Automatic Data Logging

MANAGEMENT SUMMARY

Explains do's and don'ts of automatic data logging

TABLE OF CONTENTS

1 Purpose

This document provides some pointers on automatic data logging. It explains the benefits and gives sample on what and how to log information mainly for data analysis & presentations

2 Benefits of automatic data logging

- Costs of execution are low. See § 4 Cost of execution.
- A global setup can be reused for every new project/department with limited cost. See § 4 Cost of execution.
  Project dependency can be achieved with a simple project parameter.
- Data can be made available via
  - A "Project Metrics" portal
  - Automatic imports is e.g. a project progress report (and finally in a project evaluation report)
- Consistent use of data, formulas and presentations in an organization
- Possible indicators for comparing performance of projects and departments..
  By comparing e.g. a project you compare project results and indirectly compare the quality of the data logging.

3 Prerequisites to do automatic data logging

- A script language must be available
  Php, Perl, Bourne or C shell, dos command line, etc.
  Executables are harder to develop and more difficult to maintain for non developers.
- Tool for automatic scheduling of script execution must be available e.g. Unix crontab, windows schedulers etc.
- The used development/registration tools like CM tool, Change tool, Requirements tool, etc. must support a command line interface for queries and reporting.
  Or:
  The data model the tool uses for storage must be known and the (Relational) Database Management System used by the tool permits command line retrievals.
  This way of working is not preferred because.
  - Data model might not be the same after a tool update.
  - The tool itself may do night/batch processing and:
    - Might have data locks in place that prevent data access. This causes temporary data inconsistencies during processing.
    - A lot of work to be done to combine relational information into a human readable reports.
- A (data) server that is 'always' online and active
  Needed for:
  - periodic script execution (evenings, weekend)

  - data storage
- Crontab/schedule user must have rights to access the tools.

4 Cost of execution

4.1 Initially effort to setup

- Initial cost to setup the overall concept like server, scheduler, etc.
- Initial costs to setup a data logging for a specific purpose/graph/etc..
- Initial setup of a data presentation.

4.2 Periodically costs to verify if the logging was correctly executed.

- Is data logged
  - Log error/missing detection can be part of the automated process!
    - Every time period a script runs to see if timestamp of data files has changed in the last period.
    - Log the start of a log script
    - Send email as last action in a logging script.
- Is the data still valid? See next bullet.

4.3 Effort to verify if the content is still meaningful

We need to use the data to detect this!
Make sure:

- There is a presentation available that for each data log file.
- That the data is used in structural reporting mechanisms like:
  - progress reporting
  - weekly data performance measurements

4.4 Negative effects due to processing power used for data logging

- User delays due to data logging performance loss
- Real time effects due to time spend on data logging
- Additional code to add to support data logging
- Disk space consumed/reserved for data logging

4.5 Maintenance of the logging concept

- Costs due to IT infrastructure changes
- Cost due to organizational changes
- Cost due to tool (data source) version updates
- Cost due to reporting tool changes

5 Frequency of automatic data logging

The frequency for adding data to a data log file depends on the purpose of the data (and thus the log file.)

1. Every workday night (incl. Saturday)
2. Every weekend (On Sunday Evening)
3. Every build, cleanup, ?
4. Every Master Build (Master is to be defined:-)
5. Every (planned) integration Step
6. Every baseline made (incl. project Milestone baselines)

For weekend nights you have to choose an early night time like .e.g 00:30. You have to take take into account summer time switching from 02:00 => 03:00 =>02:00 twice a year.

6 Output file types used for data logging

6.1 Incremental log files

Used to show historical trends. At regular intervals a new data line is added to the end of the incremental file.
Notes:

- New data fields can be added at the end of the line.
- Existing data field can not be removed
- A line should start with a fixed data format
  E.g: YYYY-MM-DD HHH-MM:SS

Samples:
Historical evolution of CRs in a project on a weekly basis.

6.2 Snapshot log files

Used to show information on a specific subject at a specific date, marker, or milestone.
Also used to show complex information structures (matrix) that can't be logged in a record format

Every time data is logged a new data file is created
file name suggestion: <purpose>_YYYY-MM-DD.<YYY> YYY=txt, log, ??

Samples:

- Number of different types of source files at a given date
- Distribution of error causes for PR's at a certain project phase
  Metric of PRs Inserted-in-phase versus detect-in-phase

6.3 Current log files

Used to show the latest/active content
One file is used and the file is renewed at certain point in time daily, Weekly, etc.
file name suggestion: <purpose>_latest.<YYY> YYY=txt, log, CSV, ??

Samples:

- Last time a built succeeded and the time it took
- Current summary of the CR/PR database.
  For those who don't have access!
  Or if the tools doesn't support such reporting features

6.4 Current log files Updated

A special case where the information of the old file is read and the data is processed with the new results and logged-again in the same file.

This is an error prone solution!
Better is to use incremental files and count the results when/after all lines are processed.

Samples:

- No samples known.

6.5 Data separation inside a data file

Based on the information logged in a file the 'correct' data field separator needs to be chosen. E.g. a , (comma) will not work with free text logging. See example data record/line
20080509,01:23:34,1,3,27,35,14,21,56

7 Samples of automated data logging

Samples of logging files in a directory structure

- <X>:/metrics/organisation/OFT-usage.log
- <X>:/metrics/projects/<projectid-N>/PRCR/PRCR-evolution.log
- <X>:/metrics/projects/<projectid-N>/PRCR/PRCR-Discovered-in_YYYYMMDD.log
- <X>:/metrics/projects/<projectid-N>/PRCR/PRCR-snapshot_YYYYMMDD.log
- <X>:/metrics/projects/<projectid-N>/PRCR/PRCR-snapshot_latest.log
- <X>:/metrics/projects/<projectid-N>/built/Intergrator-built.log

8 Visualization of the logging

Logged data can be visualized via a dedicated or common office tools
Some examples can be:

- An excel sheet macro reads: the /PRCR-evolution log file and created a PRCR history graph.

<show sample picture>

- The same trick can be done when creating a project progress report.
  <show sample picture>
- A intranet web server reads log data and converts this into tables or graphs

Simple scripting languages like Perl, Php, Bourne Shell, etc. (maybe with third party libraries) have sufficient functionality to do this.

9 Partial data logging on project request

When this logging concept is in place any project can benefit of the full analysis potential almost without extra overhead or costs.

Some projects might decide not to use all data because they see no need to do so.

My suggestion is to initiate a full data collection for every project even if they won't use (all) the data. This is done to preserve historic information in case they decide otherwise and to support project comparison by the organization on all projects for every data item.

It also will cost more initial setup cost (and maintenance costs) to tweak the default setting to exclude a part of the logging!

If the don't uses the tool/registration that contains the data to be retrieved then nothing can be done. Just accept then an empty log file and/or graph
Personal note: An organisation should not permit such a project behavior!

10 Example Metrics for automated data logging

What can be tracked via automatic logging?

- Product Creation (Build) related:
  These measurements are only possible if a makefile is used.
  - Build time:
    Time it takes for a built to be executed
    - Log start time at the start of a make

- Log end time at the end of a make
  No end time then the build has failed

  Suggestion to do this only for the integrator/librarian
- Options used to generate a product
  - used if different products can be generated for a single archive
  - Used if different debugging options can be used.
- Number of files in the scope of the built
- files are common for every built or for a product cluster
  - Which files are unique for a product
- Suggestion to do this only for the integrator/librarian

  Relation between created and generated file built execution

- Configuration Management tool Related
  - Number of code labels
    - script can check label format and prevent double usage
    - Script can log info on label
  - Information on files in an archive
    - Number of files
    - size (bytes/lines) per file
    - New or possible reuse or not
      See White Paper: CM reuse measurement
    - number of checked-out- and checked-in files

- Coding rules check on an archive

  Run a static code analyzer on all files in a archive
  - calculate (average) compliance to rules per file / module / archive
  - Which rule is top contester

  Maybe you need to do some active log processing via tools like AWK en GREP (or similar).

- CR/RP tool related
  - Historic distribution of all (extra) attributes
    (Found-in, caused-in, Cause, priority, urgency, severity, analysis-time, solve-time, total lead-time, analysis or other phase time, number of status transitions, average effort on PR/CR.
  - Maturity grid
    If you have an attributes Severity and you can translate the current state into a evolution value and it is possible to report both a matrix which show the number of PC/CRs in a severity/evolution cell.

- Total Cr/Pr capacity pending
  Based on the CR/PR characteristics (type, Severity, scope,?) and current state one can calculate the total amount of effort still needed to solve a Cr/Pr based on average value estimates. Summarised on all 'open' Cr/Prs is the additional project capacity needed.

- Requirements management tool related
  - Number of requirements
    Can be on hierarchical levels if used.
  - Distribution of the number of relations/links requirements have (Indication of product complexity)
  - Number of changes on a requirement
    (Indication on requirement complexity)
  - Number or (grand+) children of a parent requirement
    (Indication on requirement complexity)
  - Distribution of requirements per requirement attributes like e.g. origin, stakeholder, discipline, priority, etc.
  - Number of reused requirements
    See White Paper: CM reuse measurement.

- The "One File Template" tool (for template generation)
  The tool can log which template is created.
  (All filled-in attributes can be taken into account and logged also!)

  Note: A "One File Template" is mostly a single word file that holds all word templates of which the user has to select the template of choice using a pick list. An OFT solution has many advantages see White Paper on OFT.

*

ImQuSo Improved Quality Solutions.
Po Box 169
5540 AD REUSEL, The Netherlands

Doc. ImQuSo-WP-2008-002 Rev. 0.1 2009-04-15