
Apple Manuals


Apple][Pascal


Apple Macintosh


][Swyft Card


Canon Cat

Jef Raskin Information

Information Appliance

Trade Secrets

(see e-mail from 03 March 95 for info on how obtained)

DOCUMENT NO. 34

COMPILED BY
DAVID T CRAIG
736 EDGEWATER, WICHITA, KANSAS 67230
U.S.A.


Apple Manuals


Apple][Pascal


Apple Macintosh


][Swyft Card


Canon Cat

"RaskinDoc034_0.PICT" 134 KB 2000-02-21 dpi: 300h x 300v pix: 2015h x 2798v

THIS IS NO LONGER SECRET PER J.RASKIN

Jef Raskin Information

Information Appliance Inc. Trade Secrets

At your request I am sending you a draft list of the trade secrets owned by Information Appliance and by Dr. James Winter and myself. With respect to these items of intellectual property I am the sole agent with authority to negotiate on behalf of the owners.

These trade secrets-along with other trade secrets and the patents, copyrights, and trade marks already listed in a previous document-comprise the intellectual property basis on which ---- are building a novel and potentially very profitable technology. While some of these trade secrets were built into the Canon Cat, they are not apparent because Canon never pursued the planned development path which would have exposed them to public view; thus they remained secret though incorporated in a product. In some cases only part of a trade secret concept was revealed in the commercial product. One example is the automatic control and use of mass storage devices; in the Cat the DISK command represented only part of the complete idea for controlling mass storage, which thus remained a trade secret.

The trade secrets are covered by the mutual non-disclosure agreements executed between ---------- and include:

1. The most fundamental of the trade secrets is the concept of and general means for the implementation of a general-purpose editor, which can operate on all kinds of data including alphabetic, numeric, and graphic objects, and which replaces the usual operating system from the point of view of the user and equally from the point of view of the system designer. The editor includes such operations as deletion, replication, motion, transmission, reception, and transfers to and from mass storage directly by the user at any time without preparation or having to change modes or enable applications.

2. The method of interface design that allows modelessness in complex collections of abilities that are the equivalent of "applications."

3. The extensive and characteristic use of psychologically advantageous quasimodes (we now prefer the term "pseudomodes") to simplify operations and to eliminate psychologically damaging true modes in systems.

4. Methods of designing an interface that is at the same time easy to learn for beginners and efficient for experts. Until the creation of these methods, interface designers believed that these two properties were in opposition and that satisfying one denied the other. In fact, most workers in the human interface technology area still believe that these two properties are antagonistic.

5. The use of a single fast search (LEAP, now patented) as the basis not only of searching but of accessing large data bases while at the same time replacing cursor motion controls in text. This includes the realization that as more remote documents are searched longer patterns tend to be used; in conjunction with a sub-linear search algorithm that speeds with longer patterns, this tends to keep search times constant.

6. The unification of all structural elements (e.g. page breaks) as pure characters (or of graphic structures as graphic elements themselves) that behave in no wise differently than any other characters (or structures), which effects a simplification of both the user's mental model and the implementation.

7. Designing systems such that there is an incremental learning path so that each feature as learned is immediately and clearly useful, and yet builds on previous features and leads to complex structures without requiring any mental leaps on the part of the user. This is in strong contrast to most systems where many details have to be learned and remembered before the user is able to integrate them into a mental model of a complex construct.

8. A method of designing a system to simplify the work of third-party and other developers of software for that system along with a structure that helps assure that such extensions will adhere to the interface paradigm originally created for the product.

9. A display method that unifies the locus of user attention.

10. Methods for including setup and other parameter storage in the user's document paradigm.

11. The presenting of help and error messages as ordinary text, and of making these presentations modeless.

12. The elimination of the need for escape sequences after system messages or error messages are presented.

13. An improved method for keyboarding overstruck characters.

14. A method for allowing the uniform use of object-action (noun-verb) paradigm whereby only selections are acted upon (e.g. for printing). While the desirability of the noun-verb paradigm is widely recognized, all present systems fail to use it uniformly and the designers of such systems believe it is impossible to achieve a pure noun-verb system.

15. Recognition that on/off (whether of the system or any feature) is a mode and methods for building a system where this modality is removed.

16. The fully automatic storage and retrieval of information from mass storage while eliminating most of the potential for making errors during this process.

17. The treatment of programming, debugging, and program execution in various languages as ordinary text manipulations.

That these ideas are not obvious is clear from the fact that it takes a while for even proficient computer scientists to absorb this collection of ideas. Yet, once understood, they seem obvious and as natural as the present method of designing software system architecture, and have clear advantages both in terms of implementation efficiency and interface quality.

END OF DOCUMENT