MyDAQ, A SIMPLE DATA LOGGING AND DISPLAY SERVER

Akihoro Yamashita *, Toru Ohata
SPring-8, Mikazuki, Hyogo, 679-5198 Japan

Abstract

MyDAQ is a data logging and display system suitable for periodical (~10Hz) data such as DC voltage, temperature and vacuum pressure. MyDAQ system receives data strings from data taking PC's on the network, stores them in a relational database and displays them using web. Client user adds only a few lines of code to their data acquisition program written in C, LabView or their favorite script languages. And they can see data on a browser specifying name of data and time of acquisition. MyDAQ consists of a multi-threaded socket server, MySQL relational database server and cgi programs. It runs on Linux or Windows operating system.In this presentation, its structure, performance and scalability will be discussed.

INTRODUCTION

MADOCA [1] has successfully controlled SPring-8 accelerator complex since its commissioning at 1997. It is widely accepted not only controlling accelerator, but also controlling beamline instruments.

Web access to the log data is one of reasons of MADOCA's success. MADOCA at SPring-8 logs over 50,000 points of data acquired from accelerators and beamline forever on a relational database system. MADOCA has interfaces to view or download data via web. Everybody on the campus can plot data stored in the database on their web browsers specifying signal name and time period.

MADOCA's human readable name scheme make user to easy access to data. For example, One can easily understand name such as sr_mag_ps_q_12_3/current_adc.

Programs in MADOCA communicates each other passing text strings as messages. The program packs command and data into one message and send a message to the other programs running same or other computers. Program which receives a message interpret it and do something according to the command and data. Message oriented architecture saves number of API's. Only *send* and *receive* commands carry out the job.

Users who enjoyed MADOCA requested *mini-MADOCA* system for their rather small experiments at SPring-8 beamline. They wished to store their data from their data acquisition PC's and view them from web browsers. Data acquisition PC's are mostly Windows machines and data acquisition programs are written in various programming languages which are not supported by MADOCA. Number of data points never excesses 100 in most case at frequency less than 10Hz. The *mini-MADOCA* need to have quick and easy installation feature for beamline experiments. Because beamline users frequently install and remove experiment equipments.

We developed a new data logging and viewing system called MyDAQ satisfying above requests. MyDAQ inherits human readable naming scheme, message oriented architecture, logging database on RDBMS and web data browsing from MADOCA. In addition to those features, MyDAQ aims easy installation, quick data registration, access from programs written in multi programming languages running on multi platforms.

STRUCTURE

The structure of MyDAQ is shown in Fig. 1. A data acquisition computer attached to experimental devices packs command and data into a message. A message is send to a server via a socket connection. Receiving a message, the server interpret it to generate a SQL command and send the SQL command to the database server. The database server store data according to SQL commands. The web server accepts request from clients, retrieve data from the database server and display them.

MyDAQ's communication never take account of network security. We assume the data taking network is protected by firewalls from internet outside.

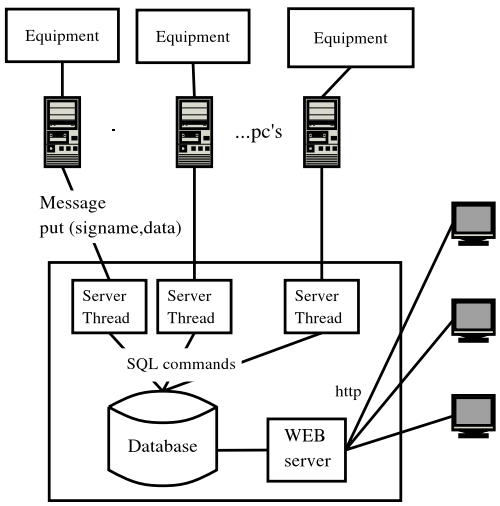

Figure 1: Structure of MyDAQ.

Message

A message is a text string which contains command, signal name and data separated by slash "/" character. For example, put/mag_ps_current/0.1 is a message ask to put value 0.1 of signal name mag_ps_current.

* aki@spring8.or.jp

Three kind of messages are used for communication. *Create* command register signal name to the database. *Put* command enter data into database. And *Quit* make disconnect the communication.

A list can contain mixed data types separated by underscore character like *0.1_32654_1e-20_2225*. To define list data, the user specify it with crate message as *create/signal_name/fifi*. *Fifi* in this example means this signal contains a list of float,integer,float and integer. The return message from the server is either *OK* or *FAIL*.

Client

Users who wish to use MyDAQ insert a few lines of codes to their data acquisition programs, which already exist. The code packs command and data into one message and send it via socket.

Every platform and programming language which support socket can be a MyDAQ client. We have developed client programs in C,Python, Ruby, Visual Basic and LabView on Linux, MacOS X and Windows platform.

The client and server communicates synchronously. Client waits for return message from server after client message is issued.

Server

A server receives messages from clients, interprets them, generate SQL commands, send them to the database server and return result messages to the client. The server program written in Python [2], an object-oriented scripting language, is implemented on Unix-like operating system. The multi-thread server library built in Python default distribution made easy to construct server. When a new connection request comes from a client, one new thread is created which has one connection to the database server. We confirmed up to 500 threads created in a server.

We have ported the server program to FreeBSD and MacOS X from Linux platform on which initial version is build. Although, we have not ported the server to Windows operation system yet, Python's OS independent feature will help to port to Windows OS. The MySQLdb [3] module of Python connects the server to the MySQL database management system (DBMS) . Because the module respects standard Python Database API Specification v2.0 [4], one can replace database management system from MySQL to other DBMS with minimum efforts

Database management system

We employed MySQL [5] as database management system. MySQL is a open source relational DBMS. Every major Linux distributions include MySQL as default packages, so user can begin to use MySQL server with no effort in most case. If the system has no MySQL, a user can obtain binary packages from the vendor's site and even compiling from the source is straightforward. MySQL has reputation of fast DBMS. But earlier version had limited functionality. For example, before version 4.1, join was not supported. We designed MyDAQ to work earlier version 4.0 which has no join nor sub query. MyDAQ has been confirmed to work with MySQL v4.0 and 4.1.

Internal structure of database

We construct database under *one signal one table* policy. A table is named as identical as signal name. Therefore, the naming rule of MySQL table regulates the signal naming. Name of a table of MySQL is allowed up to 64 alphanumeric characters plus underscore and dollar sign. One table have columns of signal list, time and microsecond time. Client program packs second and microsecond of data taking time with data into a message and send it to the server. Time is express as unix epoch time in integer. A table has an index with time key for fast data seeking. Signals are grouped for easy searching signal in web page.

Web Server

The web server receives requests, retrieves data from the database and displays them. A user requests data plot specifying data taking period and signal name. Data are plotted or downloaded from the browsers. User can import downloaded data to their analysis program.

Also the web server provides data management. User create groups and define signal and group relation at their web browser Cgi-programs written in Python serve above requests. Gnuplot [6] called from cgi-programs draws data in png format with size of about 5k bytes. MADOCA has rely gnuplot for years and no other graph package draw time plot as well as gnuplot.

Although MyDAQ runs on Apache [7] in most case, MyDAQ runs on other web server because MyDAQ does not depend on Apache's specific function.

INSTALLATION

Installation is straightforward. MyDAQ uses MySQL, Python, gnuplot and web server. Those packages are standard in most Linux distribution. Even if those are not in the system, user can install them effortlessly with pre-complied packages such as RPM.

The system with those packages, installation is just done with copying server program and cgi-programs and running several sql programs to set up the database. No compilation is needed to install MyDAQ server, because all the server program written in a scripting language. Typical user need no more than half hour to install MyDAQ server.

PERFORMANCE

We measured performance of MyDAQ in two aspects. One is performance at multi signals. And another is whether performance deteriorate after the long data acquisition. We measured time between send request and receive reply to determine the performance.

MySQL v.4.1 which was downloaded from vendor's site on high-end and low-end platform were used. Specifications of high-end and low end servers are shown in Table 1. MySQL was running with no special tuning except max number of connection parameter.

Table 1: Server specifications

	High-end	**Low-end**
Linux kernel	2.6.8 64bit	2.6.9 32bit
CPU	AMD Athlon	VIA Eden C3
Clock	2400MHz	667MHz
Memory	1GB	128MB

Performance at multi connections

We measured performance of MyDAQ with 50 to 500 simultaneous clients. Clients send request server to insert one integer at 1Hz. The results are shown in Table 2. Comparison between high-end and low end server with 50 client connection is shown in Table 3. MyDAQ serves requests at satisfactory high speed both at high-end and low end server. Even at the 500 connection, no events take 0.3sec at high end server. The low end server also performs well.

Table 2: Performance at multi connecton environment. Values with "<" means upper limit at 95% confidence level.

	100	**250**	**500**
Mean(ms)	3.35	3.46	3.32
σ(ms)	1.98	2.13	2.39
Rate (%)			
>5ms	9.1	10.78	9.11
>10ms	.67	1.91	1.19
>100ms	$<9.8\times10^{-5}$	1.4×10^{-3}	6.7×10^{-3}
>300ms	0	$< 2.4\times10^{-5}$	$<1.5\times10^{-5}$

Table 3: Performance Comparison of High-end and Low-end Server

	High-end	**Low-end**
Mean(ms)	3.88	3.24
σ(ms)	2.04	7.05
Rate (%)		
>5ms	22.2	6.38
>10ms	1.6	0.24
>100ms	$<3.8\times10^{-5}$	0.2
>300ms	0	5.7×10^{-3}

Performance after long data taking

We measured insertion time to large table. MyDAQ inserts data into table under *one signal one table*. After long time of inserts, table size becomes larger as well as index size. Large table and index may worse the performance of insertion. We create a table which simulates two years of 1Hz data taking. Table 4 shows comparison of large and fresh table at high-end server. No significant deterioration was observed after two years of data taking.

Table 4: Performance of small and large table

item	**Small**	**Large**
Mean(ms)	0.71	0.78
σ(ms)	9.0×10^{-3}	0.8
Rate (%)		
>5ms	1.0×10^{-3}	1.0×10^{-3}
>10ms	1.0×10^{-3}	0
>100ms	0	0

EXAMPLES

Recently MyDAQ was adopted by two user groups. One is the group which is testing the injector linac of SCSS[8]. During the aging of the RF gun, many measurement was required. Over twenty signals of dark current, vacuum pressure and temperature to RF power are monitored at 5 Hz. MyDAQ was installed in the Linux PC of the measurement system. Client program written in C and server programs are running in the same Linux machine.

The other group uses MyDAQ as a parameter database to compensate the monochromator at the SPring-8 beamline. The correction curves are stored in the vicinity of every measurement to check alignment of the monochromator. MyDAQ server is running on FreeBSD and LabView client program is running another Windows PC.

CONCLUSION AND FUTURE PLAN

We developed MyDAQ which acquires data from multi platform PC, stores and displays them. MyDAQ succeed to MADOCA's message passing, use of relational database and display at web browser. MyDAQ's multi-platform and multi-programming language feature is well fit to real experimental environment.

MyDAQ is sufficiently fast even running on low-end server. Script language used in server showed good performance with multi-thread function.

At the future, we plan to add API with which user programs retrieve data from the database. Programs on multi-platform and multi-language have to access the data in an unified method as MyDAQ. XML-RPC is a candidate satisfying those requirements in simple way with http protocol.

REFERENCES

[1] R. Tanaka et.al., “Control System of the SPring-8 Storage Ring”, ICALEPCS 95, Oct. 1995 , Chicago. p. 201.
[2] http://www.python.org/.
[3] http://sourceforge.net/projects/mysql-python/.
[4] http://www.python.org/peps/pep-0249.html.
[5] http://www.mysql.com/.
[6] http://www.gnuplot.info/.
[7] http://www.apache.org/.
[8] T. Shintake, “SPring-8 Compact SASE Source” , SPIE2001, San Diego, USA, June 2001.