

Business Case of the Project: APSP

Yaroslav Halchenko¹, Christopher Rebstock², Kareemullah Quadir³,
Roshan Rammohan⁴, Vinod Doshi⁵

February 16, 2002

¹yoh@cs.unm.edu

²chrisr@cs.unm.edu

³quadir@cs.unm.edu

⁴roshan@cs.unm.edu

⁵vdoshi@cs.unm.edu

Overview

The Personal Software Process is a software development process that allows the individual to apply industrial level discipline to his or her practice.

Our objective is to develop a APSP (Automated Personal Software Process) tool, which would be used together with CVS to automate the PSP process to a large extent. Tasks like monitoring and reporting (log, statistics, progress) for each individual software developer would require no manual intervention.

Purpose

To assess personal achievements and progress it is useful to have a tool which would show trends of stages where developer spends most of the time (developing, debugging, learning new technology and tools etc), provide statistics on how code-development process going on (how many lines were added/removed/modified).

Most of PSP tools existing on the market involve filling some forms which would be analyzed thereafter and necessary statistics and reports will be drawn. This process can be time consuming and could demotivate people from using PSP and may not report some aspects of Software development.

Vision

One of the main feature of this product is that monitoring will be done automatically and manual keying in of data is reduced to a bare minimum. Statistics will be accumulated by monitoring changes of files in the project by some tool running in background (by crontab for instance), also some information about CVS activity will be conducted through CVS wrappers (keeping track of updates and commits and how much information in them). Thus developer can get a lot of information where/how he/she spent most of his work time.

The APSP tool would consist of a client which would communicate with a central APSP server which would enable a developer to implement PSP irrespective of his location. The Server would automatically integrate the data.

Special Features

- Automatic and constant file monitoring.
- Client/Server Architecture would enable a developer 'roaming access' to his PSP.

- The 'Thin Client' would be platform independent.

Server

Server Responsibilities:

- Collect information from all programmers/computers.
- Provide web-backend where each individual can check report on his activity online in the inter/intra net. Specific information would be
 - Projects he was working on.
 - How much time a programmer spends on specific phases in a project.
 - * Research
 - * Planning
 - * Design
 - * Coding
 - * Debugging
 - * Testing
 - * Postmortem
 - Statistics on specific files he was working on.
 - * LOC added/removed/modified
 - * Number of transactions (commit,update,checkout) with CVS done and amount of information and time lies between them.

Client

The Client will collect information and send it to the Central APSP server. Kinds of clients:

- Monitoring-tool which would run all the time to monitor user's activity on files in specific directory.
- Set of CVS wrappers to report activity on the project in CVS
- Support tools to grab other kinds of statistics (register number of compilations and debugging time etc).
- Tool which would report change in developers activity. (developing, debugging etc)

OS and Hardware Requirements

The system will be developed and tested on the Debian 2.4 Linux distribution and hardware capable of running this would suffice.

Tool depends on the presence of CVS client on current machine and assumes that projects which the developer works on will be kept within that CVS. More specific role of CVS in the project is not just a version control system, but to also provide us with the structure of the files/directories involved in specific project, so separate human-driven interface to specify files which should be monitored becomes unnecessary.

Development and delivery platforms

- Database: MySQL
- Web Server: Apache
- Development tools: Shell Programming, Java(Client Server), Javascript(web client-side scripting), HTML

(Implementation) to make a wrapper for “cvs checkout” command which will mark that current directory is root directory of some project. So monitoring program can get that directory name and get list of files and subdirectories from CVS/Entries.

Risks

1. Significant portion of the group is not well versed in shell programming, databases and CVS.
2. We're still finding out what PSP is all about.
3. The idea of the authentication procedure is hazy.
4. Integrating various technologies/platforms/software that we plan to develop this tool might prove tricky.
5. Analysis of file modification under development (how many lines were added/deleted/EDITED) can be a tricky algorithm.
6. Defining a protocol for communication between clients and server can be hard (needs to provide caching mechanism for cases when client is off network).
7. Might not complete the project in time.

References