Published in:
Wohlgemuth, V.; Page, B.; Voigt, K. (2009): 23rd International Conference Informatics for Environmental Protection, Environmental Informatics and Industrial Ecology, Environmental Informatics and Industrial Environmental Protection: Concepts, Methods and Tools, Shaker Verlag, Aachen.

Odysseus – A Service for Distributing Dynamic Multi-layered XY Charts on the Internet

Alexander Burtscher, Michael Haase, Gernot Belger

Björnsen Beratende Ingenieure GmbH, Maria Trost 3, D-56070 Koblenz, Germany,

email: {a.burtscher | m.haase | g.belger }@bjoernsen.de

Abstract

In the context of environmental observation the visualization of data in charts (e.g. time series plots) plays an important role. Observation data are often visualized as xy charts. In contrast to the ongoing standardization in the web distribution of spatial data and their visualization as maps no standard for distributing xy charts over the internet exists. In this paper a general approach for a service for visualizing observation data as xy charts is presented.

1. Introduction

Digital sensors are increasingly being used in the context of environmental process observation (e.g. run-off gauges, rainfall gauges, emission of gases from industrial firing processes). Standards for accessing these data are available from the Open Geospatial Consortium with the *Sensor Observation Service* (OGC 2007). It is beyond question that visualization of these data plays an important role in this context.
Existing software for creating charts (e.g. *JFreeChart* 2009) is restricted to a limited set of chart types, e. g. line or column charts. The possibility to freely mix different graphical data representations in one chart is commonly not supported. Also, the possibilities of altering the presentation of a chart are rather limited, e.g. only one type of axis representation is commonly supported.
When it comes to web distribution, services like *Google Chart API* (Google 2009) or *OWTChart* (Map-Tools 2009) enable a dynamic creation of charts on request on the basis of the hypertext transfer protocol. Besides the afore mentioned limitations related to the visual chart representation, the data visualized with these services need to be delivered to the service within the request. This method requires an automated access of the data which is not always granted by the data provider. The data provider merely wants to remain in control of who can access his data.
These facts show the need to design and implement a special software for generating xy charts which can be easily configured according to the user's requirements. This software shall be able to collect data from distributed data stores and be accessible over the internet. This paper introduces to such a software.
The paper is set up as follows: The requirements for the software are outlined in chapter 2. In chapter 3 the technical approach and details related to the implementation are given. Application examples of the software are presented in chapter 4. In chapter 5 a summary of the paper is presented and future developments are discussed.

2. System Requirements

2.1 Objectives

The following main objectives were identified: The environmental data shall be depicted in xy charts. These charts shall be generated dynamically in order to include up-to-date observation data provided by other services, including those with proprietary interfaces. The application shall be accessible from the internet, it shall also be embedded into rich client applications. It shall be possible to visualize a mixture of different data sets in one chart in order to enable comparisons between these data sets (e.g. gauging data from different locations) or to present different data with a common reference in a single chart (e.g. precipitation data and runoff data) to meet special needs for information. Each data set shall be presented within the chart as a layer which can be turned on and off on demand.

2.2 Use Cases

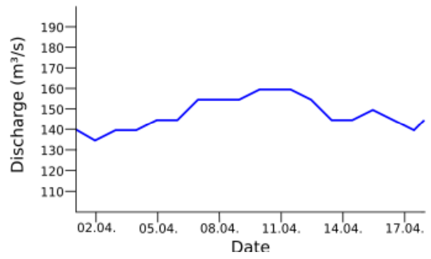


Fig. 1: Gauge chart

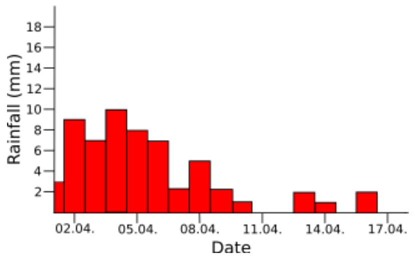


Fig. 2: Rainfall chart

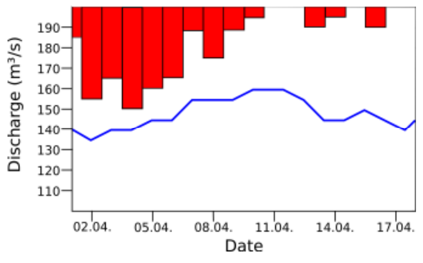


Fig. 3: Rainfall gauge chart

From the numerous use cases identified in the design process an example is quoted hereafter:
Observation data which are to be visualized are provided by two different data stores: Gauge data are collected from a service using a proprietary protocol, rainfall data are collected from a *MySQL* database server. The data shall be visualized as depicted in Figures 1 to 3. Figure 1 depicts a chart for visualizing runoff gauge measurements (line chart), Figure 2 shows a chart for visualizing rainfall measurements (column chart). Figure 3 depicts a combination of rainfall and gauge measurements within a common time period (combined line and column chart) which enables the user to compare runoff and rainfall directly. By default, these charts shall present data for a period of 14 days prior to the current date. The user shall be able to further limit this time span according to his needs.

2.3 User Roles and Actions

Three types of user roles are distinguished for the application: The **chart provider** defines charts in a chart file which is selectable for publishing. Charts are defined by combining and customizing chart elements from a catalog. The **chart user** defines, selects, views and alters charts with a graphical user interface. The user can alter charts according to his individual needs by defining the chart's size, hiding layers and setting data ranges displayed in the chart. The **chart developer** contributes new implementations of chart elements to the catalog if this is requested from chart users.

2.4 Input Data

Different types of data shall be presented in xy charts to users. These include, but are not limited to, time series data (e.g. runoff or rainfall time series) as well as polynomial functions and geometries (e.g. river profile data).
In terms of data visualization, these data types can be looked at as a set of components (dimensions) each representing a physical quantity, e.g. meters above sea-level, time or amount of rainfall in millimeters.

Two of these components are mapped to the layer coordinate system: The domain component is mapped to the x-axis, the target component is mapped to the y-axis.
Since data for visualization are provided by different data stores, interfaces for accessing these (potentially proprietary) data stores are required. Also, these interfaces may deliver semantically identical data types within different data models. It must therefore be assured that the application is not restricted to limited sets of interfaces and/or data models.

2.5 Technical Requirements

Different graphical user interfaces shall be provided. The implementation of the chart generation shall be used in desktop applications based on the *Eclipse Rich Client Platform* (Eclipse 2009) as well as HTML based internet applications.

2.6 Chart GUI

From Figures 1 to 3 it can be clearly deduced that each chart consists of different graphical elements. The main elements are axes and graphs. Of course, instead of implementing all three charts from scratch, it will make more sense to set up charts by combining them from a given set of elements.
The chart GUI generates a graphical user interface which contains input elements for selecting, viewing and altering charts. The main purpose of this visualization is to enable the user to easily understand complex contents. In addition to the chart display a legend is required which contains a name and possibly a short description for each graph presented in the chart.
The chart GUI shall provide means to the user to easily understand possibly complex contents presented in the charts. The legend is comprised of pairs of symbols and a corresponding description which are called legend items. The legend also represents the layer structure of the chart. It groups legend items belonging to a certain layer into entities, e.g. line charts with marked nodes. Furthermore, the chart GUI shall allow the user to apply methods to alter the chart according to his individual needs.

3. Technical Approach and Implementation

3.1 Chart Model

The multi-layered xy chart (further referenced to as chart) model is oriented towards the composition of several data visualizations. It consists of multiple layers and axes. Layers contain a graphical representation of data sets and are drawn in the plot. Each layer represents a coordinate system which is comprised of two orthogonal axes.

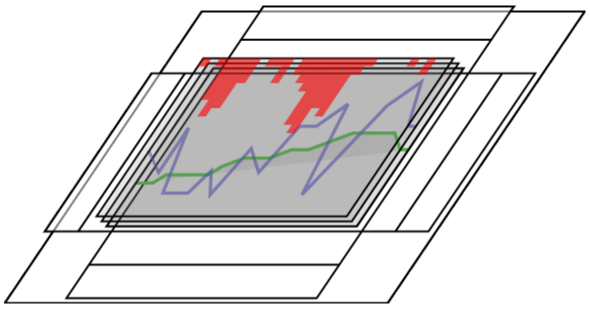


Fig. 4: Schematic view of a multi-layered chart

Each axis represents a data component. If two layers visualize data sets which contain components with identical physical units, these layers can "share" the axes representing these components in a chart. The latter allows for overlaying different data sets in order to provide easy means for a visual comparison of these data.

Layers provide a list of symbol-description pairs (legend items) which may be utilized for building a chart legend for the user's comfort. The layer coordinate systems may be defined by setting the position, direction and data range of each axis. Axes can be positioned on all four sides of the plot. Each layer can be independently hidden or shown. Figure 4 shows a schematic view of a chart, Figure 5 depicts the chart elements and relations between them in a UML diagram.

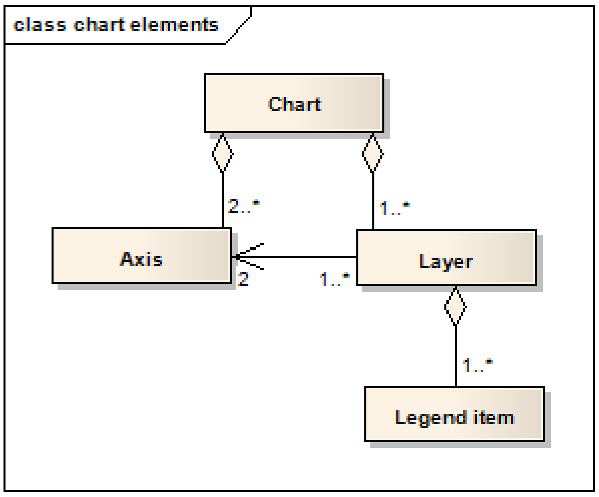


Fig. 5: Chart elements and relations

3.2 Chart Service

The chart service has been named *Observation Diagram Service* (*ODS*). The way dynamically generated charts are distributed resembles the way thematic maps are distributed by the *Web Mapping Services* (*WMS*, OGC 2004) from the Open Geospatial Consortium (OGC). A thematic map shows geographic features within a geo-spatial bounding box. Features belonging to a certain theme (e.g. streets, houses) are grouped into layers. Clients can request maps offered by the service. The user can choose which contents are displayed on the maps by hiding layers which he is not interested in as well as by defining the geographical extension of the displayed map by setting a bounding box.
In the design process of the *ODS* the *WMS* specification was used as a master specification which was adapted to the special requirements of the *ODS*. The *ODS* enables clients to create graphical user interfaces for selecting, viewing and altering charts. It generates images of charts and symbols and provides information on how to build up the user interface and how to access these images.
When a client requests a certain chart, the service creates the chart on the basis of a chart definition – contained in a XML-based chart file - by collecting all data to be presented and displaying these data in a chart. The service is stateless. Therefore, for each request the whole chart creation process must be rerun. This even holds true if identical requests are made.

The *ODS* provides three operations:

- The *GetCapabilities* operation returns information about the service and its offerings. It does so by returning a *service metadata document* which contains sections about the institution running the service, the charts offered by the service and meta data about operations to access the offered charts. The chart offerings contain the structure of each offered chart, including the available layers, the position and orientation of the contained axes and their initial data interval. It also lists available symbols for each layer along with a description. The symbol information is used to fetch symbol images with the *GetSymbol* operation in order to build a legend.
- The *GetChart* operation returns a raster image of a chosen chart. The image output is determined by several parameters: clients can define the size of the chart image, the layers to be included in the image and the data range for each axis of the chart.
- The *GetSymbol* operation returns a raster image of an icon for each visual element inside the layer. These icons can be used to build up a legend for the chart.

3.3 Implementation

The implementation of the *ODS* is called *Odysseus* and is available as open source code from SourceForge (http://kalypsobase.sourceforge.net), the code is released under the GNU Lesser General Public License. Odysseus is implemented as a servlet (Sun 2004) and can be used in servlet containers like Tomcat or Jetty. The *OGC Web Service* specification (OGC 2007-2) was applied for designing the interface of the OdysseusService. The WMS is also based on this specification.
As available charting software is lacking support for the chart model presented in section 3.1, a software component called *OdysseusChartFramework (OCF)* was developed. This component is implemented on the basis of the *Eclipse Rich Client Platform*; it provides an extensible and highly configurable engine to create charts from an XML file. Implementations of chart elements are contributed as *Eclipse* extensions. Apart from being applied in *Odysseus, OCF* can also be integrated in Eclipse-based products. A special graphical user interface was implemented for embedding the *OCF* into desktop applications which are based on Eclipse.
The chart client for web applications has been implemented on the basis of a two-tier architecture. It is implemented in *PHP* and runs on all *PHP* enabled web servers – e.g. Apache (Apache 2009). A web browser as front-end for accessing the charts is required.

4. Application Examples

Two products were developed on this basis which will be presented hereafter. A chart GUI for *Eclipse*-based desktop applications is depicted in Figure 6. It consists of two parts: the chart part displays graphs (right hand side in Figure 6), the legend part displays the corresponding legend (left hand side in Figure 6) allowing to select / deselect chart layers for graphical display with check boxes from a set of charts.

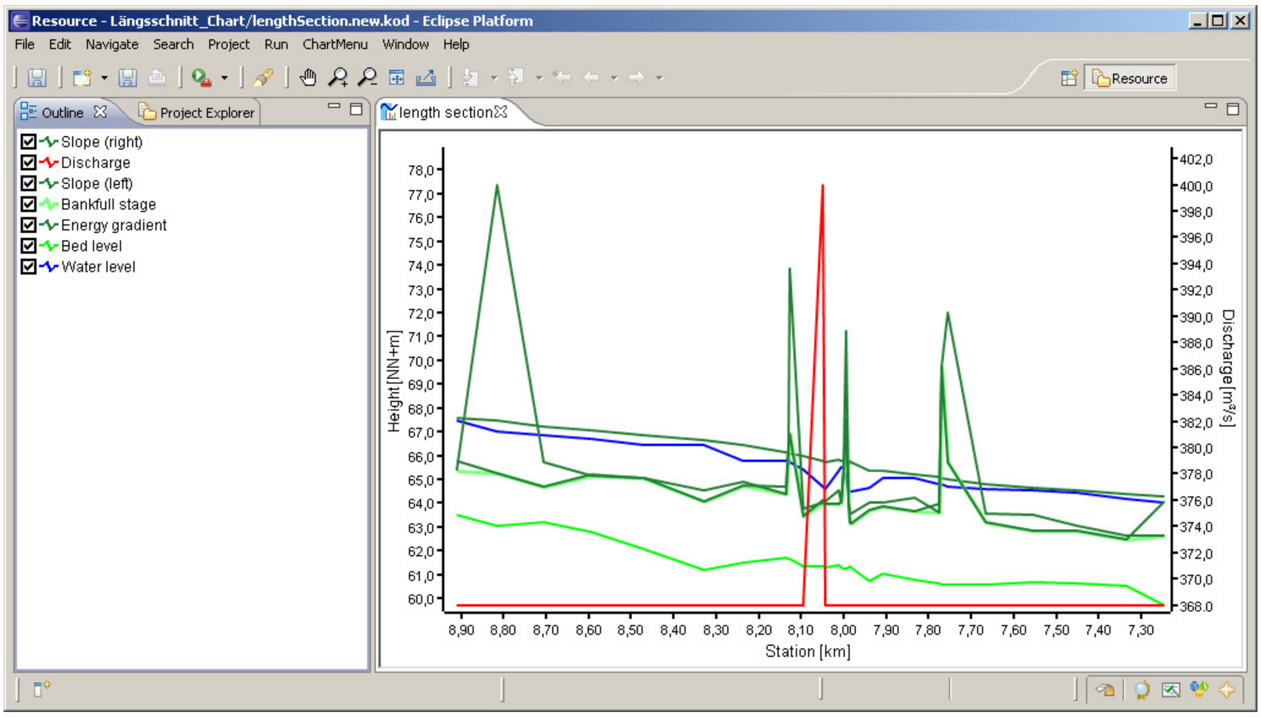
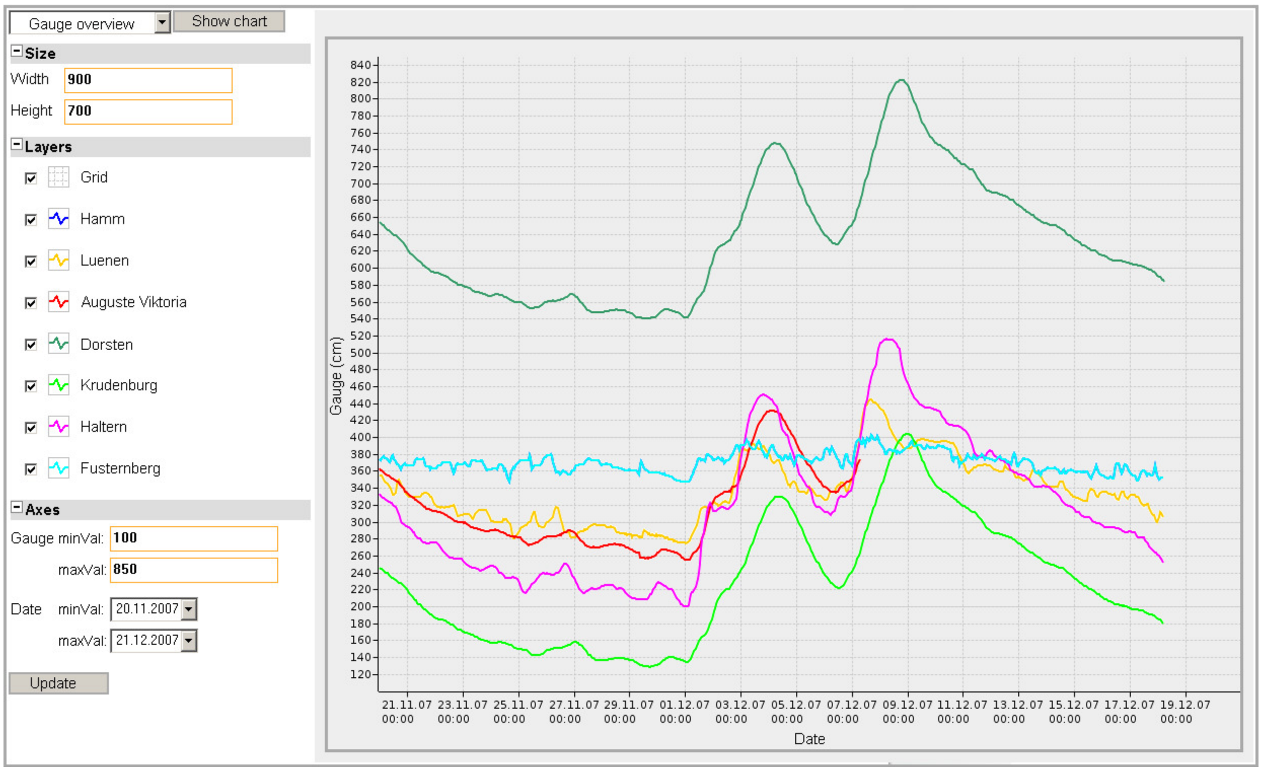


Fig. 6: Chart GUI built for Eclipse-based desktop applications

The chart shown in Figure 6 displays a longitudinal section of a river depicting parameters which are varying along the course of the river, e.g. riverbed elevations and runoff. The user can zoom and pan in the chart using his mouse. The overlay sequence of layers can be changed via drag-and-drop operations in the legend view; layers can be displayed / hidden by selecting / deselecting check boxes in the legend view. Furthermore, the user can export the currently displayed chart into an image file.
Figure 7 depicts a chart (right hand side) from a web client which displays gauging data from different river gauging stations. The user requests a chart from a web browser which is then dynamically generated in the background on the server and delivered to the client be displayed in the user's browser. He may choose a chart using a select box from a predefined set of charts. The depicted viewport of the chart is defined via input fields and / or selected from select boxes (minimum / maximum values); check boxes in the legend section allow for selecting layers which shall be displayed in the chart (left hand side in Figure 7).


Gauge overview

Size
Width: 900
Height: 700

Layers
☑ Grid
☑ Hamm
☑ Luenen
☑ Auguste Viktoria
☑ Dorsten
☑ Krudenburg
☑ Haltern
☑ Fusternberg

Axes
Gauge minVal: 100
maxVal: 850
Date minVal: 20.11.2007
maxVal: 21.12.2007

Fig. 7: Chart client for web applications

5. Summary and Further Work

To this date no standard is available for charting of e.g. sensor data. A common interest is assumed from the side of the developers of distributed environmental information systems for generating and offering environmental data charts over the internet. The design and a reference implementation of such a service is presented in this paper.
The presented service *Observation Diagram Service* in conjunction with the *Sensor Observation Services* resembles an analogy to the conjunction between *Web Mapping Services* and *Web Feature Services*. A cascading chart service on the basis of *Odysseus* is planned for the future. In this scenario a chart would embed pre-rendered chart elements collected from other chart services. The current implementation of the *Odysseus* can be easily extended to meet this new requirement.

The *Odysseus* source code and a chart example can be downloaded from the SourceForge page of the *KalypsoBASE* project (KalypsoBASE 2009). It can be used to create line and column charts from CSV formatted data. As the service implementation builds upon the presented chart framework, it can be easily extended to meet other user requirements.

6. References

Apache (2009): http://httpd.apache.org/
Eclipse (2009): http://www.eclipse.org/
Google Chart API (2009): http://code.google.com/intl/de-DE/apis/chart/
JFreeChart (2009): http://www.jfree.org/jfreechart/
Jetty (2009): http://www.mortbay.org/jetty/
KalypsoBASE (2009): http://kalypsobase.sourceforge.net
OGC (2004): "OGC Web Map Service Implementation Specification", Project Document OGC 06-042, Version 1.3.0
OGC (2007): "OpenGIS Sensor Observation Service", Project Document OGC 06-009r6, Version 1.0
OGC (2007-2): "OpenGIS Web Service Common Implementation Specification", Project Document OGC 06-121r3, Version 1.1.0
OGC (2009): http://www.opengeospatial.org/
OWTChart (2009): http://www.maptools.org/owtchart/
Sun (2003): "Java™ Servlet Specification", Version 2.4
Tomcat (2009): http://tomcat.apache.org/