A dream of an ultimate OS

Oleg Kiselyov
CIS, Inc. & University of North Texas
303 N. Carroll, Suite 108
Denton Texas 76201

mailto: oleg@ponder.csci.unt.edu or oleg@unt.edu
http://replicant.csci.unt.edu/~oleg/ftp/
ftp://replicant.csci.unt.edu/pub/oleg/

THE POINT

Database Editing

- Everything a user ever does at a terminal is just editing:
  that is, requesting/reading/modifying textual information, arranged mainly into tables/scrollable lists

- Everything the computer is in its guts and appearance is a collection of data with procedures how to get hold of them/change them: database

It is glaringly obvious. It is this way now. So what am I griping about?

GRIPES

about OSes

- A whole **mess** of disparate commands/ actions/interfaces, all doing essentially the same thing:

  filling in and editing some list/table

  notable exceptions: MacOS, MagicCap and Newton's OS

- A whole **mess** of custom-kludged databases/record managers, all doing essentially the same thing:

  – inserting a record into a "table" with hashing of a key

  – retrieving a record/field using a simple/ concatenated key

  – linking tables

  many of database managers aren't even good at it

UNIFICATION IS THE WORD

If it looks like a bag, if it holds like a bag, it should act just like ***a*** bag

bag = {folder, process table, list of connections, set of lines of text}

- Trade a multitude of "custom" database managers for one well-designed distributed database manager?

- Adabas-like or DB4-like or Oracle-like database **<u>instead</u>** of a file system?

- Looking for a word FOO in <u>something</u> and then deleting/closing/stopping this something

  be it a paragraph of text, a network connection, a subscribed newsgroup, a process

  – would require roughly the same sequence of mouse clicks/keystrokes

  – would be understood and interpreted in the same spirit by the operating system

- Side-effect: ease of linking

  in the sense of database record linking

of almost anything to anything

The desktop is probably going to look just like a ***homepage***

EVERYTHING IS EDITING

Removing a row from some table (list)

- Deleting a line of a text file:
  \C-Y, PF4, Delete, drag out (sometimes)
- Removing a file:
  rm file_name, drag out
- Killing a process:
  kill process_id, kill %job_id, select-click-on-kill
- Stopping a print job:
  lprm job_id, cancel-in-print-manager, **no dragout**
- Removing a route
  route delete default 129.120.3.250 1

The *conceptual* uniformity runs very deep

- on the level of implementation
- on the level of presentation

Unfortunately, interfaces <u>evolved</u> to a tangled mess

A FEW PEARLS
not too ripe, though

• /proc is cool
  a process is a file, too
    > ls -l /proc/00108
    -rw------- 1 root 2478080 Jun 10 12:35 /proc/00108
  Well, *not exactly*
    rm /proc/00108
    ls /proc/00108/open_files

• Drag&Drop
  files, lines of text, data in CW projects

• OpenDoc

• **Still**,
the Finder could manage folders of *files*
  which are not files
    newsgroups, FTP sites, processes, printers, print documents, **paragraphs of text**
and let use standard Finder operations
    View-by, GetInfo, Trash, Duplicate, Open, Sharing, Find, Label
on those *files* and *folders*

Nuntius, Anarchie,... have to simulate Finder

• Hypertext for free
  a document can contain an application or a folder
  The entire OS is the OpenDoc

VERMIN of PLAIN TEXT

- Huge tangle of plain text system files

  /etc/hosts, sendmail.cf,
  /etc/syslog.conf, /etc/inetd.conf,
  /etc/uucp/Systems
  .INI files....
  Even MacOS caved in: :System Folder:Hosts

- Why they proliferate:
  even a naked system can ***cat*** and ***edlin*** them

- Need low-level tools to conveniently access databases

  ResEdit + templates aren't bundled with the MacOS, but the SimpleText is

EVERYTHING IS DATABASE

- It is indeed:
  - process table
  - page tables
  - i/o request queue
  - list of open windows
  - resources.
  - SCCS, RCS
  - YP maps
  - termcap/printcap
  - domain name service, name resolver, /etc/hosts, /etc/networks, /etc/services, routing tables
  - whatis database for man pages
  - whereis database/cache for find
  - crontab

- All support basic select/insert/delete concatenated keys and linking among several tables are often required, too

- Often a microsec-tight performance is not an issue

  custom-made very tuned-in implementation needed only occasionally

- Many of these databases aren't even good: flat-file with only linear search capabilities

UNIVERSAL DB MANAGER
in the system core

- well-implemented storing/caching/search/ modification capabilities for everyone to use

- linking records in a uniform, consistent way
  - users, processes, print jobs, ...
  - forget uid, gid
  - replace #include <foo.h> with a link to foo.h
  - get Dylan-like development environment for other languages (C++, Pascal) and text documents

- weirdly-wide searchability
  better and faster replacement for scripts
  - ps aux | grep joe
  - netstat -a | grep finger
  - find /var/preserve/ \! -name /var/preserve/ -mtime +7 -exec rm -rf {} \;
  - /usr/etc/lpc status all | grep entr | grep -v 'no entries'
  - ed - /etc/passwd <<-EOF

```
    H
    /^$user:/s/^\([^:][^:]*\):\([^:][^:]*\):\(.*\)/\1:*\2:\3/
    w
    q
   -EOF
```

They are all database queries

NO FILES
only records

- Many FS are hierarchical hacked to network databases

- Good database **instead** of the file system

DB has everything a file system has and does:

  - storage management (++)
  - search (+)
  - caching (+)
  - maintaining consistency (+++)
  - locking (+)
  - access restriction (+)
  - optimized distributed access (++)

and it usually does everything above on its own

- FileMaker databases are used to catalog the contents of CDs and floppies
  FileMaker ≡ FileSystem?!

- Not everything between slashes (colons) is a directory name: it may be just a key
  http://somehost/foo/bar.html.
  http://mozart.cisi.com/map/sowlp_comp?parm=1100

WHAT IS A FOLDER THEN?

- *Folders* are saved database queries (**database views**)
  - a file (data item) can appear in as many *directories* (views) as one wishes to
  - any file modification would be visible in all views
  - searching a file system, and creating and populating a folder becomes the same activity
  - views can be stored in the database and referenced to from other views
  - there is no required hierarchy

- Database record

owner	timestamp	perm	kind	comment	blob

owner	timestamp	perm	*image*	comment	

	format	width	height	depth	compression	

```
FSMakeFSSpec(0,0,
"MyVolume:neat photos:type=image:
depth=8:width<=512", &dir_record)
```

- Rudimentary database browser
  - even a naked OS should have it
  - like a basic no-frills record browser in Paradox
    - displaying all record fields in columns
    - value=name pairs

running <u>find</u> and listing a *directory* would be exactly the same activity
Opening a file ≡ SQL query

Words of the talk

(comments on transparencies)

A DREAM OF AN ULTIMATE OS

This talk is a cut-and-paste collage I've been fooling with for many years. The title I hope would excuse slight misalignment of edges, I mean, thoughts. I wish I could show you a lot of pictures, but I have to apologize for the complete lack of artistic talent or telepathic ability. So there're going to be all words.

I move to the point right upfront. And it is two words: Database Editing. For one thing, Everything the user does at the terminal is just editing, Indeed, it is glaringly obvious that the only thing a user does at the terminal is requesting/reading/modifying textual information, arranged mainly into tables/scrollable lists.

GRIPES

I'll talk about this in some detail later

UNIFICATION IS THE WORD

And here is my dream at a higher resolution

I'll focus on a bit further in the rest of the talk.

EVERYTHING IS EDITING

The fact that all these operations is in a sense a removing a line from a table, this fact is not superficial. For one thing, there is not so many ways how one can manage a collection of objects: it is usually some kind of list or tree. Also, there is not so many different tricks how this collection can be presented to the user and be manipulated by him. The only way people work at a terminal is browsing and editing, that is, moving the mouse, typing and pressing the PgDn key, if you think of it. Disparate interfaces are not a consequence then of some fundamental differences in user or system activities, they are simply result of evolution: different subsystem/services were written by different people and modified by even bigger crowd. There are a few pearls in this mush, though.

A FEW PERLS

hypertext for free
Rearranging file icons within a folder view and rearranging paragraphs within the document are essentially the same activity.

VERMIN of PLAIN TEXT

Not that plain text file are easier for the system to parse (not!), or it's easier for the system to process/display (not really)
It is not even that it's too easy for us human to remember the syntax of all these files. I'd mention printcap and sendmail.cf and rest my case.
So why do they proliferate?
Resource fork isn't a plain text file, but it doesn't make it difficult to mess with, given ResEdit. ResEdit is the perfect example how system configuration is ought to be managed.

UNIVERSAL DB MANAGER

Any language (C, C++) and any word-processing task (say, maintaining bibliography) would benefit from Dylan development environment with storing chunks of (definitions, comments, implementations) in the database and cross-referencing those.
weirdly-wide searchability
I put this as a quiz who can figure out first what this freak does
Surely any database can do better than a dumb search, like in netstat -a | grep finger (which is used to finger fingerer). Note, all of those scripts are DB queries (and can be elementary translated to SQL whatever)

NO FILES

Not everything between slashes (colons) is a directory name: it may be just a key. This URL does not actually imply that we want to access a file bar.html in the directory foo under the DocumentRoot on the host somehost. If foo is a script, then bar.html is just a parameter passed to that script, and can be interpreted as a file name, or anything else the script wishes. In short, not everything between slashes in URL is a name of a directory. It is just **a key** specifying an object we want to take a look at.

WHAT IS A FOLDER THEN?

Database record
Every database record (item) should have some mandatory attributes like time stamps, owner, permissions, kind (document). Beyond that, the user (or application/creator) can add anything he wants. For some records, the body is just a binary object. Records of kind "image" (or in table "images") can have additional attributes like the image width, height and depth, a key of a compressor method, etc. Thus, listing all images 512-pixel wide and with depth at least 8 with a private colormap should be as easy and elementary as viewing files in a folder by date.