Interactive PMCube Explorer

Documentation and User Manual

Thomas Vogelgesang
Carl von Ossietzky Universität Oldenburg

October 27, 2016
Contents

1 Introduction 3

2 Application Overview 4

3 Data Preparation 6

3.1 Data Warehouse Schema 6
3.2 Creating a Meta-data File 9

3.2.1 Case Dimensions 9
3.2.2 Event Dimensions 11
3.2.3 Fact Table 11
3.2.4 Case Table 11
3.2.5 Event Table 13
3.2.6 Semantics 14

4 Application settings 16

4.1 General Settings 16
4.2 Paths 16
4.3 Visualization 16
4.4 Process Model Diff 17
4.5 Process Discovery 17
4.6 Plugins 17

5 Connect to Database 18
1 Introduction

The Interactive PMCube Explorer is a tool for multidimensional process mining (MPM). This manual gives a brief introduction of its usage and the necessary preparation of event data. For a general introduction of MPM and the concept of the PMCube approach, we refer to [VA16]. For this document, we assume that the reader is familiar with process mining.
2 Application Overview

Figure 2.1 shows the application’s main window after start-up. In general it consists of four parts: the ribbon menu at the top, the content area at the center, and two sidebars left-hand and right-hand of the content area.

The ribbon menu provides access to most of the features of the application which are grouped in different tabs by their semantic. The six main tabs are:

**Start** This tab offers general features like connecting the application to the database (see Chapter 5) and customizing the application settings (see Chapter 4).

**Data Integration** This tab provides tools that support the integration of data into the data cube like the editor for meta-data files (see Section 3.2).

**View** On this tab, you can change the view of the application, e.g., turn on/off the process model matrix view or hide/show particular tabs.

**OLAP** This tab provides all OLAP-related features like loading the cube and its data from the data warehouse.
2 Application Overview

Event Log  On this tab, you find all features to manipulate (e.g. filter) the event logs that were loaded from the data warehouse.

Process Mining  This tab offers advanced process mining techniques like process enhancement and conformance checking.

Consolidation  This tab provides access to the consolidation of process models.

Tip: You can hide/collapse the ribbon bar by clicking the icon on the right side (below the window close button) or double-clicking one of the tabs.

The sidebar on the left side of the main window comprises all configuration options for the OLAP query. Grouped by different tabs, you can roll-up/drill-down or slice the data cube. Furthermore, you can also omit unnecessary attributes from the query results.

In the sidebar on the right side of the main window, you find the selection and configuration of the process discovery algorithm, the selection of the classifier, and the operation history where you can undo/redo the analysis steps.

The content area in the center of the window uses a tab system to present the analysis results. The Olap Cell Preview tab shows a preview of the currently set OLAP query while the Olap Results tab presents an overview of the results of the last executed OLAP query.

Tip: The main window is based on a docking framework. Therefore, you can customize it according to your needs, e.g., by changing the size of or rearranging the different parts of the window.
3 Data Preparation

The Interactive PMCube Explorer manages the event data required for process mining in a multidimensional data cube that is stored in a relational data warehouse. Currently, the following database management systems (DBMS) are supported:

- Microsoft SQL Server (recommended)
- Oracle DB
- PostgreSQL
- MySQL

However, it is recommended to use Microsoft SQL Server which was primarily used for testing and application so far. Before analyzing processes with Interactive PMCube Explorer, the event data has to be integrated into the data warehouse. Section 3.1 explains its expected database schema.

3.1 Data Warehouse Schema

In Interactive PMCube Explorer, there are two different ways to model the case and event attributes.

**Dimension** A dimension represents an axis of the data cube that can be used to partition the event data into sublogs. Each dimension consist of a set of pair-wise distinct values. Furthermore, Interactive PMCube Explorer allows the user to define dimension hierarchies in a tree-like structure. Dimensions should be the preferred way of modeling the attributes. However, they may not be appropriate in case of many rare attribute values as can this result in sparse data cubes and missing representativeness of the process models. To avoid such problems, sets of different attribute values may be grouped into artificial categories, e.g. representing intervals of values. If this is not meaningful, the attributes can also be stored as simple attributes.

**Simple attribute** This way of modeling directly attaches an attribute to its case or event. In contrast to dimensions, simple attributes are very limited in their usage (e.g., they cannot be used for roll-up and drill-down). Therefore, it is only recommended to use simple attributes for avoiding the loss of data if the attribute cannot be meaningfully mapped to a dimension.
The database schema manages case and event attributes on different levels. This is not only reflected in the definition of OLAP operations, but also by the database schema that stores cases and events in different tables. Figure 3.1 shows the generic database schema of the data warehouse which is an extension of the traditional snowflake schema commonly used for relational data warehouses.

Similar to the snowflake schema, there is one table for each level of a dimension hierarchy and the tables of a dimension are connected by a 1:n foreign key relationship according to the hierarchy (e.g., tables $D_1.K_0, \ldots, D_1.K_m$). The Fact table references case dimension (i.e., each dimension representing a case attribute) by a foreign key reference to the table of its most fine-grained dimension level. Additionally, it also stores a unique id for the easy identification of cell of the data cube. Table 3.1 gives an overview of the columns of the Fact table.

Using a foreign key, the cell id of the Fact table is referenced by the Case table which stores each single case as a row. Each tuple requires a mandatory unique id to identify
the cases. Additionally, the Case table can also have an arbitrary number of additional attributes ($A_1, \ldots, A_p$) to store the simple case attributes (i.e., all case attributes that are modeled as simple attributes). The simple attribute columns of the Case table may have arbitrary types. Table 3.2 gives an overview of the columns of the Case table.

The Event table stores one event per row. Similar to the Case table, it references the tables of the most fine-grained dimension levels of all event dimensions (i.e., event attributes that are modeled as dimensions). The simple event attributes (i.e., event attributes modeled as simple attributes) are represented by an additional table column for each simple attribute (similar to Case table). The Event table also has a unique id and a foreign key reference to the id of the case that owns the event. Table 3.3 summarizes the columns of the Event table.

The structure of a dimension level table depends on whether the dimension level is the highest (i.e., most coarse-grained) level of the dimension. If it is not the highest dimension level, the table has a unique id column (used for referencing), a value column (holding a representation of the attribute’s value), a description column (providing a more verbose textual description of the value), and a parent_id column holding a foreign key reference to the id of the parent dimension level value. The latter can be omitted for the highest level of a dimension, because this does not have a parent. Table 3.4 gives an overview of the columns of a dimension level table.
### 3 Data Preparation

<table>
<thead>
<tr>
<th>Column Name(s)</th>
<th>Type</th>
<th>Cardinality</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>arbitrary</td>
<td>1</td>
<td>id of dimension value, must be unique</td>
</tr>
<tr>
<td>value</td>
<td>arbitrary</td>
<td>1</td>
<td>representation of dimension level value</td>
</tr>
<tr>
<td>description</td>
<td>varchar/varchar2</td>
<td>1</td>
<td>textual description of the dimension level value</td>
</tr>
<tr>
<td>parent_id</td>
<td>arbitrary</td>
<td>0 . . . 1</td>
<td>foreign key reference to the table of parent dimension level, can be omitted if no parent exists</td>
</tr>
</tbody>
</table>

Table 3.4: Columns of Event table

### 3.2 Creating a Meta-data File

To be able to load the event data from the database, Interactive PMCube Explorer requires some meta-data on the database schema which is provided by an XML-based meta-data file. To create this file, Interactive PMCube Explorer provides a meta-data editor that can be opened from the data integration tab (see Figure 3.2).

![Figure 3.2: Meta-data editor icon](image)

The editor window provides a number of different tabs to edit the meta-data file which are explained in the following in more detail.

#### 3.2.1 Case Dimensions

On the Case Dimensions tab (see Figure 3.3), all dimension levels and their relationships need to be defined. Dimensions can be added by typing the dimension’s name into the text input field on the left bottom and clicking on add dimension. The dimension and its logical root (ALL) are added to the tree-view on the left. After selecting the dimension node, one can change its name and its data type (default is string).

Selecting the ALL node and a right-clicking on it opens a pop-up menu that allows to add a new dimension level as a child of the selected node. After selecting a dimension level in the tree, one need to define the properties as specified in Table 3.5.

**Note:** To create a dimension hierarchy, the dimension levels must be specified from top to bottom (starting with the most coarse-grained dimension level).
### Property | What to enter
--- | ---
Dimension level name | Name of the dimension level (as it should be displayed in the tool)
Dimension level table name | Name of the database table representing the dimension level
Id column name | Name of the column holding the id
Value column name | Name of the column holding the value
Description column name | Name of the column holding the description
Foreign key to parent id column name | Name of the column holding the foreign key to the parent table (can be left empty if no parent table exists)

Table 3.5: Properties of dimension level

Figure 3.3: Meta-data editor (Case dimensions)
3 Data Preparation

3.2.2 Event Dimensions

On the Event Dimensions tab, one needs to specify the event dimensions and their dimension levels. The workflow and the properties are similar to the Case Dimensions tab (see Section 3.2.1).

3.2.3 FactTable

On the FactTable tab, one has to specify the table name of the Fact table and the name of the column that holds the cell id (cf. Figure 3.4). Furthermore, one needs to specify the foreign key relationships of the Fact table to the tables representing the dimension levels. Clicking the Add button opens a new dialog to create this mapping (cf. Figure 3.5).

![Figure 3.4: Meta-data editor (FactTable)](image)

In the first field, one needs to enter the name of the column holding the foreign key to the dimension level table. For the second filed, click on the right-hand button and select the dimension level that is referenced by the foreign key from the dialog. Confirm the mapping by clicking OK.

3.2.4 CaseTable

On the CaseTable tab (Figure 3.6), you need to specify the structure of the Case table by entering the following information:
3 Data Preparation

Figure 3.5: Meta-data editor (attribute editor)

Figure 3.6: Meta-data editor (Case table)
3 Data Preparation

1. The table name of the Case table
2. The name of the column that holds the case id (primary key)
3. The name of the column that holds the cell id (foreign key to Fact table)
4. For each simple case attribute stored in the Case table, enter the column name and data type of the attribute and press Add

3.2.5 EventTable

On the EventTable tab (Figure 3.7), you must specify the structure of the Event table. Enter the following information:

![Figure 3.7: Meta-data editor (Event table)](image)

1. The table name of the Event table
2. The name of the column that holds the event id (primary key)
3. The name of the column that holds the case id (foreign key to Event table)
4. For each simple event attribute stored in the Event table, enter the column name and data type of the attribute and press Add
5. For each event dimension, create a mapping between a dimension and the corresponding foreign key column in the Event table.

6. Select the attribute that should be used to order the events (usually the time-stamp).

7. Select the attribute that should be used as the default classifier (usually the activity).

8. For each simple event attribute stored in the Event table, specify the aggregation function that should be used for event aggregation.

### 3.2.6 Semantics

On the Semantics tab (Figure 3.8), you must specify the semantics of particular attributes (similar to a concept in XES) – especially for the time-stamp. Click *Add* and enter the following information in the dialog:

1. The attribute that should have the semantic
2. The concept the selected attribute should have

   For the time-stamp, enter a format string required for parsing the time-stamp which describes the time-stamp format of the attribute in the database. The format string must be defined according to the C# time format specification ([https://msdn.microsoft.com/de-de/library/8kb3ddd4(v=vs.110).aspx](https://msdn.microsoft.com/de-de/library/8kb3ddd4(v=vs.110).aspx))
3 Data Preparation

Figure 3.8: Meta-data editor (Semantics)
4 Application settings

Before starting the first analysis, it is recommended to configure the tool. To customize the application settings, click on the Application settings button in the start tab. The application settings dialog showing up provides multiple settings on different pages. The following sections will give a brief overview of the available configuration options.

Note: Changes of some particular settings will only take effect after a restart of the application.

4.1 General Settings

On this page, you can select the configuration file for the logger. Interactive PMCube Explorer uses the Apache log4net library\(^1\) (a .NET port of the log4j library). To enable the logging, create a configuration file (see log4net documentation for details) and select it in the application settings.

Additionally, you can also set maximum tree depth of XML documents handled by the serializer. However, the default value should be fine in most cases.

4.2 Paths

On the Paths page, you can configure the default paths for the import and export files (e.g., process models, images), a default folder for storing meta-data files, and the default folder for the setting files of process mining algorithms. Furthermore, you have to specify the file that will store your database connection information.

4.3 Visualization

On this page, you can customize the appearance of the process model visualization (e.g., node size, distances between nodes etc.). Most of the settings are specific for a particular process model type. To edit the appearance of a specific view model, you can select the corresponding process model notation from the drop-down list.

\(^1\)https://logging.apache.org/log4net/
4.4 Process Model Diff

On this page, you can select the algorithm that should be used to create difference models.

4.5 Process Discovery

Here, you can select the default process discovery algorithm. This algorithm will be automatically selected on application start-up. Optionally, you can also select a configuration file to provide customized default settings to the selected default algorithm.

4.6 Plugins

The Plugins page gives you an overview of all plug-ins that have been loaded at the start-up of the application. To search for a particular plug-in, the text field on the top of the page allows you to filter the plug-in list.

Tip: By default, the tool provides multiple plug-ins. Therefore, an empty plug-in list indicates that the path to the plug-in folder is not correctly set. To fix this issue, you can select the plug-in directory and confirm it by clicking the Okay button. Now, the plug-ins should be loaded when restarting the application.
5 Connect to Database

At the beginning of your analysis, you need to select the connection to your database. To open the connection dialog (Figure 5.1), click on the Connect button of the Start tab.

![Connection Dialog](image)

**Figure 5.1: Connection Dialog**

To create a new database connection, select the used database management system from the drop-down list and click on the plus icon. After that, enter the required parameters in the dialog. Table 3.2.1 explains the parameters in more detail.

For connections to Microsoft SQL Server databases, there is also the option to use the automatic authentication of the Windows operating system. You can check the checkbox to enable this option. User and password field can be left empty when Windows authentication is used. You can save the settings using the Save icon in the toolbar on top of the dialog.
5 Connect to Database

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection name</td>
<td>The user-defined name of the connection which is used in the connection list on the left. Should be unique.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the target database</td>
</tr>
<tr>
<td>Server</td>
<td>The URL or IP address of the target database server</td>
</tr>
<tr>
<td>Port</td>
<td>The port of the target database service</td>
</tr>
<tr>
<td>User</td>
<td>The name of the database user</td>
</tr>
<tr>
<td>Password</td>
<td>The password of the database user</td>
</tr>
</tbody>
</table>

Table 5.1: Parameters of the connection dialog

**Note:** The database configurations are saved in a file (for selecting this file, see Section 4.2) which is encrypted using the Windows user account. Consequently, the connection file can only be read from the user account you created it. Sharing connection information between different user accounts or machines is not possible.

Finally, you need to select the meta-data file that describes the structure of the selected database. To connect to the database, select the desired connection from the list and click on **OK**. Now the application initializes the dimension values from the database. Depending on the size of the data and your connection speed, this may take some time.

**Tip:** The application automatically saves the last used database connection and meta-data file. To reconnect quickly, you can click on the **Quick connect** button in the Start tab.