Specification

OFML Article Mappings* (OFML Part VI)

Version 1.0

Status: Release

Thomas Gerth, EasternGraphics GmbH (author and editor)

2004-09-30

*Copyright © 2003–2008 Verband der Büro-, Sitz- und Objektmöbelhersteller e.V.

Contents

1 Introduction and outline

The tables described and defined in this OFML part serve to specify the interrelationships between data, which have been created according to the specifications of various other OFML parts describing different aspects of articles. Fig. 1 illustrates these interrelationships.

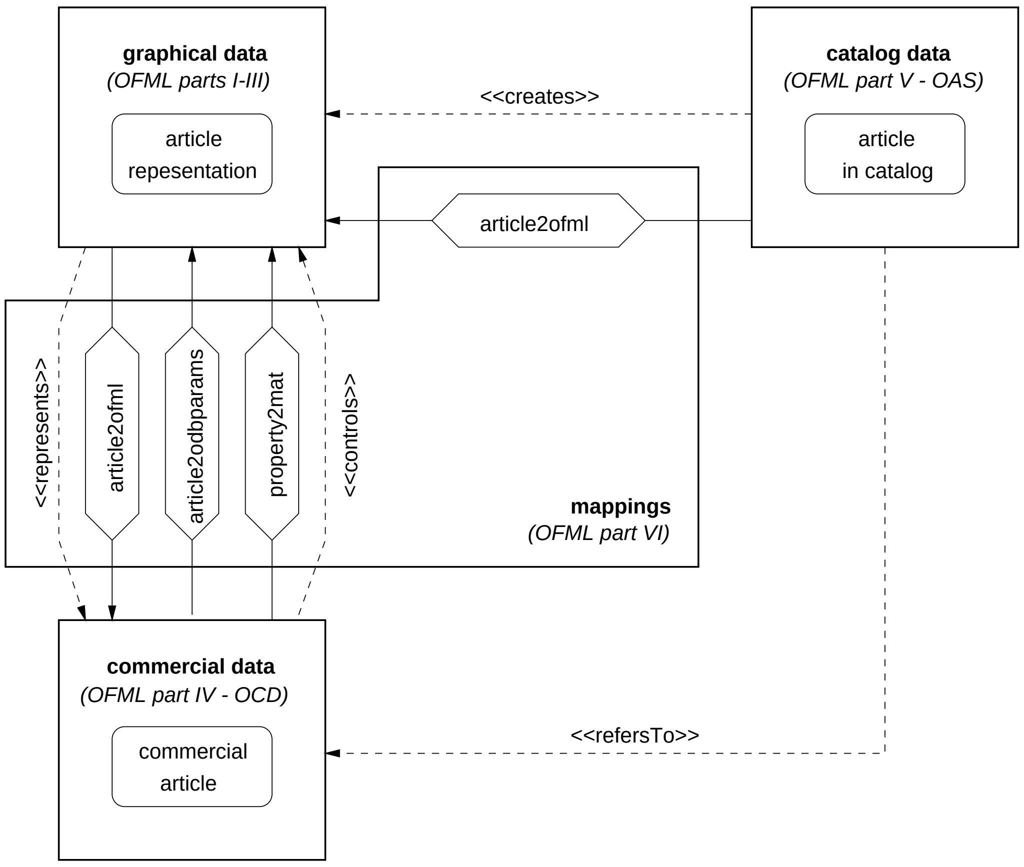

Figure 1: Position of mapping tables among OFML parts

The following mappings are used:

- *Article–OFML Mapping* (`article2ofml`)

  Defines, by which OFML-type with which parameters an article should be represented in an OFML runtime environment.

- *Article–ODB-Parameter Mapping* (`article2odbparams`)

  Defines the values of help variables for a given article configuration, which are used in the ODB–data for the graphical representation of the article (optional).

- *Property–Material Mapping* (`property2mat`)

  Defines, which property values correspond with which material layer assignments (optional).

2 General regulations

2.1 Lexical and syntactical regulations

As the physical interchange format between OFML compliant applications, CSV–tables (comma separated values) will be used. Referring to this, the following regulations are effective:

- Each of the tables described below is enclosed in exact one file. The file name is formed by the prefix "`oam_`", the specified table name and the suffix "`.csv`".
- Each line of the file (terminated by a new line character "`\n`") represents a data record. Blank lines (consisting of zero or more space characters or tabulators) will be ignored.
- The fields of a data record are separated by a semicolon.
- Lines which start with a double cross ("`#`") will be interpreted as a comment and are excluded from further processing.

During the following table descriptions, a field of a data record is specified by the following attributes:

- number
- name
- tag, whether the field belongs to the primary index of the table
- data type (see below)
- maximum length of the field (number of characters)[1]
- tag, whether the field needs to be filled necessarily (obligatory field)

The following *data types* are defined:

Char character string

The following lexical and syntactical regulations are effective:

1. All printable characters, except the semicolon, are permitted.
2. If a semicolon is to be contained in the character string, the whole field must be included in quotation marks ("`"`"). The opening and closing quotation marks will not be transferred when reading the field.
3. If the field is included in quotation marks, two consecutive quotation marks will be substituted by just one quotation mark when reading. A single quotation mark in a field included in quotation marks is not permitted.
4. If the field is included in quotation marks, space characters between the closing quotation mark and the next field separator respectively the line end will be ignored.

Params list of parameters

The following lexical and syntactical regulations are effective:

1. A parameter is specified as follows:
   <parameter-identifier>=<parameter-value>
   The individual parameters in the list are to be separated by comma.

[1]As there are no formal restrictions regarding the field lengths in CSV–tables, here only reasonable practical lengths will be stated for fields of data type `Char`, where possible.

2. For the parameter identifier all alphanumeric characters including the underline "_" are permitted. However, the identifier has to start with a letter or an underline.
3. As parameter values, integral and floating-point constants as well as literal symbols and character string constants can be stated. The syntax of integral and floating point constants corresponds to the general OFML syntax description in OFML part III. For literal symbols, the same regulations are effective as stated above for the identifier, but they always start with the character "@". Character string constants, in contrast to the general OFML syntax description, have to be included in inverted commas "'" (not in quotation marks """). An inverted comma within the character string has to be preceded with a backslash "\". A quotation mark within the character string has not to be preceded with a backslash, but the same regulations are effective as stated above for field type *Char* for the case, the character string contains a semicolon.

2.2 Directory structure

The concrete directory structure for the filing of graphical data, catalog data and commercial data is arranged by the respective OFML application. But usually, the (common) graphical data is located in one place (directory), whereas catalog and commercial data are located in separate directories corresponding to the respective sales regions, see Fig. 2.

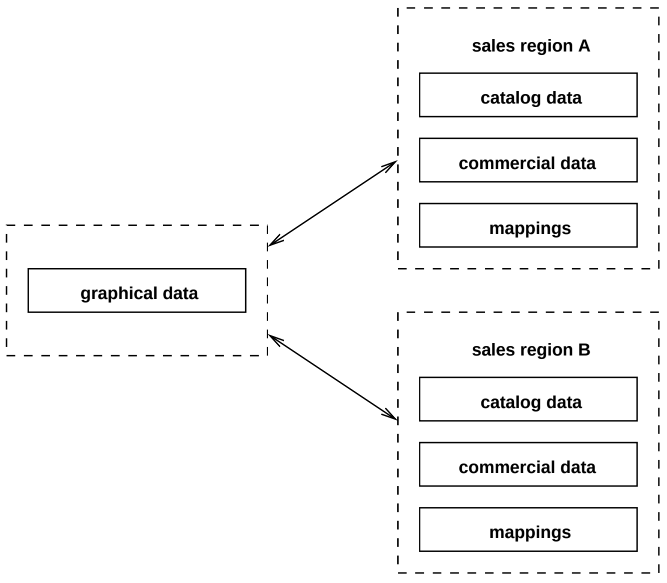

Figure 2: areas of directory structure

Regarding this general directory structure it is constituted, that the mappings have to be located in parallel to the commercial data for the respective sales region. Though there is a dependency of the mappings on the graphical data, too (which would motivate to file the mappings parallel to the graphical data), the dependency of the mappings on the commercial data is stronger, because there may be different sets of articles relevant to different sales regions respectively there can be even differences in article numbers of base articles. Furthermore, the probability and frequency of changes in commercial data is higher than of changes in graphical data.

3 The tables

3.1 Conceptual data model

Fig. 3 illustrates the conceptual data model of the OFML parts connected by the mappings and clarifies the job of the individual mappings[2]. The model puts the overview in Fig. 1, section 1, in more concrete terms.

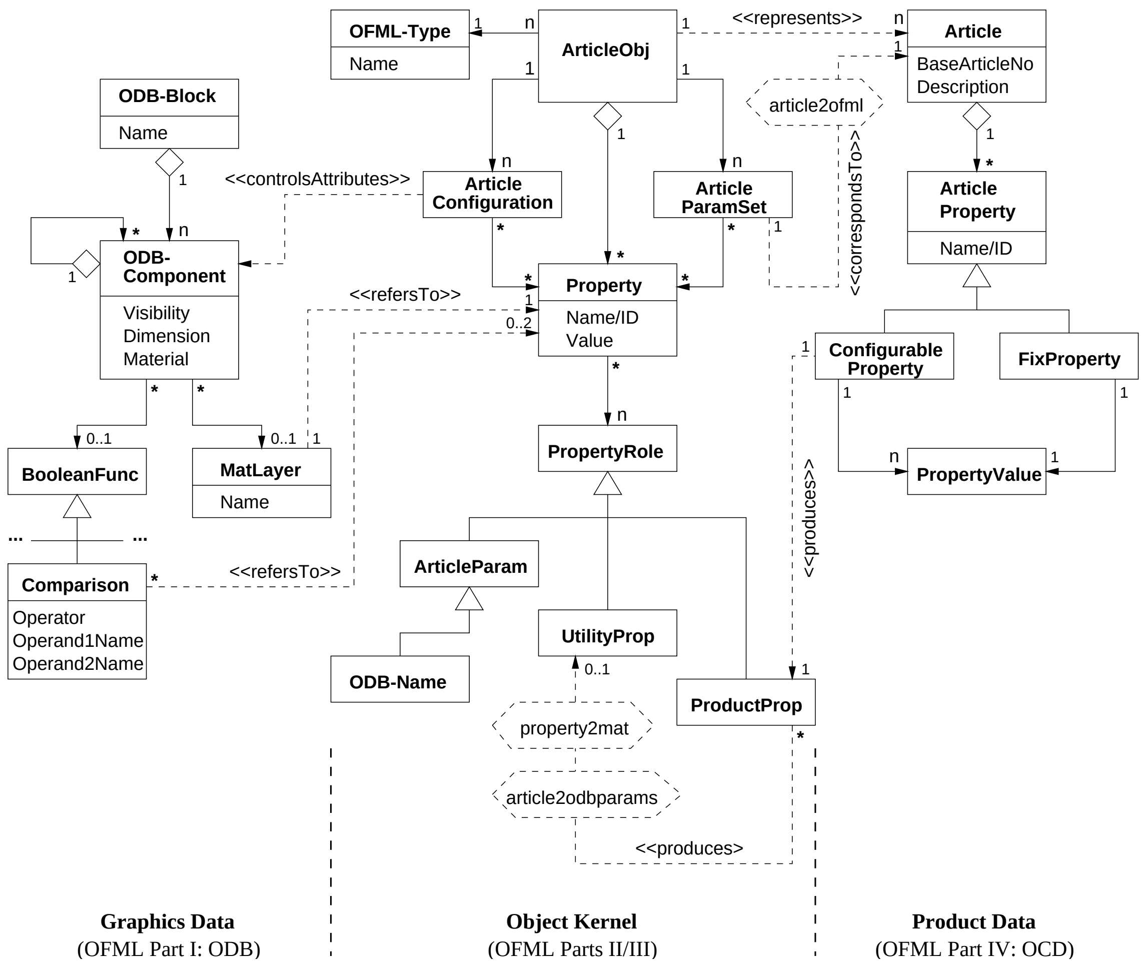

Abb. 3

[2]The notation used in this diagram is consistent with the *Unifed Modeling Language (UML)* specification (www.omg.org/uml), except from the description of the mappings.

3.2 Overview

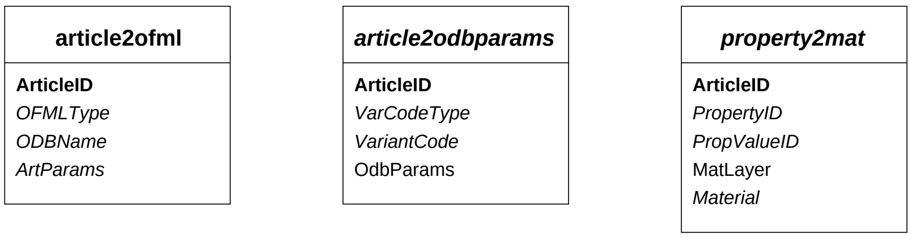

Figure 4: table overview

Key fields are highlighted by bold characters and fields, which are not mandatory, are highlighted by italic characters.

3.3 The Article–OFML Mapping

Table name: `article2ofml`
Mandatory table: yes

This mapping specifies, by which OFML type with which parameters an article should be represented within an OFML runtime environment.

Nr.	Name	Key	Type	Length	Obligatory	Explanation
1.	ArticleID	X	Char		X	base article number
2.	OFMLType		Char			fully qualified OFML type
3.	ODBName		Char			fully qualified ODB name
4.	ArtParams		Params (Char)			article parameters

Notes:

- The table may contain an entry with the character '∗' instead of a real article number in field 1. The values in the other fields of the entry then will be used by the OFML runtime environment when creating articles, for which there is no explicit entry in the mapping table.
- OFML types and ODB names have to be stated fully qualified, i.e. including the identifier of the OFML package, which contains the implementation of the type resp. the ODB data.
- As OFML types there may be stated the OFML base types *OiPlElement* and *OiOdbPlElement* (see OFML part III) or types from an OFML extension package, which are directly or indirectly derived from these base types. The initialization function of the type may possess only the two standard parameters for the father object and for the object identifier.
  OFML base types will be provided by the OFML runtime environment. OFML extension packages may be provided by various software companies or the data creators themselves and will be registered by the OFML runtime environment.
- The field for the OFML type may be empty, if there is no explicit graphical representation intended by the data creator. Then, the OFML runtime environment will use a standard representation for this article. (The subsequent fields will not be evaluated in this case.)
- If ODB (OFML part I) is used for the graphical representation of the article, in field 3 the fully qualified name of the ODB block is to be stated, which defines the 2D- and 3D-geometries of the article in the ODB tables. Then, the OFML type stated in field 2 has to

be directly or indirectly derived from *OiOdbPlElement*. The ODB name then will be saved by this type in an invisible property of the article object under the key `@ODB_NAME`.
Field 3 is empty, if the geometry is not described by an ODB block, but by the OFML type itself. This is possible only with OFML part III (language).

- The article parameters in field 4 are used if the OFML type can represent multiple base articles, but the base article number itself is not directly saved with the article object. Then, which base article the article object currently represents, follows from some other variables of current internal object state. Variables, which allow for the correlation with a base article, are designated as *article parameters*. This can be member variables and/or OFML properties. Every possible combination of values of these article parameters corresponds to exactly one base article. These combinations are to be stated in field 4.
- The ODB name stated in field 3 is a special form of an article parameter (see Fig. 3). That means, that no further article parameters are required in field 4, if every base article is mapped to an unique ODB name.

  The following two examples illustrate this topic:
  Example 1: Different articles use a common ODB geometry, but have different article parameter sets.

  ```
  854;::foo::bar::PlElement;::foo::bar::T8xx;W=@W1,UA=@U
  855;::foo::bar::PlElement;::foo::bar::T8xx;W=@W2,UA=@U
  884;::foo::bar::PlElement;::foo::bar::T8xx;W=@W1,UA=@A
  885;::foo::bar::PlElement;::foo::bar::T8xx;W=@W2,UA=@A
  ```

  Example 2: Different articles use different ODB geometries. No further article parameters are required.

  ```
  854;::foo::bar::PlElement;::foo::bar::T854;
  855;::foo::bar::PlElement;::foo::bar::T855;
  884;::foo::bar::PlElement;::foo::bar::T884;
  885;::foo::bar::PlElement;::foo::bar::T885;
  ```

  Thus, the general requirement is, that the table entries allow for an unique mapping of base article numbers (field 1) on the one hand to OFML type, ODB name and article parameters (fields 2-4) on the other hand. The reverse mapping must be likewise unique, if the base article number is not saved with the article object or if the saved base article number is not provided by a respectively overridden implementation of method *getArticleSpec()* of OFML interface *Article*[3]. In this case, the standard implementation of this method uses the mapping in order to determine the base article number based on the current object state.
- Article parameters (field 4) can be stated and processed in two different ways:

  1. *List of parameters* (field type *Params*)
     This is the standard declaration, an example is stated above. The assignment of parameter values is automatically performed by the OFML runtime environment immediately after assignment of base article number via the standard method *setArticleSpec()*. At this point, the article object has to possess an according OFML property for every article parameter. Thereby, the property key has to comply with the identifier stated in the list of parameters. (Thus, the parameter identifier "`W`" from the example above corresponds to a property with key `@W`.)
     The assignment of article parameter values then is be done via method *setPropvalue()* of OFML interface *Property*.
     An error will occur if the property type does not comply with the data type of the value stated in the list of parameters. It is not defined, whether the OFML runtime

[3]In case of doubt a biunique mapping must always be ensured.

environment proceeds with the assignment of possibly following article parameters, or whether the operation will be cancelled. In either case, the article object then will be in an invalid state.

For a given combination of OFML type (field 2) and ODB name (field 3), the article parameters have to be stated in the same order and quantity.

2. *Coding* (field type *Char*)

   If a specific OFML type is used, which was developed by means of OFML part III, it can implement its own coding for article parameters. This is accomplished via various methods described in OFML part III. In this case, field 4 has to start with the character ”`#`”, which itself does not belong to the coding of the article parameters.

3.4 The Article–ODB-Parameter Mapping

Table name: `article2odbparams`
Mandatory table: no

Defines the values of help variables for a given article configuration, which are used in ODB data for the graphical representation of the article.

Motivation:
As evident from Fig. 3, in ODB data there can be used variables, which refer to properties of the article object. During the evaluation of the ODB data by the OFML runtime environment, these variables will be replaced with the actual value of the property. Usually it is sufficient to use variables within ODB data, which directly refer to an individual commercial property of the article (for which an according OFML property was generated at the article object). However, in some cases it makes sense to use help variables, which are derived from multiple commercial properties, or in other words, refer to a specific article configuration (article variant).

Thus, this table has to be employed for a given article only if its geometry is described by ODB data *and* if the ODB data uses additional help variables transcending ordinary commercial properties.

Nr.	Name	Key	Typ	Length	Obligatory	Explanation
1.	ArticleID	X	Char		X	base article number
2.	VarCodeType		Char			type of variant code
3.	VariantCode		Char			variant code
4.	OdbParams		Params		X	ODB parameters

Notes:

- For a given base article (field 1) there may be one or more entries in the table. Each entry defines a separate set of ODB parameters (field 4).
- The variant code (field 3) defines the configurations (variants) of the article, for which the parameter set has to be applied.
- The variant code can be stated in various forms. The used form has to be specified in field 2. Currently the following forms are supported:

  FS The article configurations, which belong to a parameter set, are specified by means of the **final article specification**. A stated final article specification may be partly defined, i.e. not relevant characters may be replaced by the question mark ”`?`” and/or irrelevant characters after the last relevant character may be omitted. Thus, a partly defined final article specification refers to multiple configurations of the article.

For the current (complete) final article specification of a given article object the OFML runtime environment then determines the (best) matching table entry. If there are multiple matching entries, the entry with the highest count of matching characters will be used.

This form cannot be used, if the relevant commercial properties are not encoded in the final article specification. Partly defined final article specifications cannot be used, if the relevant commercial properties are encoded at varying positions within the final article specification (depending on the current article configuration).

For the future additional forms are planned not suffering from the restrictions stated for the form **FS** above.

- Fields 2 and 3 can be empty, if there is only one set of ODB parameters for the base article, independent of its configuration. In this case the table contains only one entry for the base article.
- For each ODB parameter in the list of parameters (field 4), the OFML runtime environment generates an according OFML property, which then is available for access by ODB data.

3.5 The Property–Material Mapping

Table name: `property2mat`
Mandatory table: no

This table defines, which property values correspond with which material layer assignments.

The motivation for this mapping is similar as for the Article–ODB-parameter mapping (see sec. 3.4). Material layers can be viewed as specific ODB help variables, which are used in the material field of the ODB table for 3D data. Conceptually, by specifying a variable in the material field (instead of specifying a concrete material) the respective ODB component will be assigned to the material layer defined by the variable (e.g. variable `SEAT` for layer ”seat surfaces”). When building and displaying the 3D graphics of an article, the OFML runtime environment then assigns the current value of the layer variable to the material of the respective ODB component (see also Fig. 3). This mapping now defines, which commercial properties affect which material layers, and for which values of these commercial properties which materials have to be assigned to the affected layers.

Nr.	Name	Key	Type	Length	Obligatory	Explanation
1.	ArticleID	X	Char		X	base article number
2.	PropertyID		Char			property identifier
3.	PropValueID		Char			value identifier
4.	MatLayer		Char		X	name of material layer
5.	Material		Char			material identifier

Notes:

- Fields 2 and 3 contain the (language independent) identifiers of the property resp. the property value, for which a layer assignment has to be accomplished.
- For a given triple of base article number, property identifier and value identifier (fields 1-3) there can be multiple table entries, each of which refers to a different layer (field 4). This is necessary, if a commercial property is operating multiple layers.

  Example:
  A chair has the construction groups *seat* and *backrest*. Within upholstery type *single–colored*, seat and backrest shall have the same material. Within upholstery type *multi–colored*, different materials may be assigned. Relationship knowledge in the commercial

data is used to control various color properties depending on the value of commercial property "upholstery type": property "color seat/back" will be used for upholstery type *single–colored*, properties "seat color" and "backrest color" will be used for upholstery type *multi–colored*. As seat and backrest may have different materials, different layers are necessary for both construction groups (`SEAT` and `BACK`). Then, within upholstery type *single–colored*, property "color seat/back" (identifier `COL1`) has to control both layers:

```
ABC123;COL1;V600;SEAT;LV600
ABC123;COL1;V600;BACK;LV600
ABC123;COL1;V601;SEAT;LV601
ABC123;COL1;V601;BACK;LV601
ABC123;COL1;V602;SEAT;LV602
ABC123;COL1;V602;BACK;LV602
...
```

- Field 3 (property value) can be empty or contain the character "*". In this case field 5 will not be evaluated, but the actual value will be assigned to the layer.

  In the above example, if the materials with the same name were stated as the property value, the table could be as follows highly simplified:

  ```
  ABC123;COL1;*;SEAT;
  ABC123;COL1;*;BACK;
  ...
  ```

- If there is only one existing material assignment for a material-layer of a base article (independent of any commercial properties), fields 2 and 3 have to stay empty. Per base article and layer there is permitted only one such entry.
- If there are common layer assignments for multiple base articles, they can be stated in a common table entry, containing character "*" in the first field. However, if there are explicit entries for a base article, the layer assignments stated there have priority.
- Material names (field 5) can be stated fully qualified. If they are not qualified or not fully qualified, the OFML runtime environment prepends the name of the package, which contains the ODB-data.