A study of Implementing a UNIX command database in PCTE

Satoshi Tanaka

School of Information Science,
Japan Advanced Institute of Science and Technology

February 15, 2000

Keywords: UNIX command, PCTE, database, semantic constraint.

1 Background

With the spread of computers, the scale of software has increased explosively. In such large scale software, code fragments have complex constraint relationships each other. Therefore to develop and maintain large scale software efficiently, it is clear that we need software databases, which software objects appearing in the software life cycles are stored, managed, and accessed. While the concepts of software databases is not new, practical software databases have not been developed yet. This fact shows software databases are difficult to develop.

2 Purpose

The goal of this study is to clarify what complicates software databases, and to establish how to develop software databases. Previous research has been focused on informal software objects like memos, informal specifications, complex diagrams, and so on. In this study, we focus on formal software objects, and take an approach to implement small software databases which preserve the essential difficulties of software databases. We experimentally implement a UNIX command database in Emeraude PCTE, which gives us information on what are the semantic constraints when we regard UNIX commands as software objects, and Emeraude PCTE as a tool to build software databases. The steps of building the software database are as follows:

1. We select several UNIX commands as the material of building software database.

2. We extract all relations in the UNIX commands, and describe them as ERA models.

3. We extract all semantic constraints in the UNIX commands, and realize them as programs that run on Emeraude PCTE

The purpose of this paper is to categorize complications of software databases from experience of implementing the UNIX command database.

3 Properties of UNIX commands

- There are many complex relations among UNIX commands.
- But there is no way to tell us information about complex relations among UNIX commands.

A program called "shell" is a UNIX command interpreter. Using shell, we combine several UNIX commands using pipe or other shell's construct. Furthermore, UNIX commands often invoke another command via environment variables (consider, for example, "man" command's PAGER). So there are many complex relations among UNIX commands. This means it is not easy to handle complex relations among UNIX commands, although they are represented as formal software objects.

On the other hand, UNIX commands are familiar enough for us to understand. This means it is relatively easier to extract relations and constraints among UNIX commands than to do so among real software objects. So we select UNIX commands as handy-sized software objects.

4 Emeraude PCTE

PCTE, Portable Common Tool Environment, is a standard of portable tool interface for open repositories. In this study, we use Emeraude PCTE, which is one of software realizing PCTE. The Emeraude PCTE repository is called the object base; information is represented as objects connected to each other by links. In PCTE, we use SDS (Schema Definition Set), which adds specific properties to ERA models such as ....(ここかく). So it is easy for us to describe relations and retrieve objects traci ng relations. SDS supports some semantic constraints (for example, cardinality). And we can program semantic constraints not supported by SDS using C language.

So we select Emeraude PCTE as a tool of experiment.

5 Describing relations

To describe UNIX command relations, we use three kinds of ERA diagrams as follows.

- meta-class diagrams

- class diagrams
- instance diagrams

We use class diagrams when we need static information, for example, a relation between command and file as command's parameter. Instance diagrams represent dynamic relations among UNIX commands, for example, a concrete file name or parameter value. We use meta-class diagrams as a meta-schema. We need meta-schema, since we need data integration to search all UNIX commands for some common properties.

6 Implementing semantic constraint

We can program semantic constraints, not supported by SDS, using C language. But it is difficult to implement all semantic constraints as program completely (such constraints are implemented partially). For example, UNIX "man" command is related with other commands via the environment variable PAGER. We expect the command specified by PAGER shows us the content of text file comfortably . But in program, it is difficult to describe such intention. So we try to implement such intention as a program partially. For example, retrieving a list of the appropriate commands as the values of PAGER. But our experimental implementation does not support new commands.

7 Classification Difficulties

It holds some problem of a software database, for example dynamic information, and relations of UNIX file system is comparatively simple.

We classify the difficulties of semantic constraints by building the UNIX command database. The resulting classification and the corresponding partial solution are as follows.

- semantic constraints about location
  - describing information about location
  - searching location by "find" command
  - consulting other users' file ".cshrc" as hints
- semantic constraints about intention
  - retrieving a list pre-defined as appropriate values
  - adding the attribute i.e. typing
  - searching all UNIX commands and retrieving relations
- semantic constraints about dynamic information
  - retrieve instance diagrams assuming that instance diagrams are available

We also discuss this classification can be observed in software development.

8 Conclusion and Future Works

The conclusion of this study is as follows.

- It is often difficult to deal with relations and semantic constraints among formal objects as well as informal ones.
- We need both of formal description and informal one. There are semantic constraints, which cannot be described without informal description. But there is a problem how to deal with informal description on a software database.
- Semantic constraints can be classified by their properties. We classified semantic constraints, which are difficult to program, — location, intention, and dynamic information —. We discussed this classification is not limited by a UNIX command database.

Our contribution for software databases is the above mentioned classification of the difficulties.

Future works of this study is as follows.

- establishing meta-class diagrams and classification of semantic constraints

  We must deal with much more UNIX commands.

- considering usefulness of classification

  We must consider whether classification of problems of software database is meaningful or not.

- implementing an instance diagram generator

  We need a tool generating (semi-)automatically instance diagrams, because dynamic information in instance diagrams are very important. But full-automatic one is not realistic, because it requires all information included in the original source code.