

Concept of configurable filters for Visual Data Mining Systems

Jörn Seger, Sefan Michaelis
Department for Electronic Systems and Switching
Faculty for Electrical Engineering and Information Technology
University of Dortmund, Germany
joern.seger@udo.edu, stefan.michaelis@udo.edu

Abstract

Visual Data Mining (VDM) is an upcoming issue on traffic engineering and on network management in general. The parameter sets in analysing network data are becoming more complex and even more in number. This causes the problem of how to display this data and how to hint on possible problems.

The VDM-Concept deals on the one hand with the fact of increasing the amount of data. On the other hand, it will open up the possibility even for non-experts to understand the elaborated information that are available within the data.

The VDM consists of a pipeline with three elements: Filtering, Mapping and Rendering. Within this paper we want to describe the filtering as the main element of VDM. We will present a concept, how data could be analysed via filtering and what should be done to implement a VDM-filter within the INTERMON environment.

A VDM-Filter is a set of sub-filters, that could be chained to produce a complex VDM-Filter on the need of the viewer. The chaining of filters has to be done via graphical user interface and should be discussed here as well.

Another main topic within the filter concept is the handling of Data Objects. These Data Objects traverses the sub-filter chain, and contains the all data, that is produced within the traversed sub-filters. Every sub-filter is able to access every data that was produced by previous filters. This sub-filter connection must also be managed by the graphical user interface.

This article mainly concentrates on a generalised and configurable design for a filter VDM system. The design is not tied to any specific application, but allows to adopt to many tasks which demand for a data mining and user feedback loop.

The design will be used for network analysis in the INTERMON project ([1]), which should be seen here as an example application for the VDM concept.

1 Visual Data Mining for Network Analysis

The classical Data Mining approach from the artificial intelligence research field was lately extended with the visualisation aspect. This introduced a human viewer centered approach to use enhanced graphical methods for the presentation of the data mining output and give the opportunity to interact with the data mining filters.

Two concepts are integrated into the general VDM approach: The feedback based knowledge discovery process and the visualisation concentrated filtering, mapping and rendering pipeline. The first process is based on the cross-industrial software process model for data mining (CRISP-DM). This CRISP-DM model ([2]) was developed as a data independent guide for knowledge discovery in huge amounts of data.

The second concept is derived from models for data preparation and visualisation ([3]). Main idea here is to process the raw data through a pipeline (F-M-R), where each stage performs the functionality to present the data in an innovative manner. At the end of the pipeline the viewer controls the output.

• Filtering:

The Filtering stage initiates the process to transform the data for visualisation. In practice this often means to reduce the amount of data as most visualisation techniques are limited to a certain number of parameters. Additionally tests have proven that human beings are only capable to cope with eight different parameters at once.

Beside simple filters to reduce data complexity more enhanced algorithms are thinkable, which could extract useful knowledge from the raw data. This features is described in the next sections.

• Mapping:

The stage of Mapping leads the viewer to select an appropriate visualisation model. The model selection

could be based on the complexity of the data (e.g. 3D models can display higher dimensional data than 2D models) or the expressivity of the visualisation (e.g. line charts are useful for time series representation, where a pie chart would be nearly useless).

- **Rendering:**

The last stage of the pipeline, the Rendering stage, is responsible to draw the visualisation directly on the screen. Main difference to the former stages is the machine dependence of this unit. As the former stages depend on logical modelling of the data, the rendering uses the native methods on the viewers side to display the models.

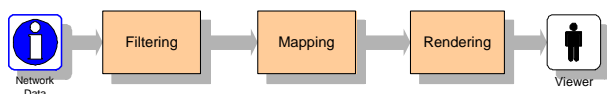


Figure 1. Filtering-Mapping-Rendering pipeline

Figure 2 shows the combination of the classical data mining approach for knowledge discovery with the visualisation pipeline concept.

The lifecycle of the classical DM process starts with the business understanding. As data mining is often used as a service for knowledge discovery in non-computer science related areas, deep knowledge of the field of application is useful to select data as DM input.

Understanding the business directly leads to understanding the data: What data can be retrieved?

Independent and Dependent Variables

To have a basic structure within the available more dimensional data sets, the parameters are always divided into two categories:

- **independent variables:**

n independent variables open up a n -dimensional space. In most cases, time is one independent variable, because there is always one information available at a specific time. Another independent variable might be a router ID.

- **dependent variables:**

Dependent variables are values, that appear with a defined set of independent variables. The most simple dependent variable is given by $y = f(x)$. Here the variable y is dependent on the independent variable x .

To determine a dependent variable, every independent variable must be set to a defined value. This is comparable to a n -dimensional array, where the index of every dimension must be set to get the dependent value or the group of dependent values.

In a normal form, the independent variables have a mapping between an ordinal counter and the collected value of this independent variable (e.g. time 07:14:45). In a two dimensional array, the data could be interpreted in the following form:

ordinal counter	⇒	1	2	3	4
		Router ID			
↓	time	4382	4527	3922	4571
1	05:23:20	12	24	456	845
2	05:23:25	45	67	213	784
3	05:23:30	0	23	462	353
4	05:23:35	12	51	159	369

Here the time and the router ID are the independent variables. At every router and at every timestamp, a parameter is measured, which is the dependent variable in this scenario.

Impact of variables

The filter concept for visual data mining is going to be placed in the context of the INTERMON project. As the goal of INTERMON is an architecture for quality of service (QoS) analysis of inter-domain network traffic, available data mainly consists of link traces of different quality (e.g. considering missing values or time scale).

Network traffic analysis could for example deliver data with the following parameters:

- **Time**→ in most cases time will be seen as the independent variable.
- **Delay**→ dependent variable.
- **Loss**→ dependent variable.
- **QoS**→ QoS classes could also be considered as independent (at least from time), so could be used to distinct different flows.
- ...

These parameters need to be considered for further survey using the visualisation and data mining algorithms. This example here should show the impact of variables (parameters) available for input to the filtering and visualisation models. The selection of one or more independent variables as a reference point (e.g. mapping the independent variables to the axis of the chart) for the rendering is vital to allow the user a valuable evaluation of the filter output.

Data Preparation

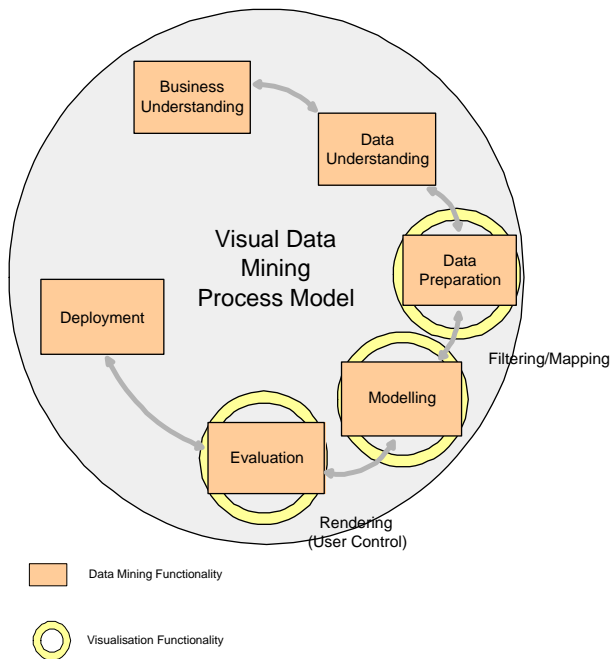


Figure 2. CRISP-DM model adapted to Visual Data mining

After the data understanding, most interesting steps for the filter design start with the Data Preparation (refer figure 2). Here the F-M-R concept can be integrated with the classical DM approach to enhance the interpretation of the data mining output.

The data preparation process in the context of data mining is very similar to the classical (i.e. statistical in this context) filtering to prepare data sets for visualisation. Input for the data mining algorithms should not be the pure, raw data, but a filtered data set (e.g. handling missing parameters).

Modelling builds the step in applying the DM algorithms on the data (the so called training sets). The output data should be directly applied to the modelling stage of the F-M-R pipeline, which is the direct stage before bringing the visualisation on the screen of the viewer.

Last stage which combines data mining with the visualisation pipeline is the evaluation phase. Visualisation strongly relies on the human aspect for evaluation of the filtered and mined data. Here it is necessary to allow the human viewer to observe the DM output visually and adjust parameters which produced the displayed results. This leads to a feedback loop, refining the filtering process from iteration to iteration, to get better results during each loop. Even though, the viewer is able to concentrate on special

aspects within every loop.

The very last stage of the process model, the Deployment, delivers the final results to the public.

The following sections concentrate on the filter approach and discuss a generic design for the application on many heterogeneous tasks. Main recommendation for the design is to be flexible enough to fit most tasks for visually guided knowledge discovery and to be independent from most formats of data. The design is actually under survey in the INTERMON context for QoS inter-domain traffic analysis.

2 Filter Classification

The generic design for VDM filters is independent from the specific filter applied to the data. Nevertheless it is useful to classify possible filters regarding their complexity and handling.

Statistical Filter

Statistical filters are the building blocks of each filtering approach, providing a well known field of research for the data analysis. Statistical filters could be classified by different characteristics: Input parameters and the specific task to perform.

The input parameters have to suit the statistical filter, e.g. for time series trend analysis having one independent variable (time) and other parameters depending from the variable (delay...).

In most cases the viewer expects the filter to perform a specific task with a specific outcome and then evaluates the filtered data based on expert knowledge. Performing a specific task in this context means, the viewer expects a certain behaviour and output when using the filter (thinking of trend filters or frequency analysis...).

Data Mining Filter

In contrast to statistical filters the usage of data mining filters often has different pre-requisites and output. Data mining, also often called knowledge discovery or learning theory through research, is used to detect *hidden* knowledge. This means to find structures in the data not expected and not obviously to see.

Many different types of DM filters can be differentiated: Algorithms which detect dependencies between parameters, find clusters or patterns inside the data etc. Most of these algorithms are based on, combine or use statistical filter for the data preparation. The full range of algorithms available is out of scope of this paper, the interested reader can find good introductions in [4] and [5].

Important for the context of the here proposed filter architecture is the *generalisation* aspect. The application of

filters of different types has to be supported. In some cases the application of more than one filter is necessary (see section 3), which leads to an interconnection of different filters (DM and non-DM filters).

3 Filter Chaining Concept

The Filter Chaining Concept was designed to build a complex VDM-filter out of a chain of subfilters (refer figure 3). These subfilters will be traversed successively, so that data, which was produced within a previous filter could be used in a later one.

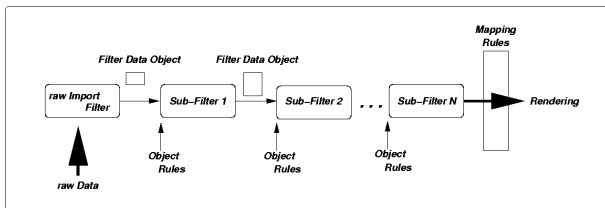


Figure 3. VDM-Filter with integrated sub-filters

Every sub-filter is an object, that is derived from a basic *Generic Data Mining Filter*, which is abstract and fulfils all common filter features for chaining. One especially derived filter is the *Raw Import Filter*. It is an abstract object as well and will be used to add communication abilities to import data from a data source.

To store all data that were imported from a database or produced by a filter, a *Filter Data Object* is defined. This object consists of a list of items (so called *Filter Data Entries*), which represents the data that was produced by one filter. The number of items increases with the number of sub-filters that the Filter Data Object traverses.

To be able to use all prior produced data with a specific sub-filter, a *Filter Object Rule* specifies, how data from the *Filter Data Object* is mapped to the input data of this sub-filter.

Data Types

To specify a value within a scalar or a table, the type of this value must be known. This type is presented as a string and is defined as follows:

Type	associated string	aggregatable	not aggregatable
integer	int	x	
floating point	float	x	
time	time	x	
string	string		x
non aggregatable int	nagint		x

Filter Data Entry

The item within a Filter Data Object is called "*Filter Data Entry*" and contains the following information:

- **Name:**
A *Filter Data Entry*, that was produced by a subfilter, has to be unique within the environment. Therefore every entry must have a name for identification.

To show the connection of a data entry and the sub-filter, that produces this data, the name is set in the following form: `<Filter name>:<Data Name>`. The `<filter name>` is the unique name of the filter, that produces the data set and the `<data name>` is an arbitrary unique string. This string will help the viewer to connect data in a more comfortable way.

- **Scalars:**
Scalars are used to store single values, e.g. maximum/minimum of a range of values.

A scalar consist of

- an unique name for identification,
- a type to define the data (refer *Data Types*), that is stored within the scalar and
- the value itself.

- **Tables:**
Tables are used to store n-dimensional data. The most simple table will consist of one independent and one dependent variable.

Every table is able to contain multiple independent and multiple dependent variables. Important is, that every collection of independent variables must have a full set of dependent variables.

Parallel Chaining

Another point on chaining is, that sub-filters or sub-filter chains could be branched and connected again. The filter rules will then be traversed logically in parallel and the Data Object will be doubled at conjunctions, where two filters want the same Data Object. At conjunctions, where a branch will be merged again, the Data Objects will be concatenated.

A parallel chain is just a visualisation tool to have more clarity. The branching of data streams is not necessary, because all available data could be processed even in a one dimensional filtering chain.

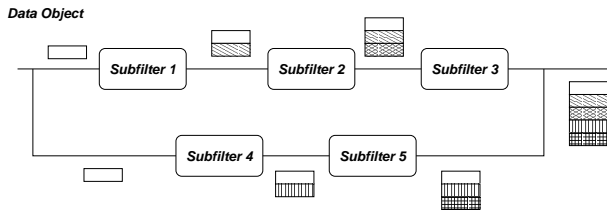


Figure 4. parallel Chains within the VDM

4 Chaining via Graphical User Interface

As mentioned before, every sub-filter has an *Filter Object Rule*. This rule specifies, how a parameter (e.g. scalar or table) within a *Filter Data Object* is connected to a requested input parameter of the next sub-filter.

To be able to connect both parameters, the viewer needs to have a graphical interface. This interface has to be intuitively and should prevent the viewers from linking unlinked values (refer figure 5).

The most important thing is to downgrade a table in dimension to fit to the input table. This could be done by fixing one or more independent parameters to a constant value. In contrast, dependent parameters could stay unlinked.

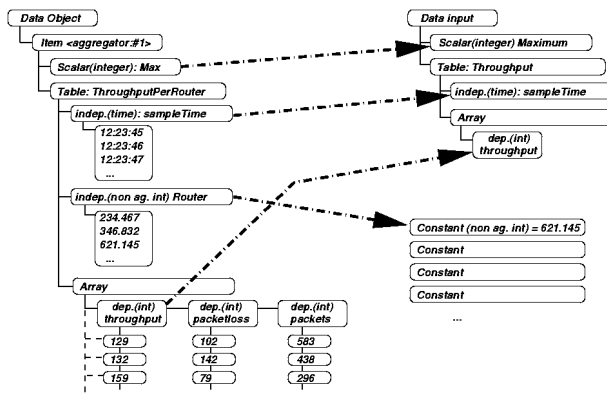


Figure 5. GUI to connect filter data with the input data of the next sub-filter

Within this scenario, the values shown here are not relevant for linking and will not be displayed in the real Filter Object Rule GUI. Before the viewer is able to connect the dependent variables, every independent variable of one table has to be connected. In figure 5 the connections are shown by arrows. In a graphical environment an indication by colour will be more effective.

The constant values have to be set due to the *Filter Data Object* information. This means, that they have to be specified by hand or by the GUI on connection time. Within the

filtering process, the constant values could be changed, if the constant is mapped to an adjustment panel.

5 Conclusion

The work on Visual Data Mining for network information has just started. Many issues have to be considered to build an expressive and effective environment for visualisation. The main elements within this environment have been identified as filtering, mapping and rendering. As the main objective even in visual data mining, filtering has been described in detail and it has been illustrated how it will be used within the INTERMON scenario for Quality of Service management and advanced traffic engineering.

Further research will analyse the generic filter concept in practical applications. The generic design of course leads to an administrative overhead in contrast to the specification of filters and data formats applied to a single, well defined task. But as the process of analysis of complex data demands for the involvement of the human for configuration and evaluation, a flexible, modular and therefore easier to adjust F-M-R pipeline enhances the quality and speed of the analysis.

This paper shows some of the problems arising along with a more flexible design, describes how to cope with them and integrates the ideas into the whole process of visual data mining.

References

- [1] INTERMON consortia: *Advanced Architecture for interdomain Quality of Service Measurement, Modelling and Visualisation*, IST-2001-34123, <http://www.ist-intermon.org/>
- [2] P. Chapman et al.: *The CRISP-DM Process Model*, The CRISP consortium, 1999
- [3] Haber, R. B.; Mc Nabb, D. A.: *Visualisation idioms: A conceptual model for scientific Visualisation Systems*. In: Shriver, B.; Nielson, G. M.; Rosenblum, L. (eds): *Visualisation in Scientific Computing*. IEEE-Computer Society Press, Los Alamitos, 1990, P.74-93
- [4] Witten, I.; Frank, E.: *Data Mining — Practical Machine Learning Tools and Techniques with Java Implementations*, Morgan Kaufmann, 1999
- [5] Fayyad, Usama (Editor), *Advances in Knowledge Discovery and Data Mining*, AAAI Press, Menlo Park CA 1996