Beyond the command-response model for PC-based
front-ends: Some design principles and their application

by DAVID E. TOLIVER
Institute for Scientific Information
Philadelphia, Pennsylvania

ABSTRACT

Personal computers are often used for front-end software that
mediates retrieval of information from databanks. The native
information retrieval language of databank systems usually
follows a command-response model of user interaction. PC
front-end software has often conformed to this model. How­
ever, this is neither necessary nor optimal, as more acceptable
models for user interfaces on PCs can and should be used
by front-ends. This paper states five practical principles for
breaking away from the older to the newer models. Examples
of the application of these principles are taken from the
author's experience in developing a new software product,
tentatively named CC-Mate, that assists with access to
Current Contents Search, a new database from the Institute
for Scientific Information.

DEFINITIONS AND SCOPE

A “databank” is a commercially accessible mainframe with a
selection of databases. Users access the mainframe in order to
search these databases. Databanks give access to bibli­
ographic, full text, scientific, business, and econometric data­
bases. Examples of databanks include Dialog, Mead, BRS,
Dow Jones, and services of CompuServe.

A “front-end” is any computer system placed between
users and databanks with the intent of monitoring activities or
assisting users with databank transactions. Some front-ends
provide access to a broad range of databases and databanks
while others focus on a small number of databases, often from
one provider. Three places have proven to be practical for
implementing front-ends: on the databank host itself (e.g.,
BRS AfterDark); on a shared access remote computer
(e.g., EasyNet); and on distributed PCs, (e.g., the Sci-Mate
Searcher).

Front-ends give easier, faster or more complete search
results and improve the cost-performance of information re­
triival. Front-ends do some or all of the following: assist in
preparing a strategy before the online session; automate log­
ging on and off databanks; simplify use of the retrieval lan­
guage; assist with syntactic details of particular databases;
capture results in electronic form; and process the results after
the session.

THE COMMAND-RESPONSE USER INTERFACE
MODEL FOR INFORMATION RETRIEVAL

Databanks first became available when 300-baud telecommu­
nication with TTY terminals was standard. The relatively slow
rates of 300 to 2400 baud via packet-switching networks is still
the standard link to databanks. On many databanks, the na­
tive retrieval language has not evolved much since it was first
designed: the databank prompts the user for some command
and responds to the command with results.

Most front-ends also follow this model of interaction. This
is not necessary and underutilizes a PC’s potential. Very little
of a PC processor is needed to handle the arrival of asynchr­
onous characters. Most front-end packages, though, use the PC
processor for much of the session only to serve characters to
the screen.

THE FULL-SCREEN USER INTERFACE MODEL
ON PERSONAL COMPUTERS

In contrast to the slow serial command-response model, the
entire video display of a PC can be refreshed almost instantly.
The screen can be divided into regions through which the user
can navigate with the mouse or cursor. Windows imply addi­
tional regions “hidden” behind the screens. Internal memory
and external mass-storage accommodate programs, support­
ing data, and even complete electronic transcripts of results.

The principles below emphasize taking advantage of the
full-screen user interface found in many PC application pack­
ages. They have been applied at ISI in the development of a
new front-end program, CC-Mate. This program assists users
of a new ISI database, Current Contents Search, available on
the BRS databank.
DESIGN PRINCIPLES FOR FRONT-ENDS AND APPLICATION EXAMPLES

Principle 1: The front-end should support user activities before and after the online search session itself. CC-Search's built-in full-screen editor allows users to prepare their search profile. Data tables with details about the structure and content of the database can be recalled. The same edit commands can be used to clean up transcripts of search results, both during the session and after logging off.

Principle 2: Front-ends should provide ongoing status and options as well as innovative features made possible by full-screen access to the profile and results. Detailed status is continually on display in CC-Mate; action and information options are continually available. Terms in the results can be selected as profile queries by "pointing and clicking"; the full text of referenced items can be ordered from ISI or from the reprint author by pressing a single function key.

Principle 3: The user and database interfaces should be isolated from each other in their own modules of the computer program. These distinct interfaces are linked only in a main event loop. Code isolation simplifies maintenance and makes it easier to extend the product to support users of other databases.

Principle 4: Temporal interleaving of the user and database interface activities improves the cost-effectiveness of the front-end. As results continue to stream into buffers from the serial line, the user can examine and edit all results, summaries, status data, and options. Immediate information gives the user more control over the session, while not halting the incoming flow of results.

Principle 5: The functions carried out by the user and database interfaces need not correspond one-for-one. For example, CC-Mate merges the Select, Limit, and Print functions of BRS under a single function called Display. This gives users results from a query in a single step. Also, when the user alters a query in the profile, the sequence of commands to the databank will be redirected without changing the profile's own sequence with which the user is familiar.

REFERENCES