darkGray, gray, green, lightGray, magenta, orange, pink, red, white, or yellow.	black
Text Orientation	The text label can be oriented about the symbol in any of the eight compass directions or in the center of the point. To set the orientation, the user should use the ORIENTATION keyword followed by one of the following directions: north, northeast, east, southeast, south, southwest, west, northwest, or center.	east
Point Label and Location	To specify the location and label for a point, the user should specify a line with some text, the X coordinate, and the Y coordinate separated by commas. – Example: Office building, 64.546, 44.325 Multiple points must be specified on separately.	NA
Polygons	The polygon section must begin with the keyword BEGIN_POLYGONS and end with the keyword END_POLYGONS. Polygons may be either closed filled polygons, closed unfilled polygons or open polygons. The user can specify the outline color, the fill color, the line thickness and the line style. As with the points, these parameters are specified and they remain in effect until changed by the user. The user may set the attributes, specify several polygons and then change the attributes for the next set of polygons.	NA
Polygon Type	To set the polygon type, the user should use the TYPE keyword and one of the following types: filled, outline, line. A filled polygon will have an outline and an interior filled with a color. An outline polygon will be a closed polygon with an outline. A line will be an open polygon that does not close.	outline

GIS Overlay File Parameter	Format Conventions	Default Value
Fill Color	The fill color of a filled polygon can be set with the FILLCOLOR keyword followed by one of the available colors: black, blue, cyan, darkGray, gray, green, lightGray, magenta, orange, pink, red, white or yellow. This will only be used for polygons that are filled.	black
Line Color	The line color of any polygon can be set with the LINECOLOR keyword followed by one of the available colors: black, blue, cyan, darkGray, gray, green, lightGray, magenta, orange, pink, red, white, or yellow.	black
Line Thickness	The line thickness of any polygon can be set with the LINECOLOR keyword followed by an integer that is the line thickness in pixels.	1
Line Style	The line style of any polygon can be set with the LINESTYLE keyword followed by one of the following styles: solid, dashed, long dashed, double dashed, dash dot, or dash double dot.	normal
Specifying a Polygon	To specify a polygon, the user should use the NUMPOINTS keyword followed by the number of points that will be in the polygon. Then place the points, one per line, following this with X coordinate and Y coordinate separated by a comma. – Example: NUMPOINTS:4 -68.75, 44.70 -68.80, 44.70 -68.80, 44.75 -68.75, 44.75	NA

2.2 Statistics Files

A TRIM.FaTE Statistics file provides necessary information about the distributions of property values for use in sensitivity and Monte Carlo analyses. In TRIM.FaTE, a sensitivity analysis generally requires a coefficient of variation (CV) for each varied parameter and a Monte Carlo analysis generally requires the distribution type (see below for exceptions). The TRIM.FaTE Statistics file is a comma-delimited text file that contains data for each property that will be varied in these analyses.

Eleven columns are required for this file: Property, Chemical, ObjectType, ObjectName, Minimum, Maximum, CV (i.e., coefficient of variation), Distribution, and three additional distribution parameters. The first line of the file must list the column names using the syntax specified in Table 7. Each row of the file is specific to a property that will be varied in the analyses. Although all 11 columns must appear in the statistics file, each column does not necessarily need to contain a data value in all rows. The inclusion of values in the columns for a particular property depends on the distribution assigned to the property and whether or not the property is chemical-specific. For example, the "chemical" field will be empty for properties

that are not chemical-specific, and the assignment of particular distributions (e.g., uniform and triangular) results in the need to provide values for additional statistical parameters (e.g., minimum and maximum estimates).

Each of these columns is described in Table 7. Note that data values are required for some columns (i.e., data values must be entered in that column for all or certain circumstances). The order of the columns in the Statistics file is not important; however, the columns *must* be named according to the names listed in Table 7 for TRIM.FaTE to identify them correctly. The user may also insert additional columns in the file to provide other information if desired (e.g., units, references).

Table 7
Statistics File Columns

Column Name	Data Type	Required? [a]	Description
Property	Text	Required for all properties.	The name of the property.
Chemical	Text	Required for statistics of chemical-specific properties only.[b]	The chemical associated with the property.
ObjectType	Text	Required for all properties.	The type of object to which this property applies (e.g., compartment, chemical).
ObjectName	Text	Required for all properties.	The name of the object (such as the compartment type) to which this property applies (e.g., air, surface water, benthic carnivore).
Minimum	Numeric	Required for properties with uniform and triangular distributions.	The minimum allowed value for this property.
Maximum	Numeric	Required for properties with uniform and triangular distributions.	The maximum allowed value for this property.
CV	Numeric	Required for properties with normal and lognormal distributions.[c]	The coefficient of variation for this property (used for sensitivity analysis only). Equal to the standard deviation divided by the mean. Required as a user input for normal and lognormal distributions only.
Distribution	Text	Required for all properties.	The type of distribution assigned for this property. TRIM.FaTE currently supports four distributions: normal, lognormal, uniform, and triangular.
Dist_Param_1	Numeric	Required for properties with a lognormal distribution but a base value of zero.	For these properties, the non-zero mean of the distribution.

Column Name	Data Type	Required? [a]	Description
Dist_Param_2	Numeric	Required for properties with a lognormal distribution but a base value of zero.	For these properties, the non-zero standard deviation of the distribution.
Dist_Param_3	Numeric	Not required (not currently used by TRIM.FaTE)	Optional parameter for distributions (may be used for future applications).

[a] A property that is "required" is one for which the user must enter a data value for any statistics file rows meeting the criteria specified. Note that *all* fields must appear in the statistics data file; however, fields that are not *required* can be left blank.
[b] If this field is left blank, TRIM.FaTE will assume that the statistics for the associated property apply to *all* instances of this property (even if the property has one or more chemicals associated with it).
[c] Note that the CV is calculated by TRIM.FaTE for properties with uniform and triangular distributions; the user can enter "N/A" for the CV field, or leave the field blank, in the statistics file for these properties.

For chemical-specific properties, the user has the option of specifying the statistics for that property as chemical-specific or more generally. This is accomplished by providing or omitting information for the Chemical field in that row of the statistics file (i.e., by including information for certain columns and leaving others blank). For example, consider the following two instances involving the property *initialConcentration_g_per_L*.

< To include a set of statistics for initial concentrations of divalent mercury in surface water, the first four columns of a row in the statistics file would read:

```
initialConcentration_g_per_L, Divalent Mercury, Compartment,
Surface water,
```

followed by the appropriate statistical data values (i.e., minimum, maximum, CV, ...).

< To include a set of statistics pertaining to initial concentrations in all surface water compartments regardless of chemical, the first four columns of that row in the statistics file would read:

```
initialConcentration_g_per_L, ,Compartment,Surface Water,
```

followed by the appropriate statistical data values. Because the second column (i.e., the Chemical field) was left blank, TRIM.FaTE would use the following statistics every time the property *initialConcentration_g_per_L* was used in surface water for *all* chemicals included in the simulation.

Many properties are not chemical-specific by definition (e.g., temperature of surface water, food ingestion rate). The chemical field should be left blank for rows pertaining to these properties. If a property that is selected for inclusion in a sensitivity or Monte Carlo analysis is *not* included in the statistics file, TRIM.FATE will return an error message describing the problem when the sensitivity or Monte Carlo analysis is run.

Note that the distributions currently supported by the TRIM.FaTE uncertainty analysis tool are:

- uniform;
- normal;
- lognormal; and
- triangular.

These are the only allowable values for the Distribution field.

The coefficient of variation (CV) is used by the sensitivity analysis tool only. For parameters assigned a normal or lognormal distribution, the CV must be entered in the statistics file by the user as a numeric value and is equal to the standard deviation divided by the mean. For parameters assigned a uniform distribution, the CV is calculated by TRIM.FaTE and "N/A" can be inserted in the CV column. For a uniform distribution, the CV is calculated internally using the equation:

$$CV = (max - min) / ((max + min) * / 3)$$

The triangular distribution is currently supported by TRIM.FaTE for the Monte Carlo analysis only and is not supported for sensitivity analysis. Therefore, a CV is not required for parameters assigned a triangular distribution, and the user can enter "N/A" in the CV column (or leave the field blank) in the statistics file for these parameters.

USER TIP: SPECIFYING THE MEAN OF A DISTRIBUTION

Note that (with the exception of lognormal distributions with a mean equal to zero) the mean of a property distribution is *not* specified in the statistics file. In the current version of TRIM.FaTE, the base value of a property (i.e., the value assigned to the property in the base case simulation) is used as the mean when that value is needed for sensitivity or Monte Carlo analyses.

For parameters with a uniform distribution, data points for the minimum and maximum allowable value must be included in the statistics data file because TRIM.FaTE uses these values to calculate the CV. A minimum and maximum value can be included for parameters with other distributions if desired, but these values are not required. If minimum and maximum values are included, TRIM.FaTE will use the minimum and maximum values as lower and upper bounds for acceptable sampled values in the Monte Carlo analysis (i.e., if a sampled value is outside of the range defined by the minimum and maximum, it will not be used in the analysis).

The "Dist_Param" columns are included in the statistics file to store additional information about the distribution. Currently, these columns are only used for properties for which a lognormal distribution has been assigned but for which the base value is 0.0. The standard deviation for the lognormal distribution is calculated by TRIM.FaTE by multiplying the property value in the scenario (assumed to be the mean) by the CV. If the property value is 0.0, the distribution would have a mean of 0.0 and a standard deviation of 0.0. By definition, a lognormal distribution cannot have a mean of zero (or any number less than zero). To prevent this from occurring, the user should use the two Dist_Param columns for any properties assigned

a lognormal distribution and a base value of zero. For these properties, the user should specify the non-zero mean in column Dist_Param_1 and the non-zero standard deviation in column Dist_Param_2 and TRIM.FaTE will use these values as needed. The Dist_Param_3 is not currently used by TRIM.FaTE sensitivity and Monte Carlo analyses; however, this data field must still be included in the Statistics file.

For more information regarding the use of the Statistics file in TRIM.FaTE analyses, refer to Module 15, TRIM.FaTE Sensitivity and Monte Carlo Analyses. An example statistics file is presented in Appendix F.

2.3 Time-varying Data Files

Purpose and Use

Time-varying data values can be entered manually by the user for a property value of an object in either a scenario or a library by setting the property form to "Unevenly Time Stepped Real Number" and inputting data by hand for individual time steps. However, for properties whose values change several times during a scenario, it is more efficient to store the data an external data file. In addition, storing time-varying data in external files can facilitate QC of a scenario setup. In TRIM.FaTE test applications, external data files have been used to store and input time-varying data such as meteorological data,[4] seasonal/growing season data (i.e., values for the properties *AllowExchange* and *LitterFallRate*), and water flow data between surface water compartments. Multiple types of data that vary at the same time intervals can be stored in a single data file by storing each unique field in a separate column. For example, all of the water flow data used in a scenario (e.g., bulk water flow rates averaged monthly for multiple lakes modeled in a scenario) could be stored as different columns of data in a single data file. For any time-varying data, the time intervals between changes in property values do not need to be evenly spaced (i.e., uneven time steps are acceptable).

To reference a Time-varying Data file for a specific property, the user sets up a reference in the scenario or library to the location of the time-varying data file for that property. This can be accomplished by completing the following steps:

- For the object of interest, highlight the property to be set equal to time-varying data (e.g., horizontal wind speed, litterfall rate, bulk water flow rate).
- Click the "Form" button at the top of the property window (in either the scenario or library window). In the *Change Property Form* dialog box that appears, select "Unevenly Time Stepped Real Number from File" and click "OK."

[4] Tools have been developed that can assist the user in preparing meteorological data inputs for TRIM.FaTE. For a complete description of the meteorological data required for the algorithms in the current TRIM.FaTE libraries, a list of possible sources of these data, and instructions on how to process the data into time-varying data files for use in TRIM.FaTE scenarios, refer to Module 10, Compiling and Processing Meteorological Data Inputs.

- A value editor window will appear below the main property window in the scenario frame. Specify the file that contains the time-varying data by either:

    (1) Manually entering the file location in the "File Name" window; or

    (2) Clicking the "Browse" button and then selecting the appropriate file using the browser in the *Choose Data File* box that appears.

  The appropriate file location (e.g., c:\models\TRIM\data\datafile.txt) will appear in the "File Name" window.

- In the value editor window, type the appropriate data delimiter (i.e., the character used to separate the data fields within the file) in the small "Delimiter" window and click the "Scan File" button.

- After TRIM.FaTE scans the datafile, the column names of the data fields stored within the data file will appear in the "Column Name:" window. The user can then select the appropriate data type using this window (note that this window functions as a pull-down menu).

- Click the "Store" button at the top of the property editor window to set the property to the values in the file. The file location will appear in the "Value" cell for the selected property in the property window.

By setting up this reference, the actual data values are retained in the data file (external to the TRIM.FaTE scenario) and are referenced by TRIM.FaTE when it encounters that property during any calculations.

Format and Syntax

Time-varying data that TRIM.FaTE reads from an external file must be stored as text in row-column format. The user can insert one or more lines of comments at the beginning of the file. However, there must be at least two lines of data for TRIM.FaTE to determine where the data begin and end along with a line preceding the data that lists the column names. All data fields must be delimited by the same delimiter in the file. Typically, punctuation characters are used as a data delimiters (e.g., comma or a semicolon). The data fields must contain unique column names for each column in the file, and the column names must be listed together on a separate line before the data begins. Data columns must be entered as follows:

- The first column must be called "Date" and contain the date the data becomes valid (i.e., the date that the value should be implemented as the value for that property), in mm/dd/yyyy format (e.g., 10/16/2000).

- The second column must be called "Time" and must contain the time that the data become valid, in hh:mm:ss format (i.e. 13:30:00).

- The third column must be called "Time Zone" and must contain the time zone (e.g, EST).

- The subsequent column(s) should be labeled according to the type of data contained in the file. Note that it is the user's responsibility to ensure that the data are entered in the correct units. Any number of columns can follow with data values in them.

Below is an acceptable file format for a time-varying data file:

```
This is a time-varying data file that can be read by TRIM.FaTE.
Note that these comments appear freely and do not need to be marked with a
slash or other delimiter.
They can even contain a few delimiters(, , ), but not the same number of
delimiters as the data lines below.

Date,Hour,Time Zone,temperature (K),winddirection,windspeed

1/1/1991,00:00:00,EST,281.0,112.7,0.5
1/1/1991,01:00:00,EST,282.1,113.8,0.1
1/1/1991,14:30:00,EST,280.1,113.8,0.1
1/2/1991,00:00:00,EST,283.0,114.8,0.8
1/2/1991,20:00:00,EST,284.0,115.3,0.3
```

TRIM.FaTE determines where the actual data values in a data file begin and end by scanning the file and counting the number of columns on each line (as indicated by the presence of the delimiting character). When TRIM.FaTE identifies several subsequent lines that contain the same number of columns, it assumes that any line with this number of columns is a data line. TRIM.FaTE will then go back to the top of the file and parse the first line that it identified with the correct number of delimiters and assume that this line contains the column names. If there is a line before the column names that contains the same number of delimiters as a data line, TRIM.FaTE will use this line and probably will not read the data correctly. For example, if a comment line is included that contains six commas, and the data are comprised of seven columns delimited by six commas, TRIM.FaTE will use that comment line to label the data columns. To correct this, the user should ensure that same number of delimiters used in the data do *not* appear on a comment line.

Data files can be prepared in a spreadsheet program and then saved in the comma separated variable (CSV) format using the "Save As" function. Note that when a spreadsheet program saves a file in the CSV format, a comma may be inserted as a delimiter between every column. Consequently, the program may add commas to end of the text for any comment lines entered by the user in a single cell at the beginning of the file. This can be problem because the extra commas that appear on the comment lines will cause problems when TRIM.FaTE reads the data file. For example, a data file created as a spreadsheet and then saved in the CSV format could look like this:

```
This is a data file that would confuse the prototype,,,,,
These comments lines were inserted by the user in individual cells at the
beginning of the file and the following commas were added by the program
for the empty columns,,,,,

Date,Hour,Time Zone,temperature (K),winddirection,windspeed

1/1/1991,00:00:00,EST,281,112.7,1.0
1/1/1992,00:00:00,EST,282,113.7,0.5
1/1/1993,00:00:00,EST,283,114.7,0.7
1/1/1994,00:00:00,EST,284,115.7,1.3
```

To correct this problem, the user must manually remove the unwanted commas from the comment lines after the CSV file has been generated but before the data file is entered into TRIM.FaTE. This can be accomplished by opening the CSV file in a text editor and editing it.

If the data file only has one line of data, TRIM.FaTE will be unable to determine where the data begin and end. Too few lines of data will be present for TRIM.FaTE to determine the number of delimiters on a data line – at least two lines of data must be present for TRIM.FaTE to determine the number of delimiters. An input data file cannot be used if the file contains only one line of data.

3. EXPORT FILES

Export files can be created based on a TRIM.FaTE library or a TRIM.FaTE scenario. There are several different types of export files that can be produced for each. These exports provide the user with additional information about the configuration of a TRIM.FaTE library or scenario. The options for exporting library and scenario diagnostic outputs are described below.

3.1 Library Exports

There are three different types of library exports that can be produced by TRIM.FaTE: Object Exports, Property Exports, and Property Type Exports. Each of these types of exports, and the steps involved in creating them, is described below.

3.1.1 Object Export

An Object Export is a text file that contains all of the information in the library in a text format that can be edited in a text editor and re-imported into TRIM (see Section 1.1.1 for more information about importing Object Import files). This export can be useful for making significant changes to a library because the user can use text editor functions, such as search and replace, to make changes to the library.

The Object Export is created by completing the following steps.

(1) Select “Export” from the file menu of the FaTE Library window.

(2) Select “Object Exporter” in the Choose Exporter window and click “OK.”

(3) Specify a name and location for the export file in the File Browser and click "Export Objects."

3.1.2 Property Export

A Property Export is a semicolon-delimited text file that contains the complete list of properties, and their corresponding values and formulas, that are present in the library. This export can be useful for sorting and reviewing the properties in the library in a spreadsheet program. The defaults (when present) for the algorithm, chemical, compartment type, and source properties used to create a scenario are all included in this file.

The Property Export is created by completing the following steps.

(1) Select "Export" from the file menu of the FaTE Library window.

(2) Select "Property Exporter" in the Choose Exporter window and click "OK."

(3) Select the property type(s) (e.g., All, Real Number) to export and click "OK."

(4) Specify a name and location for the export file in the File Browser and click "Export Objects."

3.1.3 Property Type Export

A Property Type Export is a semicolon-delimited text file that contains the complete list of property types in the library. This export can be useful for reviewing and editing the descriptions and units for property types in the library in a spreadsheet program. Edited Property Type Export files can then be imported into a library as described in Section 1.1.2. The Property Export is created by completing the following steps.

(1) Select "Export" from the file menu of the TRIM.FaTE Library window.

(2) Select "Property Type Exporter" in the Choose Exporter window and click "OK."

(3) Specify a name and location for the export file in the File Browser and click "Export Objects."

3.2 Scenario Exports

There are five types of scenario exports that can be produced by TRIM.FaTE:

- Scenario Property Values and Results at a Time Point Exports;
- Property Exports;
- Outdoor Environment Exports;
- Link Exports; and
- Scenario Configuration and Results Export in MySQL Format.

Each of these types of exports, and the steps involved in creating them, is described below.

3.2.1 Scenario Property Values and Results at a Time Point

TRIM.FaTE can create a series of HTML files containing summarized results from a simulation as well as the values of all of the properties in the scenario (e.g., scenario properties, compartment properties, link properties) at an user-specified time point. These files, collectively referred to as the "HTML Export," can be created before or after a running a simulation. When performed after a simulation, this export can be particularly useful in evaluating model results because the user can see the transfer factors and fluxes between compartments. A detailed description of this export, as well as the steps required to create it, is provided in Section 1.3 of Module 14, Simulation Results and Analyses.

3.2.2 Property Export

The scenario Property Export is a semicolon-delimited text file that contains the property values for the scenario, including any spatial variation in property values/formulas across the scenario compartments. It also includes the link, scenario (e.g., start time and end time of run, export options), sink, and volume element properties that have been defined in the project scenario. This export can be useful for documenting the properties in a scenario and for sorting and reviewing these properties in a spreadsheet program.

The scenario Property Export can be created before or after a simulation by completing the following steps.

(1) Select "Export" from the file menu of the FaTE Scenario window.

(2) Select "Property Exporter" in the Choose Exporter window and click "OK."

(3) Select the property type(s) (e.g., All, Real Number) to export and click "OK."

(4) Specify a name and location for the export file in the File Browser and click "Export Objects."

Alternatively, to help in documenting a simulation, the user can have TRIM.FaTE automatically perform a Property Export before starting a simulation by setting the scenario property *exportPropertiesBeforeRun* to "true."

3.2.3 Outdoor Environment Export

The Outdoor Environment Export is a semicolon-delimited text file that contains the list of volume elements, compartments, and links defined for the project scenario, along with the algorithms associated with each link (collectively, these components comprise the "outdoor environment" of the scenario). This export can be useful for documenting the volume elements, compartments, and links in a scenario in a format that can be viewed in a spreadsheet program. The Outdoor Environment Export can be created before or after a simulation by completing the following steps.

(1) Select “Export” from the file menu of the FaTE Scenario window.

(2) Select “Outdoor Environment Exporter” in the Choose Exporter window and click “OK.”

(3) Specify a name and location for the export file in the File Browser and click “Export Objects.”

Alternatively, to help in documenting a model run, the user can have TRIM.FaTE automatically perform an Outdoor Environment Export before starting a simulation by setting the scenario property *exportOutdoorEnvironmentBeforeRun* to “true.”

3.2.4 Link Export

The Link Export is a text file that contains a list of all of the links and their associated algorithms in a scenario. This export can be useful for documenting the links and their associated algorithms in a scenario in a format that can be viewed in a spreadsheet program. The Link Export can be created before or after a simulation by completing the following steps.

(1) Select “Export” from the file menu of the FaTE Scenario window.

(2) Select “Link Exporter” in the Choose Exporter window and click “OK.”

(3) Specify a name and location for the export file in the File Browser and click “Export Objects.”

3.2.5 Scenario Configuration and Results in MySQL Format

The results of a TRIM.FaTE simulation, including the compartment results, deposition estimates, and details about the configuration of the scenario, can be exported in MySQL database format. This export, referred to as the “MySQL Export,” consists of a Property Export, an Outdoor Environment Export, and a Results Export (if simulation has completed successfully) for the simulation written in MySQL database format. This export is used for providing the necessary information about a TRIM.FaTE simulation to other TRIM modules, although users familiar with MySQL may find it useful. A detailed description of this export, as well as the steps required to create it, is provided in Section 1.4 of Module 14, Simulation Results and Analyses.

APPENDIX A
EXAMPLE OBJECT IMPORT FILE

```
// Example Object Import file
// This file could be used to add additional objects to an existing library

Version: 1

Ptype: AirDensity
DataType: FloatingPoint
DefaultValue: 2.5
Description: Density of particles in air
Units: g/m3
Max: 5.0
Min: 0.0

CompCompartment: Agriculture - General
Property: acceptableAbiotic
Value: Soil | Surface Soil | Surface Soil - Default
Property: concentrationOutputFactor
Value: 1.0
Property: isBiotic
Value: false
Component: Leaf - Agriculture - General
Component: Leaf Particle - Agriculture - General
Component: Root - Agriculture - General
Component: Stem - Agriculture - General

Algorithm: Ingestion of Worm by American Robin
Property: category
Value: Ingestion
Property: chemicalCategory
Value: All
Property: doesTransformChemical
Value: false
Property: doesTransportChemical
Value: true
Property: enabled
Value: true
Property: isDefaultForCategory
Value: true
Property: receivingChemicalName
Value: ReplaceMe
Property: receivingCompartmentCategory
Value: Bird | American Robin
Property: sendingChemicalName
Value: ReplaceMe
Property: sendingCompartmentCategory
Value: Worm | Worm - Default
Property: transferFactor
Form: Formula
Value: [[ReceivingCompartment.PopulationSize * ReceivingCompartment.BW *
TheLink.FractionSpecificcompartmentDiet *
ReceivingCompartment.FractionDietWorm *
ReceivingCompartment.FoodIngestionRate *
ReceivingCompartment.Chemical.AssimilationEfficiencyFromWorms /
SendingCompartment.TotalMass]]
```

APPENDIX B
EXAMPLE COMPARTMENT IMPORT FILE

```
Version:1
VolumeElement:SW_Penob
Compartment:Common Loon
Compartment:Mallard
Compartment:Macrophyte
Compartment:Water Column Carnivore
Compartment:Water Column Herbivore
Compartment:Water Column Omnivore
VolumeElement:Sed_Penob
Compartment:Benthic Carnivore
Compartment:Benthic Invertebrate
Compartment:Benthic Omnivore
VolumeElement:SurfSoil_E1
CompositeCompartment:Deciduous Forest
Compartment:Red-tailed hawk
Compartment:Short-tailed Shrew
Compartment:Raccoon
Compartment:Mouse
Compartment:Bald Eagle
Compartment:Mink
Compartment:Tree swallow
Compartment:White-tailed Deer
Compartment:Long-tailed Weasel
Compartment:Black-capped Chickadee
Compartment:Arthropod
```

APPENDIX C
EXAMPLE PROPERTY IMPORT FILE

```
Version: 1
Scenario: LeadSmelting
Run: BaseRun

Scenario: LeadSmelting
Property: averageResultsFiles
Value: true
Property: averagingInterval
Value: monthly
Property: outputDir
Value: C:\\TRIM\\data\\LeadSmelting\\LeadSmelting

VolumeElement: Air_ESE2
VolumeElement: Air_SSE2
VolumeElement: Air_SSW2
VolumeElement: Air_WSW2
VolumeElement: Air_WNW2
VolumeElement: Air_NNW2
Property: top
Form: InputFromFile
Value: C:\\TRIM\\data\\LeadSmelting\\MetData.csv, Mixing Height (m), ","

Compartment: Surface water in SW_Silver
Property: Flushes_per_year
Value: 0.0
Property: CurrentVelocity
Value: 0.0

NewLink:
SendingCompartment: Red-tailed Hawk in SurfSoil_NW2
ReceivingCompartment: Surface water in SW_Silver
ReciprocalLink: Surface water in SW_Silver to Red-tailed Hawk in SurfSoil_NW2
Algorithm: Default

NewLink:
SendingCompartment: Red-tailed Hawk in SurfSoil_NW2
ReceivingCompartment: Chickadee in SurfSoil_NNE2
ReciprocalLink: Chickadee in SurfSoil_NNE2 to Red-tailed Hawk in SurfSoil_NW2
Algorithm: Default
```

APPENDIX D
EXAMPLE RUN IMPORT FILE

```
Version: 1
Scenario: LeadSmelting
NumberOfRuns: 3
Run: HighChloride

//Test run with higher chloride concentration in Silver Lake

Scenario: LeadSmelting
Property: averageResultsFiles
Value: true
Property: averagingInterval
Value: monthly
Property: outputDir
Value: C:\\TRIM\\data\\LeadSmelting\\LeadSmelting\\HighChlorideResults

VolumeElement: Air_ESE2
VolumeElement: Air_SSE2
VolumeElement: Air_SSW2
VolumeElement: Air_WSW2
VolumeElement: Air_WNW2
VolumeElement: Air_NNW2
Property: top
Form: InputFromFile
Value: C:\\TRIM\\data\\LeadSmelting\\MetData.csv, Mixing Height (m), ","

Compartment: Surface water in SW_Silver
Property: Flushes_per_year
Value: 0.0
Property: CurrentVelocity
Value: 0.0
Property: ChlorideConcentration_mg_L
Value: 7500.0

NewLink:
SendingCompartment: Red-tailed Hawk in SurfSoil_NW2
ReceivingCompartment: Surface water in SW_Silver
ReciprocalLink: Surface water in SW_Silver to Red-tailed Hawk in SurfSoil_NW2
Algorithm: Default

NewLink:
SendingCompartment: Red-tailed Hawk in SurfSoil_NW2
ReceivingCompartment: Chickadee in SurfSoil_NNE2
ReciprocalLink: Chickadee in SurfSoil_NNE2 to Red-tailed Hawk in SurfSoil_NW2
Algorithm: Default

Run: LowChloride

//  Test run with low chloride concentration in Silver Lake

Scenario: LeadSmelting
Property: averageResultsFiles
Value: true
Property: averagingInterval
Value: monthly
Property: outputDir
Value: C:\\TRIM\\data\\LeadSmelting\\LeadSmelting\\LowChlorideResults

VolumeElement: Air_ESE2
VolumeElement: Air_SSE2
VolumeElement: Air_SSW2
```

```
VolumeElement: Air_WSW2
VolumeElement: Air_WNW2
VolumeElement: Air_NNW2
Property: top
Form: InputFromFile
Value: C:\\TRIM\\data\\LeadSmelting\\MetData.csv, Mixing Height (m), ","

Compartment: Surface water in SW_Silver
Property: Flushes_per_year
Value: 0.0
Property: CurrentVelocity
Value: 0.0
Property: ChlorideConcentration_mg_L
Value: 0.005

NewLink:
SendingCompartment: Red-tailed Hawk in SurfSoil_NW2
ReceivingCompartment: Surface water in SW_Silver
ReciprocalLink: Surface water in SW_Silver to Red-tailed Hawk in SurfSoil_NW2
Algorithm: Default

NewLink:
SendingCompartment: Red-tailed Hawk in SurfSoil_NW2
ReceivingCompartment: Chickadee in SurfSoil_NNE2
ReciprocalLink: Chickadee in SurfSoil_NNE2 to Red-tailed Hawk in SurfSoil_NW2
Algorithm: Default

Run: Chickadee

/*Test run with additional link from Red-tailed Hawk in SurfSoil_SE3 to
Chickadee in NNE2*/

Scenario: LeadSmelting
Property: averageResultsFiles
Value: true
Property: averagingInterval
Value: monthly
Property: outputDir
Value: C:\\TRIM\\data\\LeadSmelting\\LeadSmelting\\Chickadee

VolumeElement: Air_ESE2
VolumeElement: Air_SSE2
VolumeElement: Air_SSW2
VolumeElement: Air_WSW2
VolumeElement: Air_WNW2
VolumeElement: Air_NNW2
Property: top
Form: InputFromFile
Value: C:\\TRIM\\data\\LeadSmelting\\MetData.csv, Mixing Height (m), ","

Compartment: Surface water in SW_Silver
Property: Flushes_per_year
Value: 0.0
Property: CurrentVelocity
Value: 0.0

NewLink:
SendingCompartment: Red-tailed Hawk in SurfSoil_NW2
ReceivingCompartment: Surface water in SW_Silver
ReciprocalLink: Surface water in SW_Silver to Red-tailed Hawk in SurfSoil_NW2
Algorithm: Default

NewLink:
SendingCompartment: Red-tailed Hawk in SurfSoil_NW2
ReceivingCompartment: Chickadee in SurfSoil_NNE2
ReciprocalLink: Chickadee in SurfSoil_NNE2 to Red-tailed Hawk in SurfSoil_NW2
```

```
Algorithm: Default

NewLink:
SendingCompartment: Red-tailed Hawk in SurfSoil_SE3
ReceivingCompartment: Chickadee in SurfSoil_NNE2
ReciprocalLink: Chickadee in SurfSoil_NNE2 to Red-tailed Hawk in SurfSoil_SE3
Algorithm: Default
```

APPENDIX E
EXAMPLE GIS OVERLAY FILES

E.1 Sample Overlay File Containing Labeled Points

```
PROJECTION:LatLon
ELLIPSOID:WGS 1984
CenterLongitude:-68.7

BEGIN_POINTS
SHAPE: circle
FONT: Arial
FONTSTYLE: Italic
FONTSIZE: 14
# This will illustrate the orientation feature.
ORIENTATION:north
north, -68.755, 44.72
SHAPE: square
SYMBOLCOLOR: red
ORIENTATION:northeast
northeast, -68.743, 44.714
SHAPE: triangle
SYMBOLCOLOR: orange
ORIENTATION:east
east, -68.735, 44.705
SHAPE: diamond
SYMBOLCOLOR: green
ORIENTATION:southeast
southeast, -68.743, 44.695
SHAPE: cross
SYMBOLCOLOR: magenta
ORIENTATION:south
south, -68.755, 44.69
SHAPE: star
SYMBOLCOLOR: pink
ORIENTATION:southwest
southwest, -68.767, 44.695
SHAPE: circle
SYMBOLCOLOR: cyan
ORIENTATION:west
west, -68.775, 44.705
SYMBOLCOLOR: gray
ORIENTATION:northwest
northwest, -68.767, 44.714
ORIENTATION:center
SHAPE:none
center, -68.755, 44.705

END_POINTS
```

E.2 Sample Overlay File Containing Polygons

```
PROJECTION:LatLon
ELLIPSOID:WGS 1984
CenterLongitude:-68.7

BEGIN_POLYGONS
TYPE:filled
FILLCOLOR:pink
LINETHICKNESS:2
LINESTYLE: solid
NUMPOINTS:4
-68.75, 44.70
-68.80, 44.70
-68.80, 44.75
-68.75, 44.75
TYPE:filled
FILLCOLOR:orange
LINETHICKNESS:5
LINECOLOR:magenta
NUMPOINTS:5
-68.82, 44.76
-68.80, 44.75
-68.81, 44.73
-68.83, 44.73
-68.84, 44.75
LINECOLOR:Black
TYPE:outline
NUMPOINTS:4
-68.81, 44.73
-68.83, 44.73
-68.81, 44.68
-68.83, 44.68
TYPE:line
LINECOLOR: red
LINESTYLE: dashed
NUMPOINTS:2
-68.80, 44.68
-68.70, 44.72
END_POLYGONS
```

APPENDIX F
EXAMPLE STATISTICS FILE

```
Property,Chemical,ObjectType,ObjectName,Minimum,Maximum,Units,CV,Distribution,Dist_param_1,Dist_param_2,Dist_param_3
AlgaeGrowRate, ,Compartment,Surface water, , ,/day,3.00E-01,lognormal,,,
AlgaeRadius, ,Compartment,Surface water,0, ,um,3.00E-01,lognormal,,,
AlgaeUptakeRate,Elemental Mercury,Compartment,Surface water,0, ,nmol/[um2 - day - nM],3.00E-01,lognormal,,,
AlgaeUptakeRate,Divalent Mercury,Compartment,Surface water,0, ,nmol/[um2 - day - nM], 3.00E-01, lognormal,,,
AlgaeUptakeRate,MethylMercury,Compartment,Surface water,0, ,nmol/[um2 - day - nM],3.00E-01,lognormal,,,AlgaeWaterContent,
,Compartment,Surface water,0,1,N/A,3.00E-01,triangular,,,
```