The INTERLIS 2 reader and writer module (ili2fme) provides the Feature Manipulation Engine (FME) with access to INTERLIS 2 transfer files.

ili2fme is licensed under the LGPL (Lesser GNU Public License).

ili2fme includes software developed by The Apache Software Foundation (http://www.apache.org/).

ili2fme is in stable/production state.

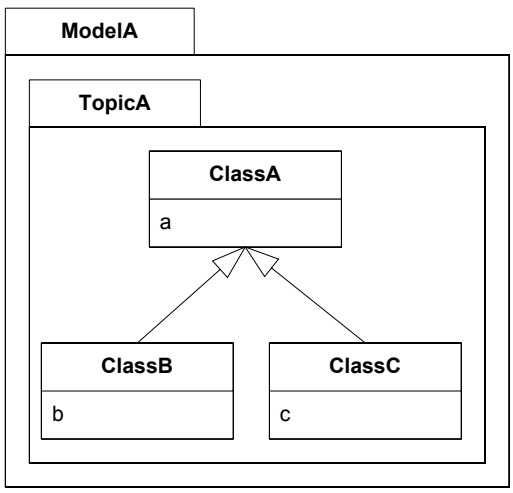
The current version of ili2fme can be found at http://www.eisenhutinformatik.ch/interlis/ili2fme/.

This chapter assumes you are familiar with FME and the INTERLIS 2 format. For more information about FME, please read the FME documentation. For more information about INTERLIS 2, go to http://www.interlis.ch.

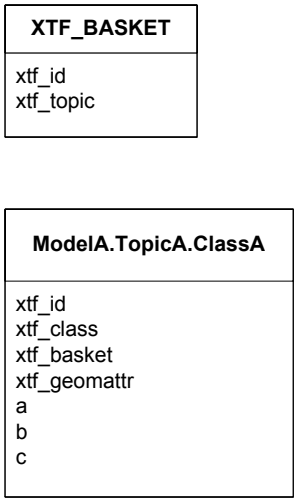
Please send comments about ili2fme to ce@eisenhutinformatik.ch.

Overview

Features read from a INTERLIS file consist of a series of attribute values. They may have no geometry. The attribute names are as defined in the INTERLIS model. The feature type of each INTERLIS feature is the qualified INTERLIS name (for INTERLIS 2 the base class name, for INTERLIS 1 the table name). The mapping of the inheritance hierarchy is done with a super type strategy. Attributes of non-root classes are shifted to the root, as illustrated by the following figure:



INTERLIS model



FME feature schema

INTERLIS 2 Quick Facts

Format Type Identifier ch.interlis.ili2fme.Ili2fme

Reader/Writer	Both
Dataset Type	File
Feature Type	Class name
Typical File Extensions	.xtf, .xml, .itf, .ili
Automated Translation Support	Automated reading
User-Defined Attributes	Yes
Coordinate System Support	No
Generic Color Support	No
Spatial Index	Never
Schema Required	Yes
Transaction Support	No
Geometry Type Attribute	xtf_geomtype
Geometry Support	

Geometry	Supported	Geometry	Supported
aggregate	no	polygon	yes
circles	no	donut polygon	yes
circular arc	no	line	yes
elliptical arc	no	point	yes
ellipses	no	text	no
none	yes	3D	yes

Reader Overview

FME considers a INTERLIS transfer file to be a collection of features. The feature types are determined by scanning the transfer file and then reading the appropriate INTERLIS model/schema files. The model files have the extension .ili and should be located in the same folder as the transfer file and/or in the folder ${FME}/plugins/interlis2/ilimodels. A transfer file may need multiple model files. There are no DEF lines required.

Reader Keywords

The following table lists the keywords processed by the INTERLIS 2 reader. The table shows only the suffixes prefixed by the current <ReaderKeyword> in a mapping file. By default, the <ReaderKeyword> for the INTERLIS 2 reader is `ch.interlis.ili2fme.Ili2fme`.

Keyword Suffix	*Value*
Models	Required INTERLIS-Models to read the dataset (without extension .ili; separated by semicolons ';'). Or the special value XTF (the value XTF has special meaning

	independently of the extension of the data file), in which case the models are determined by inspecting the transfer file. Default Value: XTF
ModelDir	Folder containing the .ili-Files. These files are scanned for INTERLIS-Models. You may use %XTF_DIR as placeholder for the directory of the data file that you will read. Multiple directories may be separated by semicolons ';'. Default Value: unset/empty

Writer Overview

The INTERLIS 2 writer module stores features into a INTERLIS transfer file. The Models keyword (see below) must be set to the name of the output model/schema. The appropriate INTERLIS model/schema files are then read before the output starts. The model files have the extension .ili and should be located the same folder as the transfer file and/or in the folder ${FME}/plugins/interlis2/ilimodels. A transfer file may need multiple model files. There are no DEF lines required.

The appropriate feature types are expected by the writer, as if the same model would have been read by the INTERLIS 2 reader.

Writer Keywords

The following table lists the keywords processed by the INTERLIS 2 writer. The table shows only the suffixes which will be prefixed by the current `<WriterKeyword>` in a mapping file. By default, the `<WriterKeyword>` for the INTERLIS 2 writer is `ch.interlis.ili2fme.Ili2fme`.

Keyword Suffix	*Value*
Models	Specifies the INTERLIS-Models (without extension .ili) that the written dataset should adhere to(separated by semicolons ';'). Default Value: no default value.
ModelDir	Folder containing the .ili-Files. These files are scanned for INTERLIS-Models. You may use %XTF_DIR as placeholder for the directory of the data file that you will write. Multiple directories may be separated by semicolons ';'. Default Value: unset/empty

Feature Representation

In addition to the generic FME feature attributes that FME Workbench adds to all features (see *About Feature Attributes* on page 1), this format adds the format-specific attributes described in this section.

Attribute	*Description*
xtf_id	Value of the TID XML-attribute out of the INTERLIS 2 transfer file. Unique across all feature types.
xtf_class	Qualified name of the INTERLIS 2 class name. This is different from the feature type name in the case of non base classes. In the figure above would ModelA.TopicA.ClassB be a possible value be.
xtf_basket	Value of the BID XML-attribute out of the INTERLIS transfer file. May be used as foreign key to a feature of the feature type XTF_BASKET (see below). On writing, this may be used to write multiple baskets of the same topic.
xtf_geomattr	Name of the geometry attribute read. A INTERLIS 2 class may define multiple geometry attributes. Only one is read by this reader (see also the section on Limitations).

The reader creates an additional feature type XTF_BASKET. And the writer expects this feature type as well. This technical feature type has two attributes: xtf_id and xtf_topic.

Attribute	*Description*
xtf_id	For each basket in the INTERLIS 2 transfer file, the value of the BID XML-attribute.
xtf_topic	Qualified name of the INTERLIS 2 topic name. In the figure above would ModelA.TopicA be a possible value.

Features written to the INTERLIS transfer file are expected to have the same structure, as they would have had when read.

Limitations

- multiple geometries per object/feature
- geometries in structs
- arcs
- custom line forms
- line attributes
- same attribute name in different classes with the same base class
- incremental transfer
- recursive structure attributes

Installation

Requirements

For the current version of ili2fme, you will need a JRE (Java Runtime Environment) installed on your system, version 1.4.1 or later.

The JRE (Java Runtime Environment) can be downloaded for free from the Website http://www.java.com/.
ili2fme was tested with FME Version 2006 GB (20060620 - Build 2651) and 2004 ICE 3 (20041108 - Build 1378).

Files

To install ili2fme, choose a directory and extract the distribution file there.

Copy the files and subdirectories of "${ili2fme}/FME Suite" to your FME directory.

FME requires that the path to the directory containing the file jvm.dll is on the PATH (Typically this DLL is in the folder c:/program files/java/j2re/bin/client/). Edit the PATH variable (select Windows Start menu::Control Panel::System::Advanced::Envirnoment Variables), if this is not the case. DO NOT MOVE/COPY the jvm.dll to another directory!

Add your standard INTERLIS models to the directory "${FME}/plugins/interlis2/ilimodels".

At runtime, ili2fme requires the following files:

```
${FME}/xerces-c_2_6-interlis2.dll
${FME}/plugins/iom_fme.dll
${FME}/plugins/ili2c.jar
${FME}/plugins/ili2fme.jar
${FME}/metafile/ch.interlis.ili2fme.Ili2fme.fme
${FME}/formatsinfo/interlis2.db
```

Configuration

To use ili2fme with the FME Universal Viewer, FME requires you to set an

environment variable: FME_VIEWER_THREADING=SINGLE (FIXME: same place as PATH).

ili2fme doesn't use or require any windows registry entries or user settings file.

On FME version before 2006GB:
Add the contents of the file "${ili2fme}/formats/formats_db-preFME2006GB.txt" to the file "${FME}/formats.db".
Add the contents of the file "${ili2fme}/formats/gallery_db-preFME2006GB.txt" to the file "${FME}/gallery.db".

How to migrate/update an existing ili2fme installation

Just copy the files and subdirectories of the new "${ili2fme}/FME Suite" to your FME directory.

FAQ

Usage

I am getting the following error: "missing model Roads"

In the folder of your data-file or your folder ${FME}/plugins/interlis2/ilimodels there is no .ili-file containing a "MODEL Roads". Move the file Roads.ili to the folder of your data-file or the folder ${FME}/plugins/interlis2/ilimodels.

My destination format is INTERLIS 2 and I'm getting the follwing error: "model name not specified"

You must change the Parameter "Models" to the name of the INTERLIS model (without extension .ili) that you intend to write (on the Destination Dataset).

My destination format is INTERLIS 2 and I'm getting the follwing error: "missing mandatory attribute xtf_class."

The appropriate feature types are expected by the writer, as if the same model would have been read by the INTERLIS 2 reader. That means: Every feature type must have the Attributes xtf_id, xtf_class, xtf_basket, xtf_geomattr. There must be a feature type XTF_BASKET with attributes xtf_id and xtf_topic.

I have an INTERLIS model "Roads.ili". Should I place into the folder ${FME}/plugins/interlis2/ilimodels?

Yes, if you read or write data according to that model more than once. (ili2fme will also look in the folder of your data-file for INTERLIS models.)

Is the ordering of the model names as a value of the FME-keyword "Ili2fme_Models" significant?

No, any ordering will do.

If a model imports other models (like "Units" or "CoordSys"), should I name all models as value of the FME-keyword "Ili2fme_Models"?

No, but all required models (including indirectly imported ones), all required .ili-files, should be in the folder of your data-file or the folder $(FME)/plugins/interlis2/ilimodels.

If a model imports other models (like "Units" or "CoordSys"), which one should I name as value of the FME-keyword "Ili2fme_Models"?

Use the most specific one (the one that imports directly or indirectly all the other ones). The imported models will be used automatically.

If a model extends another one, which one should I name as value of the FME-keyword "Ili2fme_Models", the base model or the extended one?

Use the extended one. The base model will be used automatically.

I would like to convert to a particular INTERLIS model. How can I import the feature types?

Import the INTERLIS model file (file with the extension .ili), instead of a INTERLIS data file. (You have to change the Filetype in the file selector dialog to "All Files" to see the ili-files.)

Mapping

How to map XTF_GEOMATTR, XTF_ID, XTF_CLASS, XTF_BASKET if INTERLIS 2 is the destination format?

XTF_GEOMATTR is the name of the INTERLIS attribute that will hold the geometry of the Feature. Typically a constant like "Geometry" (the actual value depends on your INTERLIS model).
XTF_ID is the XML attribute TID and should be unique across all feature types. Typically the value of the primary key of the source feature.
XTF_CLASS the qualified name of the destination INTERLIS-class. Typically a constant like "ModelName.TopicName.ClassName" (the actual value depends on your INTERLIS model).
XTF_BASKET is the foreign key of a feature of type XTF_BASKET.

If an attribute is of type enumeration (like „color: (red,green,blue);": Is it possible to get the values (0,1,2,...) instead of the resolved names?

No. In INTERLIS 2 the resolved name is the value. In INTERLIS 2 there is no mapping of an enumeration to a numeric.

How to specify at export the kind of transfer (FULL, INITIAL, UPDATE) and the kind of feature operation (INSERT, UPDATE, DELETE)?

These values are required for incremental transfer. The current version of ili2fme doesn't support incremental transfer.

How are foreign keys mapped?

The value of the REF XML-attribute of the role (association end) gets the property value of the feature, that contains the role.

How are 1-1 associations mapped?

Like defined by the INTERLIS 2-encoding rules. The end class of the second role (association end) gets the property with the reference/foreign key. The property gets the name of the first role.

How are BAG/LIST-attributes mapped?

BAG/LIST-attributes are mapped as list attribute.

How is inheritance mapped?

ili2fme uses a super class strategy. All attributes of specialized classes are shifted to the root class of the inheritance tree.

My INTERLIS model contains a lot of classes, but in FME, I see only a few of them as feature types. Why?

ili2fme uses a super type strategy to map the inheritance tree of the INTERLIS classes. Only root classes in INTERLIS become feature types in FME.

Configuartion

I've modified the file formats.db. Why does FME still report: "No Reader named `ch.interlis.ili2fme.Ili2fme' is available in this FME version"?

- Maybe jvm.dll is not found by FME.
- Maybe jvm.dll is found, but in the wrong directory.

FME requires that the path to the directory containing the file jvm.dll is on the PATH (Typically this DLL is in the folder c:/program files/java/j2re/bin/client/). Edit the PATH variable, if this is not the case. (Do not move jvm.dll!)
The Java Runtime Environment (JRE) requires that jvm.dll is in the directory as installed by the JRE. Therefore you can not move the jvm.dll to another directory.
Check that no directory set in the PATH variable before the entry for the JRE location, contains a jvm.dll.

Changes

ili2fme 1.2.0 (2006-05-09)

- version info added to DLL and jar
- log error msg to fme if iom_java.dll can not be loaded
- xtf-reader: log model name if extracted from xtf
- xtf-reader: do just schema read (if ili-file given instead of xtf-file)
- bug xtf-writer: doesn't write struct elements
- bug: don't free iom native objects from gc thread

ili2fme 2006-03-29

- xtf-writer: improved error message if XTF_BASKET features missing
- xtf-writer: improved error message if xtf_id,xtf_basket,xtf_class attributes in a feature type missing
- xtf-writer: improved error message if MODELS and MODEL_DIR not present in the mapping file
- xtf-writer: test if attribute value exists before getting it
- xtf-reader: create feature types of all directly or indirectly used ili-models
- apply subclass strategy to CLASSes of model INTERLIS (e.g. METAOBJECT)
- bug: doesn't work with universal translator
- bug: work around FME to handle UNC-filename (translate "//share/file" back to "\\share/file")

License

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful. (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.) These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it. Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library. In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices. Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy. This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange. If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation,

is not a derivative work of the Library, and therefore falls outside the scope of this License. However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables. When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law. If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.) Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications. You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy. For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself

accompanies the executable.It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library. If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances. It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice. This section is intended

to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS