

Maintenance Documentation

i-scream Historical Database Reporter

Data collected from the i-scream monitoring system may be stored in a database for subsequent analysis. The i-scream *historical database reporter* is used to produce graphical representations of data over time and meta data of such data, enabling users to make use of the data in many possible ways. This document provides an overview of the structure and mechanisms involved within the historical database reporter itself.

Revision History

| | | | |
|----------|------------------|--------------------|-------------------|
| 28/03/01 | Initial creation | Committed by: pjm2 | Verified by: tdb1 |
| | | | Date: 28/03/01 |
| | | Committed by: | Verified by: |
| | | | Date: |
| | | Committed by: | Verified by: |
| | | | Date: |
| | | Committed by: | Verified by: |
| | | | Date: |
| | | Committed by: | Verified by: |
| | | | Date: |

- Introduction 2
- Producing builds..... 2
- Providing configuration 2
- Running chronology 2
 - Initial checks 2
 - Preparations 2
 - Database connection..... 2
 - Enabling extremely large queries 3
 - Obtaining machine data for a report period..... 3
 - GIF image generation..... 3
 - Writing meta data files 3
 - Tidying up 4
- Generated output 4
 - Charts 4
 - Meta data..... 4
 - day.inc..... 4
 - report.inc 4
 - machines.inc 4
 - i-data.txt 4
 - i-maxmin.txt 4

Introduction

Data collected from the i-scream monitoring system may be stored in a database for subsequent analysis. The i-scream *historical database reporter* is used to produce graphical representations of data over time and meta data of such data, enabling users to make use of the data in many possible ways. This document provides an overview of how the historical database reporter works and is complementary to the javadoc generated pages.

Producing builds

Builds of the historical database reporter are automatically added to the i-scream web site overnight, along with all other builds using the same Makefile.

Providing configuration

The user may alter the contents of the *DBReporter.properties* file in order to specify their database connection string, database driver, etc. The default contents of this file work as-is, due to the MySQL driver in *mm.mysql-2.0.4-bin.jar* being present as part of the build.

Details of the exact contents of this properties file may be found in the user documentation for the historical database reporter.

The file *reportlist.conf* is the report list that is provided as part of the build. This contains all standard report attributes are collected by a typical i-scream monitoring system. The formatting of the contents of this file may also be found in the user documentation.

Running chronology

Initial checks

The main method of the historical database reporter is found in the file *DBReporterMain.java*. When this method is invoked, the command line arguments are processed. The settings for the program are then read from the properties file specified by the user. The reading of the settings is handled by *DBReporterSettings* and this object is also used to store the settings for use throughout the program.

The program exits if the properties file does not specify a valid directory.

Preparations

The start and end times for the reports are calculated. These are stored in the database as the number of seconds since the epoch (GMT). By default, the settings make the program produce reports for the day previous to the current day. Please read the user documentation for details of how to set this report period.

The *DBReporter* object is created with details of the report period. Before this is used, a *ReportList* object is created. This reads in the list of reports from the conf file that was specified in the main properties file. A reference to this object is passed to the *DBReporter* so that it may know which reports to produce.

Database connection

When the *runReports(ReportList reports, String webDir)* method in the *DBReporter* is invoked, a connection is made to the database. This same connection is used throughout the

lifetime of the running program. If this connection cannot be established, then the program exits with an error message.

Enabling extremely large queries

As we may be producing extremely large query results, we need to ensure that there is sufficient space available for this. By default, during a select, MySQL uses system memory to write temporary tables. This is often of insufficient size, so we have to instruct MySQL to write these temporary tables to disk. This slows down the issuing of select statements, however, it needs to be done. This is done by issuing the following statement: -

```
SET SQL_BIG_TABLES = 1;
```

If this could not be set successfully, the user is notified and the program continues. The program will exit if there becomes insufficient memory available to the MySQL daemon.

Obtaining machine data for a report period

A query is issued to select distinct machine names for the report period. A *ReportMaker* is used to produce reports for these machines using the *produce(reports, machine)* method.

The *ReportMaker's* *produce* method creates a LinkedList of *PlotData* objects. Each row from the database is fed to these *PlotData* objects, which automatically average and scale the data as it is added. The *ReportMaker* issues the query to the database to select all rows that relate to the machine within the report period. This takes some time, so each report for an individual machine is generated at the same time. This reduces the number of queries made as well as reducing the number of *XMLPackets* to make. Each row from the database supplies an XML string that is parsed into an *XMLPacket* by an *XMLPacketMaker*. For each value corresponding to a value in the *ReportList*, we add the date and attribute value to the *PlotData* object. Non-numerical values are ignored and not added to the *PlotData* object.

GIF image generation

For each full *PlotData* object, we output a GIF chart to: -

```
[output directory]/[date]/[machine name]/[report name]/i-chart.gif
```

The contents of the meta data files contained in `[output directory]/[date]/[machine name]/[report name]` are also generated at this point using the *ChartDataWriter*.

An *IScreamChart* is used to create a *BufferedImage* representation of the data contained within the *PlotData* object. An *IScreamChartWriter* is then used to write the GIF image to disk. This makes use of the Acme GIF encoder package.

Writing meta data files

To assist the production of a web-based interface to the historical reports, a file named *report.inc* is created in each `[output directory]/[date]` directory. This contains an option list of each report name and friendly report name in HTML format. In the same directory, a file named *machine_name.inc* is also created. This contains an option list of all machine names for this report date.

A single file named *day.inc* is created in the root of the output directory. This contains PHP mark-up to specify the date of the last generated report.

Tidying up

When all reports have been created, the *DatabaseRemover* is used to delete all rows that are older than the DeleteOffset from the start date of the report period.

Generated output

Charts

Each report for each machine produces a GIF image that represents the average (mean) value for the report attribute over time. These are stored in a hierarchical directory structure of the form: -

```
[date]/[machine name]/[report name]
```

This directory structure allows the use of third party tools to allow easy navigation and summarisation of the reports. An example of such a tool has been produced by i-scream, and allows viewing of historical data via a web server. This tool is part of a collection of web utilities that is available from the *Builds* section of the i-scream web site, entitled "i-scream PHP web reports".

All chart files have a file name of "i-chart.gif".

Meta data

day.inc

This is a file containing PHP mark-up that specifies the day of the last report generated. This is used by the web interface provided by the i-scream team. This is located in the root of the output directory.

report.inc

This file is located in the [date] directory and is used by the i-scream web interface. It contains an option list of all reports produced for the particular date.

machines.inc

This file is also located in the [date] directory. It is also used by the i-scream web interface. It contains an option list of all machines for which reports are available.

i-data.txt

This is a file containing the data used to produce the chart. It is produced for each report and located in [date]/[machine name]/[report name]. This contains pairs of values, one pair per line, and may be used by future i-scream (or possibly third party) tools to produce reports of large periods of time without the requirement of long term database storage.

i-maxmin.txt

This file is written to [date]/[machine name]/[report name] and contains PHP mark-up to specify the maximum and minimum values for the report attribute during the report period.