Preprocessing CVS Data for Fine-Grained Analysis
By Thomas Zimmermann and Peter WeiBgerber

Summary:

A CVS repository contains information directly related to the evolution and progress of a software project, specifically who has changed what and why. The information by itself however is not correlated with regard to comprehending the software development process. In order to better understand the mechanism, the CVS information needs to be processed to create the mappings required for comprehension. By pre-processing the information stored in CVS, these mappings can be created and heuristics applied to massage the information into a more usable form. The CVS data itself can be propagated to a database by parsing the CVS log. Information regarding revisions can be stored as an entity, transactions regarding the author and the log message are stored in another, and in addition Files, Directories, Tags(User assigned symbolic tags for revisions) and Branches encompassing Branch points and names are also assigned to entities. One of the key scopes of this process is to determine an association between the files that have been modified. To do this two approaches can be used, either a fixed window reference or a sliding window reference. The fixed window reference will group all transactions within a predefine window as being related, and hence imply an association between the files. The sliding window approach allows for a more dynamic approach by instead defining a maximum time interval between transactions, and all transactions within that interval are considered related. The key consideration that both approaches attempt to deal with is that commits are not instantaneous, but instead happen over a period of time as a developer checks in the modified or updated code. Sliding windows however provide the more flexible approach as they can deal with transactions of any size. Both analysis indicate candidates for further examination, which can then undergo a more fine grained examination by examining the actually files and directories to determine which files have undergone changes. Other issues that need to be considered are infrastructure changes to the software modules themselves which produce large transactions as well as the merging of branches, which simply reproduce prior changes to the software projects. Both of these scenarios would create “noise” in the statistics being gathered and extracted leading to misleading correlations in the analysis. To cope with these large transactions can be avoided by setting a maximum value on the sliding window being used, and segments larger than the value are ignored. Detecting merger transactions is more difficult but can be accomplished through some automated processes but still requires examination by person(s) for verification.

CVS repositories can provide insight into the development process of software by examination of the log entry information as well as the changes to the files and directories involved. However the information involved needs to be pre-processed to create the correlations between the modules based on the information stored in the commit stage, and then files and directories involved can be examined in further detail. Though not a perfect process, this approach provides for some automation in using CVS for software analysis.