Lehrstuhl fuer Systeme und Informationsverwaltung

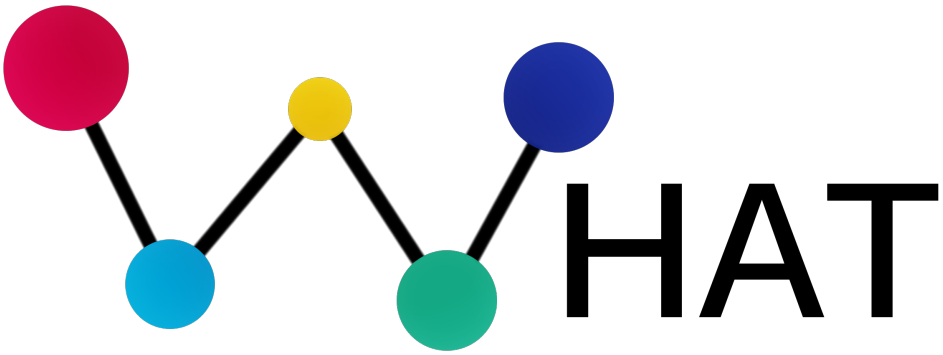
Prof.Dr.Ing. Klemens Boehm
Hoang Vu Nguyen
Marco Neumann

Karlsruher Institut für Technologie

Praxis der Softwareentwicklung (PSE)
WS 2012/2013

Visualizing and Statistically Analyzing Access Behavior to Scientific Databases

Functional Specification



WareHouse Analyzing Tool

November 18, 2012

Functional Specification **Alexander Noe**
Design **Jonathan Klawitter**
Implementation **Anas Saber**
QA / Testing **Niko Moraitakis**
Final **Lukas Ehnle**

About this document

Why this specification is written

PSE is a mandatory course for students of Bachelor Informatik in the Karlsruhe Institute of Technology (KIT). Therefor groups of five to six students are formed to write programs of 'medium size' (about 5000 Lines of Code).

This specification is the one of PSE Group #10 in the winter semester 2012/13.

What this specification does

The purpose of this document is a outline of the functional specifications and requirements for the WHAT application. Is tries to give a complete and exact model of the future system. In the course of this it wants to give the developers answers to all possible question concerning what they have to implement.

What this specification does not

This specification does not tell how the system should be implemented. Also this is not a project plane.

A 'Living Document'

Software development is a dynamic process. So with proceeding development the content of this specification will change continuously.

Skyserver

The concept of the system described in this specification should work with any database storing it's query-log. The SkyServer will function as example and testing reference for this system.

Questions and comments

If you have any questions or comments regarding this document feel free to send us an E-Mail.

Introduction

PSE is a mandatory course for students of Bachelor Informatik in the Karlsruhe Institute of Technology (KIT). In this course we have to form groups of five or six people to write a program of 'medium size' (~5000 Lines of Code).

Our task given in this course consists of analyzing and visualizing the server log of the SkyServer. This is one of the biggest public databases for astronomical data. It contains pictures and informations of astronomical data and tries to form a 'map of the universe'.

This assignment is one of two assignments which will be handled in English. Because no one of us speaks English as his first language, we want to apologize that our documentation may contain simple or in some parts incorrectly used language. Our focus lies more on having easy-to-understand and correct documentation than on perfect English, which looks like written by a native speaker.

We are going to write a web-frontend in javascript allowing users to access the finished application from everywhere with recent browsers. The actual application can be spliited into two different parts. The first part - which will be referred as 'Parser' and is going to convert the CSV-formatted serverlog from Skyserver into a data-warehouse. It will be accessible via admin-login to the webpage where we can enter a logfile, which will be parsed into our warehouse. The second part - which will be referred as 'Analyzer' works on this data-warehouse and allows every user of our webpage to create various charts, for example scatterplots and histograms. Due to the fact that we split our project in two smaller parts, many parts of this specification are going to be splitted in two.

We have to state that no one of us has much experience in working with databases and javascript. Therefore we can't guarantee the correctness of all information stated in this specification. Some of the specifications may be altered when we design or implement the actual program.

Contents

1 Goals

With this programm the user should be put in the position to visiualize the prepared data of queries, runned against his data base.

1.1 Parser

1.1.1 Core criteria

- The Parser can read a CSV-formatted log file from skyserver.sdss.org and fill a data-warehouse with multi-dimensional user information. The dimensions of this data are location of the user, location of the database and time of the server-access. It will also add several measures such as delay time or
- The Parser will recognise invalid logs and won't add them to the data-warehouse. Every log with a mistake won't be accepted, because an error-message is not as bad as a corrupted data-warehouse.
- The Parser will be accessible with admin-login to the webpage.

1.1.2 Optional criteria

- The parser will contain a small manual which is designed for experienced users. (optional because this parser is only used by it's developers)

1.1.3 Exclusion criteria

- The Parser can only use CSV-formatted logs from skyserver.sdss.org. It can neither read logs in another format nor logs from another source.
- There is no way to avoid using this Parser when adding data to the warehouse. This can stop corrupting the warehouse to guarantee correct data in the warehouse.
- The Parser doesn't correct mistakes in the log file.
- Due to being only a tool for it's developers, our parser doesn't need to be easy to use for first-time users.

1.2 Analyzer

1.2.1 Core criteria

- The analyzer can use the data-warehouse which is filled with user data from skyverser.sdss.org to create various charts. Those charts are created by user command and are visible on a javascript-website.

  - scatterplots
  - histograms
- The analyzer can take filter information via webpage from the user to use only certain data for creating charts.

1.2.2 Optional criteria

- The analyzer will support different chart-types.

1.2.3 Exclusion criteria

- WHAT takes correct data in the warehouse for granted. It doesn't check the data because it already checked by SLP.

1.3 Website

1.3.1 Core criteria

- The website has an graphical interface in javascript, so that users can enter which chart they want.
- The website contains small guides for experienced users on how to create a specific chart. (similar to the guides at skyserver.sdss.org)
- The website has an admin-login page to get administrative rights, which are needed to use the parser.

1.3.2 Optional criteria

- The language of the website can be changed into different languages. (e.g. German)

1.3.3 Exclusion criteria

- The website got no function for normal users to enter data into the data warehouse.

2 Usage

2.1 Applications

Wat.

2.2 Target groups / Audience

The target group of this application are people that want to analyze and visualize data collected from the skyserver. This implies

1. People that know about the skyserver. Since the skyserver only allows queries via sql and has an arcane website this greatly reduces the prospective audience. We do not expect the webserver to run into scaling issues.
2. People that are interested in what other people are using the web server for. Our project mainly visualizes sql queries from other people. Knowing sql, while not a prerequisite, would allow the user to fully utilize the software.

This is a technical audience.

That said, this does not prevent the (web) user interface from being functional, usable and prettier than what you would expect a group of computer science students to design.

2.3 Operating conditions

The program is mainly used as a website, with the primary difference being that the server has to be started if the capacity for it to run all the time on a dedicated machine doesn't exist. The program needs a server to run.

If a dedicated server exists, the program can be used from anywhere with a decent network connection with the server.

If not, the program can still be run on the same computer as the server (on localhost), but the server will have to be started first.

3 Operating environment

Whereas the programm and the data warehouse run on a sever, the access to it will be via web browser on a workstation computer. The web page making this access possible needs a hosting web server too.

3.1 Software

- The server hosting the data warehouse needs Oracle SQL / MySQL. (?)
- The server hosting the web page needs a recent version of th Java Runtime Environment / Java VM

This software is required on the workstation computer:

- Latest version of a modern browser
  - Chrome or Firefox will be supported
- JavaScript (enabled)

3.2 Hardware

The server has to be fast enough to support all clients. This depends on the expected number of clients. Most computations will be done on the server.

The client needs to be fast enough to visualize the data received from the server. The required hardware thus depends greatly on the amount of data that needs to be visualized. That said, any recent computer, with for example, an Intel® Core™2 Duo CPU E8400 @ 3.00GHz × 2 processor, 4GB(2x2GB) of 667Mhz DDR2 SDRAM, running Ubuntu Linux 12.10 Quantal Quetzal, as a point of reference, should be able to visualize about 50.000 data points on a scatterplot with ease.

3.3 Orgware

The server needs to be able to connect to the client with a reasonable latency. Also it has to be set up before the client will be able to connect to it.

To fulfill optional requirements, the programm musst be able to run queries against the origin database.

3.4 Product interfaces

To fulfill optional requirements .csv files have to be imported.

4 Functional requirements

4.1 Main functions

This functions are required to fulfill the core criteria.

General

/F10/ Provide access via web page

/F20/ Provide options for chart types

/F30/ Show diagrams

/F40/ Show histograms

/F50/ Provide option for the two variables of the histrogram

/F60/ Provide option for the interval or selection of the x-axis variable

/F70/ Show scattered plots

/F80/ Provide option for the two variables of the scattered plot

/F90/ Provide option for the intervals or selections of the two variables

/F100/ Show bubble charts

/F110/ Provide option for the three variables of the scattered plot

/F120/ Provide option for the intervals or selections of the two variables

/F130/ Show information about chart types

/F140/ Show information about selectable variables

Adminstrator specific

This functions are required for the adminstrative business.

/F150/ Provide access via web page with administrator rights

/F160/ Provide the oppertunity to initialize the data warehouse *

/F170/ Provide the oppertunity to pass log-files to the parser

/F180/ Provide the oppertunity to clean the data warehouse

Parser specific

The following functions specify the parses functionality. Thereby /FXU/* to /FXZ/* specify the function /FXT/.

/F190/ Extract specific data from the log-files

/F200/ Extract access database, access time and user information (dimensions)

/F210/ Extract number of rows, elapsed time, busy time (measures)

/F220/ Extract type of data requested from the `where` part

/F230/ Transform the data to fit into the data warehouse schema

/F240/ Load the data into the data warehouse

Analyzer specific

/F250/ Run specific queries against data warehouse

/F260/ Build up diagrams with received data

/F270/ Build up histrograms

/F280/ Build up scattered plots

/F290/ Build up bubble charts

4.2 Extending functions

To fulfill the optional goals the following functions are required.

/F300/ Select language on web page

/F310/ Switch language to German or other

/F320/ Build up and show bubble map

/F330/ Build up and show other chart types

/F340/ . . .

5 Data

The product data are divided in two groups, static data, which are delivered with the program and Data-Warehouse data, which come from the Skyserver server logs, are parsed and loaded into the Data-Warehouse by the SLP. They can be replaced by clearing the Data-Warehouse and loading new data into it with the SLP.

5.1 Static Data

/D10/ Language files

/D20/ Manual

/D30/ Source Code

/D40/ Documentation

/D50/ Graphics for GUI

/D60/ HTML backbone

/D70/ Javascript files

/D80/ Stylesheet files

5.2 Data-Warehouse data

/D90/ Database accessed by Users

/D100/ Server accessed by Users

/D110/ City of Users

/D120/ Country of Users

/D130/ Time of Skyserver Access

/D140/ Number of rows accessed

/D150/ Elapsed time of access

/D160/ CPU busy time of access

6 Nonfunctional requirements

6.1 First Whatever

/NF10/ blabla

/NF20/ tadaa

6.2 Next Whatever

/NF30/ 30!

/NF40/ 40!

7 Global test cases

7.1 Test Case

/T10/ Start internet browser /F10/

/T20/ Type the web address - say pse10-analyzer.edu - in browser’s address bar

/T30/ Call the Home Page with a Table Bottun /F20/

/T40/ Select a Type of chart /F30/

/T50/ Call the diagram Page

/T60/ Call histogram /F40/

/T70/ Call the option for tow variables of the histrogram /F50/

/T80/ Call the option for the interval or selection of the x-axis variable /F60/

/T90/ Call scattered plots /F70/

/T100/ Call the option for tow variables of the histrogram /F80/

/T110/ Call the option for the interval or selection of the x-axis variable /F90/

/T120/ Call scattered plots /F70/

/T130/ Call the option for tow variables of the scattered plots /F80/

/T140/ Call the option for the interval or selection of tow variables /F90/

/T150/ Call bubble charts /F100/

/T160/ Call the option for three variables of the scattered plots /F110/

/T170/ Call the option for the interval or selection of tow variables /F120/

/T180/ Leave the web page

8 Models

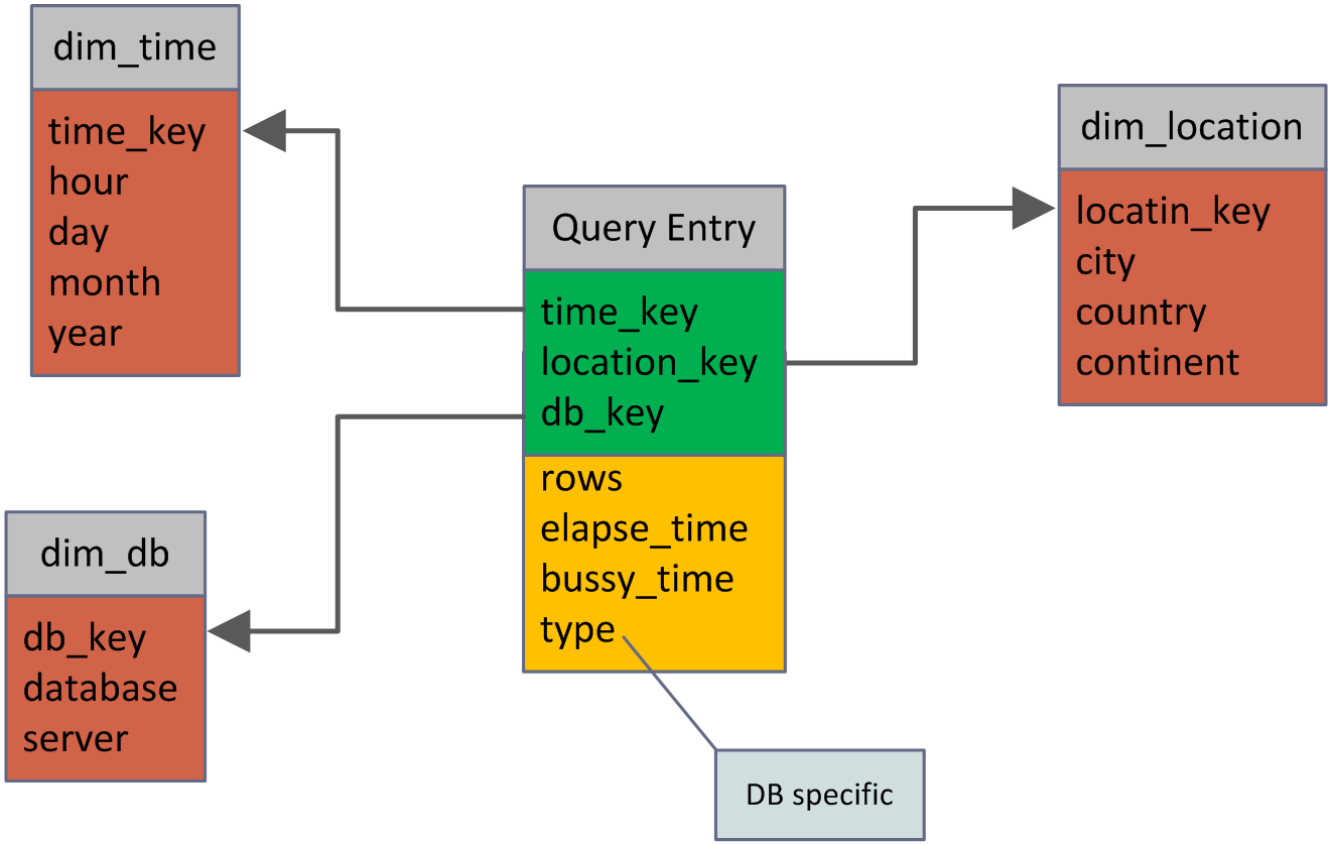
8.1 User stories

8.2 Object models

8.3 Dynamic models

8.4 Web interfaces

8.5 Warehouse model



9 Development Environment

Operating System

- Windows 7
- Fedora 17
- Whatever Niko uses - EDIT THIS, NIKO!

Object Models

- Microsoft Visio

Version Control

- Git

Miscellaneous

- Github (Hosting of Repository, Issue Tracking)
- Travis (Integration)
- LaTeX (Documents)
- Gradle (Building)

10 Glossary