GRANTA MI™ version 11

GRANTA MI FEA Exporter Author Guide



GRANTA MI[™] is the leading system for materials information management in engineering organizations. It enables you to control, analyze, and securely share critical corporate data on materials and processes, managing the materials information lifecycle.

GRANTA MI[™] is the leading system for materials information management in engineering organizations. It enables you to control, analyze, and securely share critical corporate data on materials and processes, managing the materials information lifecycle.

www.grantadesign.com

© Granta Design 2018 All rights reserved

CES Selector and GRANTA MI are trademarks of Granta Design Ltd. For other Granta product trademarks, see www.grantadesign.com/smallprint.htm

CATIA[®], ENOVIA[®], SIMULIA[®], SolidWorks[®], and Abaqus/CAE[®] are registered trademarks of Dassault Systèmes or its subsidiaries in the United States or other countries.

Microsoft[®], Excel[®], PowerPoint[®], Internet Explorer[®], SQL Server[®], Windows[®], and Windows Server[®] are registered trademarks of Microsoft Corporation or its subsidiaries in the United States or other countries.

Granta Design Ltd. makes reasonable efforts to explicitly acknowledge all trademarks cited in our literature or on our website. If you would like us to add or alter an acknowledgement, please <u>contact us</u>.

Release notes, documentation, and Knowledge Articles for the current and all previous GRANTA MI releases are all available on the Granta Support website. Go to <u>www.grantadesign.com</u> and click SIGN IN to log into your My Granta page, then click on **Documentation**.

We welcome your feedback on this document. Please let us know if anything is unclear, if you spot an error, or have an idea for new content, by emailing <u>docs@grantadesign.com</u>.

Document version: MI11/03 Published: April 2018

Table of Contents

1 Introduction			5
	1.1	Scope of this document	5
	1.2	Other useful resources	5
2	Expor	rter files	7
	2.1	Developing your own exporters	7
	2.2	Exporter configuration file parsing and checking	
3	Expor	rter config options	9
4	Expor	rter details option	11
	4.1	Package, Model, Description	
	4.2	Applicability	
	4.3	Output file options	
	4.4	Export file encoding	
	4.5	Other options	
5	Datal	base export options	15
	5.1	Database Key and GUID	
	5.2	Unit systems	
	5.3	Tables	
6	Table	export options	16
	6.1	Exporting all attribute data from a table	
	6.2	Exporting table version control information – <versioncontrol></versioncontrol>	
	6.3	Exporting table data quality information - <dataquality></dataquality>	
	6.4	Exporting access control information - <accesscontrol></accesscontrol>	
	6.5	Exporting parameter information – <fullparameterdetails></fullparameterdetails>	
	6.6	Exporting attribute data - <dbattributes></dbattributes>	
	6.7	Exporting user input – <userdata></userdata>	
7	Attrib	oute export options	20
	7.1	Identifying the attribute - <dbattribute></dbattribute>	
	7.2	Using aliases	
	7.3	Exporting meta-attributes	
	7.4	Exporting functional data	
	7.5	Exporting Embedded Equations and Logic (EEL) data	

8	Specif	Specifying the transforms		
9	Adding (importing) an exporter to a database			
	9.1	To add a new exporter	. 29	
10	FEA E	xport features in MI applications	. 30	
	10.1	FEA exporters in MI:Viewer	. 30	
	10.2	FEA exporters in MI:Explore	. 31	
11	Debug	gging exporters	. 32	
	11.1	Parsing errors	. 32	
	11.2	XSLT transforms	. 32	
	11.3	Missing attributes	. 32	

1 Introduction

Granta FEA exporters allow you to export data from a GRANTA MI database in a format that can be imported into a number of different Finite Element Analysis (FEA) packages.

All data types except File and Picture data can be exported, including Functional data (grid and series), and Equations and Logic data. Quality ratings may also be exported.

1.1 Scope of this document

The process of exporting data from a GRANTA MI database and importing it into a Finite Element Analysis (FEA) package can be split into two stages;

- 1. Extracting the data from the database. Data is output as XML that conforms to a published schema (this XML is referred to as "initial XML").
- 2. Using XSLT (Extensible Stylesheet Language Transformations) to manipulate that data into a format that the FEA package can read.

This document describes how to develop an exporter configuration that will produce the initial XML containing the required data from your GRANTA MI database. It does not cover how to write XSLT to transform the initial XML data into the format required by the target FEA package.

The XSLT specification is large and complex, and for information on how to write XSLT you should consult XSLT reference documentation or online resources, for example the <u>XSLT Tutorial at</u> <u>w3schools.com</u> (external link).

1.2 Other useful resources

Some additional files that can be used alongside this document are provided in *FEA_Schema.zip*:

- **ConfigSchema.xsd** defines the schema of exporter configuration files (.exp files). This file is includes comprehensive documentation on all the XML elements and attributes included in the schema, and is a useful source of information when you are modifying exporters, or developing new exporters.
- InitialXml.xsd and matml31.xsd describe the Initial XML output schema, that is, the XML that will be transformed by customer-written XSLT.



The *FEA_Schema.zip* file is included in the GRANTA MI reference documentation set and can be downloaded from MI:Viewer by clicking **Help > Reference Documentation**:

		GRANTA MI R	eference Documentation
		GRANTA MI 11.0	Administrator Guide.pdf
		GRANTA MI 11.0	Automatic Link Creator CLI.pdf
	About M	:Viewer	Bulk Data Importer Guide.pdf
Adv	What's n	ew	CAE Exporters.pdf
	MI:Viewe	er Help	Configuration Guide.pdf
	Reference	<u>e Documentation</u>	Excel Exporter Reference Guide.pdf
			Excel Importer Reference Guide.pdf
		GRANTA MI 11.0	FEA Exporter Author Guide.pdf
		GRANTA MI 11.0	Help and Documentation.pdf
		GRANTA MI 11.0	Installation Guide.pdf
		GRANTA MI 11.0	MI Viewer Home Page Author Guide.pdf
		GRANTA MI 11.0	Optimize Guide.pdf
		GRANTA MI 11.0	Record Version Control.pdf
		GRANTA MI 11.0	System Requirements.pdf
		GRANTA MI 11.0	Tabular Roll-up Module CLI.pdf
		GRANTA MI 11.0	Text Importer Reference Guide.pdf
		MI Explore 5.1 Co	onfiguration Guide.pdf
		FEA_Schema.zip	

2 Exporter files

An exporter for a given FEA package and model is comprised of two (or more) files.

- The **exporter configuration file** (.exp), a file that defines the structure of the initial XML, including the name of the target CAE analysis package, the material model type, the attribute data to be exported, and the transform files that can be used with it.
- One or more XSLT **transform files**, which convert the initial XML into the format required by the host application.

Note: XSLT can be used to run arbitrary malicious code, and exporters should only be installed from trusted sources.

Exporters are added to a GRANTA MI database by importing the exporter configuration (.exp) file and associated XSLT file(s) into the database for which it was written using the Schema tool in MI:Admin. For details, see Section 8.

The exporter source files for each database are located in subfolders named after the database key under the MI:Server *exporters* folder (C:\Program Files\Granta\GRANTA MI\Server\exporters), for example:



2.1 Developing your own exporters

When writing your own FEA exporters, you may want to store the files on disk while they are under development. You can structure the files as you wish, however, any files that you create should be kept completely separate from the Granta-supplied exporter files, to avoid being overwritten by future updates.

Placing custom exporter files in the specially-named __User_Exporters_ subfolder will ensure that your source files are preserved. The __User_Exporters_ folder is located within the exporters folder for the database, for example:

C:\Program Files\Granta\Granta MI\Server\exporters\MyTestDB_User_Exporters_

After they have been loaded into the database, any changes made to exporter configuration (*.exp*) files in the __User_Exporters__ folder will be picked up by MI:Server after clicking **Refetch exporters** in MI:Admin. Changes to the XSLT will be picked up when the files are called during an export.

2.2 Exporter configuration file parsing and checking

When a user initiates an export from GRANTA MI, MI:Server searches for files with the *.exp* file extension and then parses and checks each matching file it finds.

Note that only standard XML special characters are allowed in XML content. For example, use ° instead of °.

If an exporter fails either parsing or checking, an error message will be displayed to users with administrative privileges (see Section *11, Debugging* exporters). The problem must be remedied before MI:Server will display the exporter in the list of available exporters and thus allow users to use it.

Parsing and checking process

The *.exp* configuration file is parsed to check that it is valid XML and then the following checks are performed:

1. XSLT files specified in the Transform elements are checked to ensure that they exist and are valid XML.

Note that GRANTA MI only supports XSLT v1.0 at this time.

- 2. The database tables are checked to ensure that they exist.
- 3. Data such as attribute names and parameter names are checked to ensure that they are not blank.
- 4. Other inconsistencies are flagged.
- 5. If any of these checks fail, then the exporter will not be shown in the list of available exporters in MI:Admin.

More checks (such as whether the attributes required by the exporter can be accessed by the current user) are performed during the export process.

3 Exporter config options

A number of exporter properties are defined as attributes in the <ExporterConfig> element, the root element of the configuration file. For example:

```
<ExporterConfig xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
AbsoluteUnitsOptional="true" UseAbsoluteUnits="true"
Name="ABAQUS_6_Lin_TempDep_Iso" MaxRecords="50"
xsi:noNamespaceSchemaLocation="../ConfigSchema.xsd">
```

AbsoluteUnitsOptional

Set this to true if the FEA package can accept both absolute and relative units.

• If *true*, the user will be given the choice of whether to have absolute units in the exported data; for example:

Choose the unit system of the exported data	SI (Consistent)	¥	1
Use absolute temperature units			
			_

• If *false*, the value of UseAbsoluteUnits determines whether the export will contain absolute units.

UseAbsoluteUnits

Determines whether units such as temperature are output as absolute (i.e. K or R) or relative (C or F). If AbsoluteUnitsOptional is true, then the user can override this value.

Name

A name for this exporter (a string)

MaxRecords

Sets the maximum number of records that the FEA package can import. If this value is set to a number less than 1, then no limit is assumed.

This value is displayed on the Export page in MI:Viewer:

	✦Record List 🔸 🔶 📋		View	Tools	Units
	Export				
	You have 1 record(s) from table 'MaterialUniverse	' in your list.			
	Select an exporter from the list below.				
	The number on the right hand side of the table is the	maximum number of records that can be exported.		_	
	CATIA V5	<pre><exporterconfig <="" maxrecords="50</pre></th><th>" th=""><th>></th><th></th></exporterconfig></pre>	>		
	Sotropic	Exports isotropic data to the Granta-CATIA interchange format.		500	
	- Abaqus 6)	
	Linear, temperature-independent, isotropic, thermal, plastic	Exports temperature independent, isotropic data to the Abaqus format.		50	
	 ANSYS Workbench v12 onwards 				
	Linear, temperature-dependent, isotropic, thermal, plastic	Exports isotropic data to ANSYS Workbench format. Where temperature dependent da available it will be exported, if not room temperature data will be exported. Where stress strain curves are available they will be exported to the plastic hardening model.	tais s-	50	
1	Autodesk Inventor 2012				

4 Exporter details option

4.1 Package, Model, Description

The <Details> element in the exporter configuration file defines the target CAE analysis package and material model type. It can include the following elements.

<Package>

Defines the target CAE analysis package. For example:

<Package>Abaqus 6</Package>

<Package>CATIA V5</Package>

In MI:Materials Gateway exporters, <Package> is used to identify the Gateway Host; see the *MI:Materials Gateway Installation & Configuration Guide* for more information on exporter configuration for Gateway.

<Model>

Defines the material model type. For example:

<Model>Linear, temperature-dependent, isotropic, thermal, plastic</Model>

<Model>Isotropic</Model>

<Description>

Provides a description of the exporter. For example:

<Description>Exports isotropic data to the Granta-CATIA interchange format. </Description>

The Package, Model, and Description all appear on the Export page in MI:Viewer:

	🗲 Record List 🛧 🔸 💠 怕		View	Tools	Units
-	Export				
	You have 1 record(s) from table 'MaterialUniverse Select an exporter from the list below. The number on the right hand side of the table is the	' in your list. maximum number of records that can be exported.			
<package></package>	CATIA V5				
	◎ Isotropic ●	Exports isotropic data to the Granta-CATIA interchange format.	5	500	
	Abaqus 6	(December 1 and 1			
	Linear, temperature-independent, isotropic thermal, plastic	CAPPOID CONFIGURATION Percent Configuration	5	i0	
	 ANSYS Workbench v12 onwards 				
	 Linear, temperature-dependent, isotropic, thermal, plastic 	Exports isotropic data to ANSYS Workbench format. Where temperature dependent da available it will be exported, if not room temperature data will be exported. Where stress strain curves are available they will be exported to the plastic hardening model.	ita is 5 ss-	50	
	Autodesk Inventor 2012			-	

4.2 Applicability

The applications in which an exporter will be used can be specified as part of the exporter configuration. This allows applications to filter the exporters available to users, so that that only relevant exporters are available within each application.

The <Applicability> element in the <Details> section of the exporter configuration file defines the target application for the exporter. For example:

```
<Applicability>
<Tag>MIMaterialsGateway</Tag>
</Applicability>
```

The <Tag> value can be one of:

- MIMaterialsGateway available in MI:Materials Gateway applications only; not available in MI:Viewer
- MIViewer available in the MI:Viewer application; not available in MI:Materials Gateway.

MI:Explore can be configured to include only exporters with the MIViewer applicability tag, only exporters with the MIMaterialsGateway applicability tag, or all exporters; see the MI:Explore Configuration Guide for details.

4.3 Output file options

The optional <OutputFile> element can be included in the <Details> section to specify the default output FEA filename.

When an FEA export output file is saved in MI:Viewer, the default filename *feaExport.txt*. is suggested. Most browsers allow the user to choose a different filename.

By including the OutputFile element in the exporter configuration file, the exporter author can exert some control over the suggested filename.

The OutputFile element may include three elements, specifying the file extension and file name:

FileNameConvention

One of:

Value	Output filename is
Default	feaExport
RecordName	The record full name
TreeName	The record tree name, that is, the name displayed in the Contents tree in MI:Viewer. Note that when exporting more than one record, the TableName naming convention will be used instead of RecordName or TreeName.
TableName	The name of the database table where the record is located followed by the number of records exported in parentheses – i.e. (N)
Custom	A filename defined with the Custom element.

Extension

The file extension (without a period).

Custom

A user-defined filename. (Only applies when FileNameConvention is set to Custom). This may contain the following special replacement tokens:

- {recordname} The name of the first record in your record list
- {treename} The tree-name of the first record in your record list
- {tablename} The table name of the exported records
- {numberofrecords} The number of records to be exported

Example 1: RecordName

When exporting a single record named "Kevlar 149 aramid fiber", the sample code above would result in a default output filename of: *Kevlar 149 aramid fiber.out*. When exporting 10 records, the sample code above would result in a default output filename of: *MaterialUniverse (10 records).out*.

Example 2: Custom

```
<Details>
<OutputFile>
<FileNameConvention>Custom</FileNameConvention>
<Extension>dat</Extension>
<Custom>{recordname}-from-{tablename}</Custom>
</OutputFile>
</Details>
```

When exporting a single record named "Kevlar 149 aramid fiber", the sample code above would result in a default output filename of: *Kevlar 149 aramid fiber-from-MaterialUniverse.dat*.

When exporting more than one record (assuming that "Kevlar 149 aramid fiber" was the first record in the list) then the above sample will still generate the same filename (because {recordname} and {treename} always refer to the first record only).

4.4 Export file encoding

The optional <FileEncoding> element can be included in the <Details> element to specify the encoding for the output file.

By default, when a user chooses to save a file to disk, the file saved as UTF-8 encoded text. The file will not include the UFT-8 byte order mark (the characters \EF \BB \EF). Using this encoding scheme, the exported data will be able to be read by the majority of Windows applications.

However, in some circumstances it may be necessary to export the file in a different encoding. You can set the encoding of the output using the FileEncoding element. If omitted, the default UTF-8 encoding will be used.

To set the encoding you must specify the code page identity. Some common codepages are:

Code Page Identity	Name (and alternative names)
65001	UTF-8
1252	Windows-1252
65000	UTF-7
20127	US-ASCII, US, ASCII, cp367
1200	UTF-16LE, UTF-16, Unicode
1201	UTF-16BE, unicodeFFFE

Some encodings include a special series of characters at the start of the file or byte stream which denote the encoding and byte order of the following data. These characters are known as the byte order mark (or BOM for short).

Most common windows text editors (which can handle many different file encodings) will recognize the byte order mark and will not try to display it. Sometimes, if the byte order mark is not present, certain applications may not open the file correctly because it assumes the incorrect encoding for the file. (For example, Notepad may mistakenly assume that a file is US-ASCII encoded when in fact it is UTF-8 encoded and some non-ASCII characters may be displayed incorrectly).

However, some applications may not know to ignore these characters and treat them as data (you may see spaces or strange symbols at the start of the file).

You can specify whether or not the byte order mark is included in the file by setting the IncludeBOM attribute in the FileEncoding element.

Both the CodePage and IncludeBOM attributes are required: even if the code page has no byte order mark, the IncludeBOM attribute is still required.

```
<FileEncoding CodePage="1252" IncludeBOM="false" />
```

4.5 Other options

Other output options are available, for example, allowing the output folder to be specified; see the ConfigSchema.xsd file for more information.

5 Database export options

5.1 Database Key and GUID

The <Database> element specifies the database from which data is to be exported. The database GUID and key must be specified. For example:

```
<Database Guid="ea6b62b6-4e8a-4f78-bc4b-cf51ec2351ff" dbKey="Metals_Test">
```

5.2 Unit systems

The <UnitSystems> element under <Database> specifies the unit systems available for exports from this database.

A GRANTA MI database has a number of **unit systems** defined as standard. These include common unit systems such as *Metric*, *US Customary*, and *UK Imperial*, as well as unit systems that are typically used for exporting material models, such as *SI (Consistent)*, *CGS (Consistent)*, *IPS (Consistent)*, and *FPS (Consistent)*.

To export unit system information, include a UnitSystem element in the configuration file, for example:

```
<UnitSystems>

<UnitSystem Name="SI (Consistent)" />

<UnitSystem Name="CGS (Consistent)" />

<UnitSystem Name="FPS (Consistent)" />

<UnitSystem Name="IPS (Consistent)" />

<UnitSystem Name="mmNs (Consistent)" />

</UnitSystems>
```

5.3 Tables

The <Tables> element under <Database> defines the tables from which data will be exported.

Although the schema allows more than one table to be defined, it is recommended that you only have one <Table> element per exporter file.

```
<Tables>
<Tables>
<Table Name="MaterialUniverse" Guid="0000DD92-0011-4FFF-8FFF-0000FFFF0000">
...
</Table>
<Table>
<Table>
```

6 Table export options

The <Table> element identifies the table from which data will be exported. The table name or GUID must be specified, for example:

```
<Table Guid="E59175DD-8E12-4425-9540-2EC81454C423">
<Table Name="MaterialUniverse">
```

6.1 Exporting all attribute data from a table

Include an AttributesIncluded attribute on the Table element to export the data from all the attributes in the specified table. For example:

```
<Table Name="xxx" AttributesIncluded="AllIncludingMeta | AllExcludingMeta "/>
```

Options are:

- AllIncludingMeta export the data from all attributes and metadata in the table.
- AllExcludingMeta export the data from attributes only; do not export metadata attributes.

6.2 Exporting table version control information – <VersionControl>

To export record version control information for a table, include a VersionControl element in the <Table> element:

```
<VersionControl Include="true | false" />
```

For example:

```
<Tables>
    <Tables>
    <Table Name="PolymerUniverse">
        <VersionControl Include="true" />
        <DataQuality Include="true" />
```

Note that the attribute data version number is always output to the initial XML and does not rely on this setting.

6.3 Exporting table data quality information - <DataQuality>

To export quality ratings for the data in a table, include the following element in the <Table> element:

```
<DataQuality Include="true" />
```

For example:

```
<Tables>
<Table Name="PolymerUniverse">
<DataQuality Include="true" />
```

If <DataQuality> is omitted, then no data quality information will be included in the initial XML.

6.4 Exporting access control information - <AccessControl>

For databases with a permission-based access control schema configured, access control information is exported automatically. Note that records and data are only exported if the user has permissions to see that data.

To suppress access control information for a table, include <AccessControl Include="false" /> in the <Table> element. For example:

```
<Tables>

<Table Name="PolymerUniverse">

<AccessControl Include="false" />
```

Regardless of this, the Access Control settings for the user always determine which records and attributes are included; this setting does not allow a user to subvert Access Control.

6.5 Exporting parameter information – <FullParameterDetails>

To export detailed information about all parameters available to functional attributes in the table, include the following element in the <Table> element:

```
<FullParameterDetails Include="true" />
```

```
See also: Section 7.4, Exporting functional data.
```

6.6 Exporting attribute data - <DBAttributes>

The <DBAttributes> element specifies a list of the attributes to be exported in the initial XML. Within this element, you include one or more <DBAttribute> elements.

See Section 7, Attribute export options.

6.7 Exporting user input – <UserData>

Users can be prompted to enter additional data which is exported along with the record data, for example to specify

- A thermal expansion data reference temperature
- The failure criterion for brittle materials.

6.7.1 User input scope

User data may relate to the whole export operation, and/or to each record being exported.

Element	Description
General	Defines user input that relates to all of the data being exported. For example, a reference temperature when exporting thermal expansion data.
Record	Defines user input that relates to individual records being exported.

6.7.2 Input

The <Input> element describes the data that can be entered.

<Input xsi:type="inputtype" Value="value" Name="name" Optional="true|false">

Attribute	Description	
xsi:type	 The input data type. One of UserInput – the user can enter any string value. ChoiceInput – the user must select one from a list of options. See example below. IntInput – the user must enter an integer value. FloatInput – the user must enter a valid floating point number. See example below. 	
Value	The default value of the input. The user may overwrite this.	
Name	The input name	
Optional Specifies whether the input is optional (true) or mandatory (false). If Optional="false", the exporter software will ensure that the user enters a variable.		

ChoiceInput example

Available choices are specified using <Choice> elements as shown in this example, where each <Choice> defines the text that will appear on the options in the list box. For example:

```
<UserData>

<Record>

<Input xsi:type="ChoiceInput" Value="MODIFIED MOHR" Name="FAILURE_CRITERIA"

Optional="false">

<Prompt>Failure Criterion</Prompt>

<Choices>

<Choice Text="NONE" />

<Choice Text="MODIFIED MOHR" />

<Choice Text="MODIFIED MOHR" />

<Choice Text="MAXIMUM SHEAR STRESS (TRESCA)" />

<Choice Text="DISTORTION ENERGY (VON MISES)" />

</Choices>

</Input>

</Record>

</UserData>
```

Additional Data

Please enter the following data. * represents a required field. The type of the data is shown to the right of the question.

Please enter data for: Titanium, alpha-beta alloy, Ti-6AI-4V, solution treated & aged			
Failure Criterion	*	MODIFIED MOHR	choice
		NONE	<u>7</u>
		MODIFIED MOHR	
		MAXIMUM SHEAR STRESS (TRESCA)	
		DISTORTION ENERGY (VON MISES)	
and a second			

Note that ChoiceInput can also be used to provide Boolean (e.g. yes/no) user input options.

float

FloatInput example

6.7.3 Prompt

Reference temperature for thermal expansion data.

A <Prompt> element within an Input element specifies the text label on the input field on the Export page in MI:Viewer. You can see this in the two examples above.

295.78

7 Attribute export options

The <DBAttributes> element includes one or more <DBAttribute> elements which specify each attribute to be exported.

7.1 Identifying the attribute - <DBAttribute>

GRANTA MI attributes can be identified by their standard name, name, or GUID. The **standard name** should always be used, if defined. Attributes and meta-attributes that cannot be identified using a standard name may be identified using their name or GUID.

XML Attribute	Description
StandardName	The attribute's standard name, where one is defined. For example:
	<pre><dbattribute standardname="Tensile strength, ultimate"></dbattribute> <dbattribute standardname="Compressive strength, ultimate"></dbattribute> <dbattribute standardname="Tensile strength, yield"></dbattribute> <dbattribute standardname="Thermal expansion coefficient"></dbattribute></pre>
Name	The attribute name. For example: <dbattribute name="Designation"></dbattribute>
	However, note that the Name may change, and is not necessarily unique, and so using the standard name is recommended.
Guid	All database objects have a GUID (globally unique identifier) which can be used to identify them if no standard name is defined. For example:
	<dbattribute guid="0000003F-0001-4FFF-8FFF-
DD92FFFF0000" name="Density"></dbattribute>

7.2 Using aliases

Aliases can be used to specify attribute and parameter names in the initial XML.

7.2.1 Attribute aliases

By default (when no alias is defined), the name of an exported attribute in the output file will be the same as its name in the source database. For example:

Attribute name	Tens. Yield Stress (L-Dir) with Temp.
Output name	Tens. Yield Stress (L-Dir) with Temp.

However, it is possible to specify an alternative name using an alias; this can be the Standard Name, or any other name. In this case, the specified alias name will be output, for example:

Attribute name	Tens. Yield Stress (L-Dir) with Temp.
Standard name	Tensile Strength, yield with temperature
Output (StandardNameAsAlias="true")	Tensile Strength, yield with temperature
Output (Alias="Yield strength w Temp")	Yield strength w Temp

Aliases are specified for attributes using the optional Alias or StandardNameAsAlias attributes on the DBAttribute element.

Attribute	Description
Alias="newname"	Use the specified string <i>newname</i> as the attribute name in the output file. This can be any string.
StandardNameAsAlias="true"	Use the attribute's standard name in the output file.

7.2.2 Parameter aliases

Aliases can be used for parameter names as well as attribute names, by including Alias or StandardNameAsAlias attributes in the ParameterAlias element.

For example, to output the parameter name "Stress Ratio" as "R ratio":

<ParameterAlias StandardName="R-Ratio" Alias="R Ratio" />

For example, to output the parameter name "Stress Ratio" with its standard name:

<ParameterAlias StandardName="R-Ratio" StandardNameAsAlias="true" />

Without an alias, the parameter name will be output as "Stress Ratio".

7.2.3 Summary of alias options

The effect of using different options to name output objects is summarized in the following table, with examples using the ParameterAlias element.

	Name of parameter	Standard name of parameter	Name output in the export
<parameteralias StandardName="StandardTemp" StandardNameAsAlias="true" /></parameteralias 	Temperature	StandardTemp	StandardTemp
<parameteralias StandardName="StandardTemp" /></parameteralias 	Temperature	StandardTemp	Temperature
<parameteralias StandardName="StandardTemp" StandardNameAsAlias="true" Alias="ExplicitTemp" /></parameteralias 	Temperature	StandardTemp	Error - cannot specify both
<parameteralias Alias="ExplicitTemp" /></parameteralias 	Temperature	StandardTemp	ExplicitTemp
<parameteralias StandardName="StandardTemp" Alias="Temp" /></parameteralias 	Temperature	StandardTemp	Temp

7.3 Exporting meta-attributes

You can export all meta-attributes for a particular attribute by adding the XML property IncludeAllMeta="true" to the DBAttribute element. For example:

<DBAttribute StandardName="Tensile strength, ultimate" IncludeAllMeta="true" />

This will export the attribute with the standard name "Tensile strength, ultimate" and all of its meta-attributes. Since meta-attribute names are not unique within the table (for example, two attributes can both have meta-attributes called "Notes") the meta-attributes are given the alias (see Section 7.2) of "ParentAttributeName_MetaAttributeName".

If you want to export specific meta-attributes, you can do so by including a ParentAttribute element within the DBAttribute element. For example:

```
<DBAttribute Name="Notes">
<ParentAttribute StandardName="Tensile strength, ultimate" />
</DBAttribute>
```

7.4 Exporting functional data

Functional data is data is stored as a collection of points and parameter values in the database. The format of the data that is output in the initial XML depends upon what is specified in the DBAttribute configuration element in the exporter configuration file.

The values of the parameters used for the interpolation are taken from the user's current preferences, set on the **Export Options** page in MI:Viewer (see below).

For example, assume that the attribute Stress is a function of parameters Strain and Temperature. The exporter configuration file contains the line:

```
<DBAttribute Name="Stress" />
```

In MI:Viewer on the **Export Options** page, if the user clicks **Set Parameter Values**, they will be given the opportunity to set the values of Temperature and Strain. The value output in the initial XML will be a single simpleValue element that will contain a range value – calculated by interpolation.

Information about all the parameters available to functional attributes in the table can be exported with the data; see Section *6.5, Exporting parameter information*.

7.4.1 Grid data

Functional data can be stored in two separate formats: grid and series.

Grid functional data can be thought of as a multi-dimensional table of data. It is permissible to interpolate grid data to obtain a value for the attribute provided that we specify parameter values that are within the range covered by the grid.

When interpolating, if there are missing points around the point of interest, the next nearest points are used in the interpolation and the missing points are ignored. Note that we never extrapolate values.

For example, imagine that we have grid data for Stress vs Strain and Temperature. Our data is populated for values of Strain between 100 and 500 MPa (steps of 50 MPa) and Temperatures

between 0 and 500 degrees Celsius (steps of 10 degrees Celsius). We are able to interpolate a value of Stress for any combination of the parameters Strain and Temperature provided that we choose a value of Strain between 100 and 500 MPa and a value of temperature between 0 and 500 degrees Celsius.

We may also restrict the value of just one parameter to obtain a new (interpolated) grid. For example, we could choose a fixed value of temperature and get a one-dimensional grid (Stress vs Strain) for that temperature.

Instead of specifying which parameters are fixed, the exporter configuration specifies which parameters are "free parameters" (i.e. not fixed).

7.4.2 Series data

Series data (also called a series graph) can be thought of as a collection of line graphs. For example, we could have series data for Stress vs Strain. This would consist of Stress values for a range of Strain values, which one could plot on a simple line graph. Now the attribute being measured may well depend upon other parameters – in our example, Stress also depends upon Temperature. The series graph for Stress would have several lines – each line is the Stress vs Strain curve for a given temperature.

In this case the parameter Temperature is referred to as a *constraint*. The only free parameter is Strain (the free parameter is often referred to as the X parameter since it is most often plotted on the X axis of charts).

Each line in the series graph may cover a different range of X and the points in different series need not be at the same values of X. Each line may also have a different number of points. Each line is essentially independent of the other lines in the graph; they just quantify the same attribute value.

The most important difference between series and grid data is that we cannot interpolate between constraint values in series data.

For example, let us consider a series graph of Stress vs Strain (as the free parameter) and Temperature (as a constraint). The graph may consist of three curves:

- A curve measured at 50 degrees Celsius for values of Strain between 100 and 400 MPa.
- A curve measured at 150 degrees Celsius for values of Strain between 200 and 300 MPa.
- A curve measured at 200 degrees Celsius for values of Strain between 100 and 500 MPa.

We may **not** infer a value of Stress for Temperature = 75 degrees Celsius, Strain = 250 MPa even though it may appear that we are able to. We may only interpolate a value of Stress at 250 MPa Strain for temperatures of exactly 50, 150 and 200 degrees Celsius. In other words we may only interpolate along a single line – not between lines.

This of course has important ramifications for exporting functional data stored as a series. If we are not exporting the entire graph, data will only be output if the parameter values are set to exactly match the constraint values of a line within the graph.

7.4.3 Free parameters

You can leave a parameter value unset by including the parameter in the FreeParameters element. For example:

```
<DBAttribute StandardName="Specific heat capacity with temperature">
    <FreeParameters>
        <Parameter StandardName="Temperature" />
        </FreeParameters>
```

Now, if the user clicks **Set Parameter Values** in MI:Viewer, they will not see **Temperature** in the list of parameters (since we are not restricting Temperature to a single value). The output will now contain a complexValue element, which contains either a series or grid element (depending on whether the original data was series or grid – see below).

7.4.4 Exporting functional data without interpolation

If you do not want any interpolation to occur, include EntireGraph="true" in the DBAttribute element, for example:

<DBAttribute Name="Stress" EntireGraph="true" />

This instructs the exporter to output the functional data in its entirety, that is, the whole of the graph will be output in the initial XML, no interpolation will be done and the data will output exactly as it is stored in the database (unit conversion not withstanding). The user does not get to set any parameter values.

Note that you cannot specify EntireGraph="true" and any free parameters, since EntireGraph="true" implies that all parameters are free.

7.4.5 Grid and Series data format in the initial XML

When EntireGraph="true"

If we specify EntireGraph="true", then series graphs will be output in a series element and grid data will be output in a grid element. Note that both series and grid elements share the same inner format. The difference between the inner XML is that series graphs constraint parameters are marked with isconstraint="true" in the parameter elements.

With no free parameters

If we do not specify EntireGraph="true" or any free parameters for functional data, then the software attempts to interpolate a (single) attribute value (based upon the user's chosen parameter values). This value will be output as a range in the initial XML (since each 'point' in both grid and series data is in fact a range value).

This value can only be calculated for grid data if:

• The parameter values lie within range for grid data.

This value can only be calculated for series data if:

- The parameter values exactly match the constraints values for a line within the series graph.
- The parameter value of the graph's X parameter lies within the range of the line.

With free parameters

Grid data: If we specify one or more free parameters for a functional attribute then data can only interpolate if the parameter values of the fixed parameters lie within the range of the grid data. If this is the case, then a grid element is written to the initial XML. There will only be one parameter value for each of the fixed parameters and the parameter element will have isconstraint="true".

Series data: Data will only be output if:

- There is only one free parameter specified.
- The free parameter is the X parameter for the series graph.
- The values of the parameters exactly match the constraint values for a series in the graph.

If these conditions are met, then a series element is written to the initial XML. This will contain only one line from the series graph.

7.4.6 Warnings generated during export

During the export, it may not be possible to output any data for some functional attributes. Most of these errors are caused by the failure to interpolate functional data.

7.5 Exporting Embedded Equations and Logic (EEL) data

Embedded Equations and Logic (EEL) attribute values can be exported. The 'value' exported for EEL data in the initial XML consists of two components:

- An mfdMetaData element that contains:
 - Information about the expression
 - o Information about the curves defined in the database
 - \circ The parameter extents (i.e. the domain over which the expression is valid)
- The numeric value of the EEL data. Like float functional data, the numeric value of EEL data can be exported as a single numeric value or as series data.

Note that user-defined curves (which are stored in the user's session) are not exported. Only the EEL data that is stored in the database will be exported.

7.5.1 Exporting EEL data as a single numeric value

If both of the following are true, then the EEL's expression will be evaluated at the parameter values specified in the "default curve":

- EntireGraph is not set to true in DBAttribute and
- DBAttribute does not include any FreeParameters

If an EEL's data uses an expression that is not a function of any parameter (for example, it is a function of the record's attributes only), then it will always be exported as a single point irrespective of the settings in the exporter configuration file.

Parameter values

Note that the parameter values used to evaluate EEL data are likely to be different from those used to interpolate float functional data: all float functional data use the same set of parameter values for interpolation (these can be set by the user running the export). The parameter values used to evaluate EEL data are those stored in the database (in the default curve's parameter values).

If an MI:Viewer user has overridden these parameter values, it will not affect the exported output. Similarly, if the MI:Viewer user has created user-defined curves (which are stored in the user's session), these will not be exported. Only the data that is stored in the database will be exported.

7.5.2 Exporting EEL data as series data

If the DBAttribute element in the exporter configuration

- has EntireGraph="true" or
- specifies a FreeParameters element (with exactly one Parameter)

then the EEL data will be exported as a set of curves. The data's default curve along with all other curves are exported.

Parameter values

Note that the parameter values used to evaluate EEL data are likely to be different from those used to interpolate float functional data. The parameter values for each exported curve are stored along with the curve in the database. User-defined curves (which are stored in the MI:Viewer user's session) will not be exported.

The free parameter

If no FreeParameters section is included, then the free parameter is taken to be the EEL data's default X axis parameter.

You can specify the free parameter in the FreeParameters element (see Section 7.4.3). The EEL data must be a function of the specified free parameter, otherwise no data will be exported.

The free parameter extent

The minimum and maximum values of the free parameter are determined by the parameter's extent, which is stored in the EEL data.

By default, the expression is evaluated at 100 points between the minimum and maximum values unless:

- The parameter extent's minimum is exactly equal to the maximum in which case the expression will be evaluated at this 1 point only
- The free parameter is a discrete parameter in which case the expression will be evaluated at all the discrete values that are defined within the EEL data's free parameter's extent.

Controlling the number of points exported per curve

If the free parameter is not discrete, the number of points exported for each curve in the EEL data can be controlled by added the attribute NumberOfPointsPerCurve to the DBAttribute element, for example:

<DBAttribute Name="Stress" NumberOfPointsPerCurve="10" EntireGraph="true"/>

If you enter a number that is less than 1, it is ignored and the default value of 100 is assumed. If you enter 1 point, then the EEL data will be evaluated at the parameter extent's low value only.

Free parameter value spacing

If the free parameter's "scale type" (as defined in the parameter's definition in MI:Admin) is *Linear*, then the software chooses evenly-spaced values between the minimum and maximum value.

If the parameter's scale type is *Logarithmic*, then logarithmically-spaced values between the minimum and maximum value are used. Note there is no facility to explicitly choose the free parameter values in the exporter definition.

8 Specifying the transforms

The <Transforms> element specifies the names and locations of the transform XSLT files that will perform the conversion from initial XML to a format suitable for the target FEA package.

For example:

```
<Transforms>

<Transform xsi:type="XSLTransform" Filename="..\TESTtransform1.xslt"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" />

<Transform xsi:type="XSLTransform" Filename=".\XSLT\TESTtransform2.xsl"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>

</Transforms>
```

The specified Filename is relative to the location of the exporter configuration file on disk; see Section *2, Exporter files*.

9 Adding (importing) an exporter to a database

An exporter is added to a database by importing its configuration file and associated XSLT file(s) into the database for which it was written. This is done using MI:Admin; you will need administrative privileges to the database.

- If database security groups have been set for the database, you must be a member of the Admin database security group <u>and</u> a member of one of the system security groups.
- If database security groups have not been set, you must be a member of the Admin system security group.

9.1 To add a new exporter

To import a new FEA exporter to a database, use the Schema tool in MI:Admin:

- 1. On the Edit Files page, click on the Exporters tab and then click Import Files.
- 2. Once the files have been added to the database, click **Re-fetch Exporters** to save the files to disk in a directory on the GRANTA MI server.



Figure 1. Files page, Exporters tab in the MI:Admin Schema tool

10 FEA Export features in MI applications

10.1 FEA exporters in MI:Viewer

Where exporters are available for the data in the Record List, they can be accessed by clicking **Export** on the Reports page in MI:Viewer. If there are no exporters for the data in the selected records, the **Export** button will be disabled.

Reports			
Add records by clicking on the $$ 'Add to list' tool on the \Leftrightarrow menus thr MI:Viewer.	oughout GRANTA		Perform an operation on the records in your list.
You have 8 record(s) in your list.	Sort		
O MI: Training	^		Comparison Table
= MaterialUniverse (8 records	s shown)	\rightarrow	T: Comparison Chart
Titanium, alpha-beta alloy, Ti-6Al-4V, aged	×	export	exporters
Titanium, alpha-beta alloy, Ti-6Al-4V, annealed, generic	×		
Titanium, alpha-beta alloy, Ti-6Al-4V, solution treated & aged	×		• X-Y Chart
🖬 Aluminum, 7075, wrought, T6	×	1	
Aluminum, 7075, wrought, T73	×		Export .
Low alloy steel, AISI 4130, air melted, normalized	×		🗠 i 🖑
Low alloy steel, AISI 4130, air melted, quenched & tempered	×		
250 maraging steel, maraged at 900F	×		
			Run Reports
			There are no reports available for the records in your list.

The available exporters are listed on the Export page. The package and model name, description, and maximum number of records that can be exported, are all specified in the exporter configuration file.

←Record List ↑ ↓	V	ïew Tools	Units
Export			
You have 8 record(s) from table 'MaterialUniverse	' in your list.		
Select an exporter from the list below.			
The number on the C < Package> he table is <	Description> ecords that can be exported.		
○ Isotropic ← < <u>Model></u>	Exports isotropic data to the Granta-CATIA interchange format.		500
- Abaqus 6			
 Linear, temperature-dependent, isotropic, thermal, plastic 	Exports temperature dependent, isotropic data to the Abaqus format. If temperature dependent data is not available then room temperature data will be exported.		50
Linear, temperature-independent, isotropic, thermal, plastic	Exports temperature independent, isotropic data to the Abaqus format.		50
- ANSYS Workbench v12 onwards			
Clinear, temperature-dependent, isotropic, thermal, plastic	Exports isotropic data to ANSYS Workbench format. Where temperature dependent available it will be exported, if not room temperature data will be exported. Where st curves are available they will be exported to the plastic hardening model.	data is ress-strain	50
Autodesk Investor 012	And a set of		

Only Exporters with no <u>Applicability</u> tag specified, or with an <u>MIViewer</u> <u>Applicability</u> tag will be listed on this page.

10.2 FEA exporters in MI:Explore

MI:Explore users can access the available FEA exporters by clicking **Export** on the toolbar.

The package, model name, and description are all specified in the exporter configuration file.

Export data	×
All items (29)	
Available package CATIA V5	<pre><description></description></pre>
Available exporters:	Exports isotropic data to the Granta-CATIA interchange format.
	Export Close

Only Exporters with no <u>Applicability</u> tag specified, or with an <u>MIViewer</u> <u>Applicability</u> tag will be listed:

Export data				×
All items (29)				
CATIA V5				•
Abaqus 6 ANSYS Workbench v12 onwards	×			?
Autodesk Inventor 2012 CATIA V5 Pro/ENGINEER Wildfire 4.0	-1	E	xport	Close
Pro/ENGINEER Wildfire 5.0 SolidWorks 2011				
	- 18			
Deriver ellipste	×			

11 Debugging exporters

Users with administrative privileges for the database will see a slightly different user interface to non-administrative users, allowing them to troubleshoot exporter problems.

11.1 Parsing errors

If parsing or checking errors are detected in any exporters, a message and link are displayed below the exporter list; clicking on the message should provide some information about the error.

|--|

See also Section 2.2, Exporter configuration file parsing and checking.

11.2 XSLT transforms

Admin users will see a list of available Transforms list on the **Export Options** page. Users who do not have Administrator privileges will not see this list.



The Transforms list allows the Admin user to choose which (if any) XSLT transforms are running during the export, and can aid debugging of XSLT transforms if there is more than one XSLT specified in the exporter. You may also specify that no XSLT transforms are run (and thereby get the export's initial XML).

11.3 Missing attributes

If one or more attributes specified in the exporter could not be found in the database, they are listed on the **Export Options** page, for example:.

The following attributes could not be found:
 StandardName [Statistical basis] Name [] Identity [] Guid [] StandardName [Electrical conductivity] Name [] Identity [] Guid [] StandardName [Tensile stress with strain] Name [] Identity [] Guid []
You may still continue with the export.

Users who do not have Administrator privileges are not notified of missing attributes. These details are hidden from the non-admin user because the attribute may not be accessible to the user and we do not want to show a non-privileged user the names of restricted attributes.