LEA : A LANGUAGE FOR MULTIAPPLICATION INTERACTION ON THE BUROVISEUR

Najah NAFFAH*

INRIA, B.P. 105 - 78150 Le Chesnay - FRANCE

Editiorial remark:

The oral presentation to this paper will be presented under the title "Interfache rules for office work at the bureauviseur" LEA, the language, summarizes these interface rules in a consistent manner.

* The author acknowledges the editorial and the translation work that has been done by Reinhard Kofer, SIEMENS - WEST-GERMANY.

1) Abstract.

This contribution proposes a simple language for manipulating "electronic objects" in the complex world of integrated office system applications. The objects are displayed on the Buroviseur screen by the different application programs within the KAYAK system. The Buroviseur is the name of the office workstation [2,3] constructed at INRIA, in the KAYAK Pilot Project [1]. Each of these application programs is operated by the user, and each owns an Information space composed of "electronic documents" by the means of which the user interacts according to different modes of operation :

- Acquiring information and results of invoked actions.
- Changing or upating information.
- Entering or creating information.
- Deleting information.
- Copying and moving of information from one space to another.

In the design of the language, we have started by indentifying this list of common interactions that are applicable in each service and obtained a systematic but simple method to use the set of services whatever their nature is.

Whatever the requested operation on an electronic document may be, the user utilizes command menus to which he points, or he uses parameter lists which he fills in to express his requests. For the pointing action a mouse device is used.

The manipulation language we are proposing is based on graphic, and sometimes vocal interactions with the office workstation. Its originality resides in providing the user with a common set of interactive rules to talk with each application without having to remember the particularity of the application, and to communicate objects between the applications.

2) Introduction.

From the start of the KAYAK pilot project [1], we focused our work on the design of a full set of services that coexist together in an integrated system offering the user a whole range of facilities. The following services are now implemented or under implementation :

- A distributed electronic mail system (AGORA) [4].
- Document production (PLUME) [5].
- Storage-archiving and retrieval (VISION) [6, 7].
- Telephone activity management (ATELO) [8].
- Calendar and Clock [9].
- (Numeric) calculator.
- Various professional applications.
- Emulation mode for Data Processing and Telematic applications.

Faced with this variety of applications through one common workstation, it is mandatory to offer simple standard rules for communication with the services. This will avoid rejection phenomena against a "sophisticated" but badly designed system, where the user has to readapt to each application and remember the specific dialogue rules. Likewise it will increase work efficiency with electronic documents.

In addition to this ergonomic factor, a technical criterion is taken into consideration. It deals with the nature of tasks that office workers execute and also with the permanent relations (interdependencies) between those tasks : for example, a document is archived, after it has been received, by

either electronic mail, or produced by the Editor, or sent through Electronic mail. An archived document is consulted in parallel to an ongoing telephone conversation.
For this reason, the office environment can be defined as a dynamic "milieu" of interdependent tasks that are executed by those who work in offices.

In such a context, it is necessary to provide the user with simple ways to ask for information and control transfer between communicating applications.
Our solution tries to meet those two requirements.

3) Solution.

The solution has been defined around the notion of a unified command language that acts, on one hand, upon the documents in the application information space, and on the other hand, insures communication between applications when needed.

We call the interpreter of this language LEA (for: Langage d'Exécution inter-Applications). This language will be presented along practical examples of dialogues with applications, accompanied by typical representations on the screen.
The following conventions are used :

- The applications are defined by "principal actions". Each action is represented on the screen by a box containing a verb or a symbol.
- The information space managed by an application is represented by a named electronic document. Each electronic document is made visible on a screen window (or

physical viewport), where the window dimensions are intially chosen by the application. This window can be in either one of two states : "VISIBLE" or "REDUCED".
In the VISIBLE state, the information that the application wants to communicate is displayed in the window.
In the REDUCED state, the window is shrunk to a box in which we find a name assigned to the window by the application.

- The state transition from VISIBLE to REDUCED can be demanded by either user or application program.
- The entire electronic document in a window carries a name, which is displayed in a rectangular stripe on top of the window.
- All windows will be situated in a (screen) zone called the "active document zone".
- The electronic documents contain information objects that are manipulable by the users :
    - . Backward scrolling.
    - . Forward scrolling.
    - . Logical objects scrolling.
    - . Copying of objects to target windows.
    - . Moving of objects to target windows.
    - . Deleting an object.
    - . Etc...
- An application can put several windows on the screen (e.g. browsing a file of stored documents).
- A "descriptor" window is displayed under application control. It contains fixed and variable fields and ser-

ves to express the user request (for example : filling out a message header for a message to be sent, or for parameter acquisition for retrieval in the archival data base).

- The applications communicate with the screen in two modes : "virtual mode" and "real mode".
- In virtual mode the window handler is provided with a standard representation of the generic document [10] which is independent of any physical representation of data. The mapping to physical viewports is done by an appropriate module in the screen manager.
- In real mode, a bitmap is handed over to the screen manager and is directly display without additional processing. (E.g. a facsimile page, or a Videotext alphamosaïc page). It is absolutely necessary that the application announces the communication mode over its interface to the window handler in the early dialogue phases.

4. LEA - Functions.

LEA functionalities are divided in three categories :

- The first one concerns the Initialisation phase where protection mechanisms are managed, and a global view of the available services and their commands are presented to the user.
- The second category deals with a common set of functions that apply when interacting with documents produced by

any application. This set is called the set of Generic Commands, and will be explained below.

- The third category corresponds to a first control of the commands directed to the applications which make the final processing. This control provides a unique and coherent type of interaction between the user and his multiapplication environment. The concept of "DESCRIPTOR VIEW" is introduced to achieve this goal.

4.1 Initialisation.

In this function, LEA manages the start-up, and the initial presentation of services.

4.1.1 Start-up.

After power-up of the Buroviseur, the user is prompted for his identification and a password. At system generation time, this user identification is entered (in general, this is the user's name) and a default password is assigned to the name. Later on the user can change his password as often as he wants to do so. Each time he must supply his old password (before change). In general, the password is alphanumeric and is introduced on the keyboard.

However, for added security reasons, the user is allowed to enter a vocal password that may exist alone, or is added to the first written password.

After the system accepts the password, the user may select any service he wants to activate. The following figure lists

the services available on the office automation system, as they appear in the service zone, along with their "principal actions" and the relevant explanations.

We propose that a fixed zone on the screen be reserved for the display of symbols which represent those services, and for their main commands, as outlined in fig. 1.

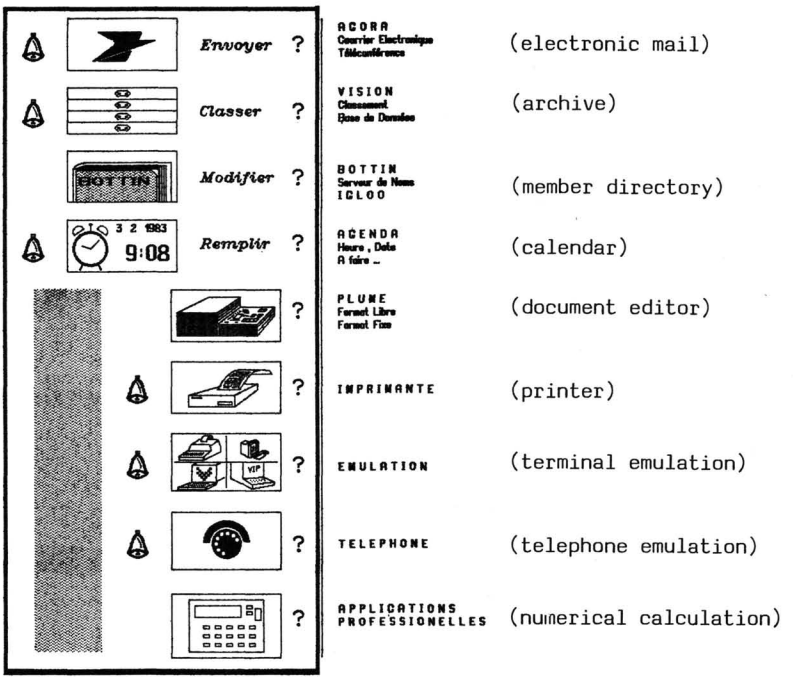

Fig 1 : Service Zone.

However, the interaction with some services will necessitate the display of information on the whole screen (e.g. display many pages of the same document, show the tree of filed documents). The situation leads to the provisional deletion of the service zone. In order to activate a new service, LEA will provide a mechanism to call upon the services window by pointing to a unique box that may exist systematically in a fixed location on the screen, or is shown when a special combination of the mouse keys is generated.

Anyhow, there must be a preplanned screen zone, where LEA can be started to activate the services. After start-up of some services and manipulation of some documents, the screen will appear as shown in fig. 2.

In the active documents zone, the documents appear either by their name only (REDUCED state) or by their name plus the contents of a logical document page (VISIBLE state).

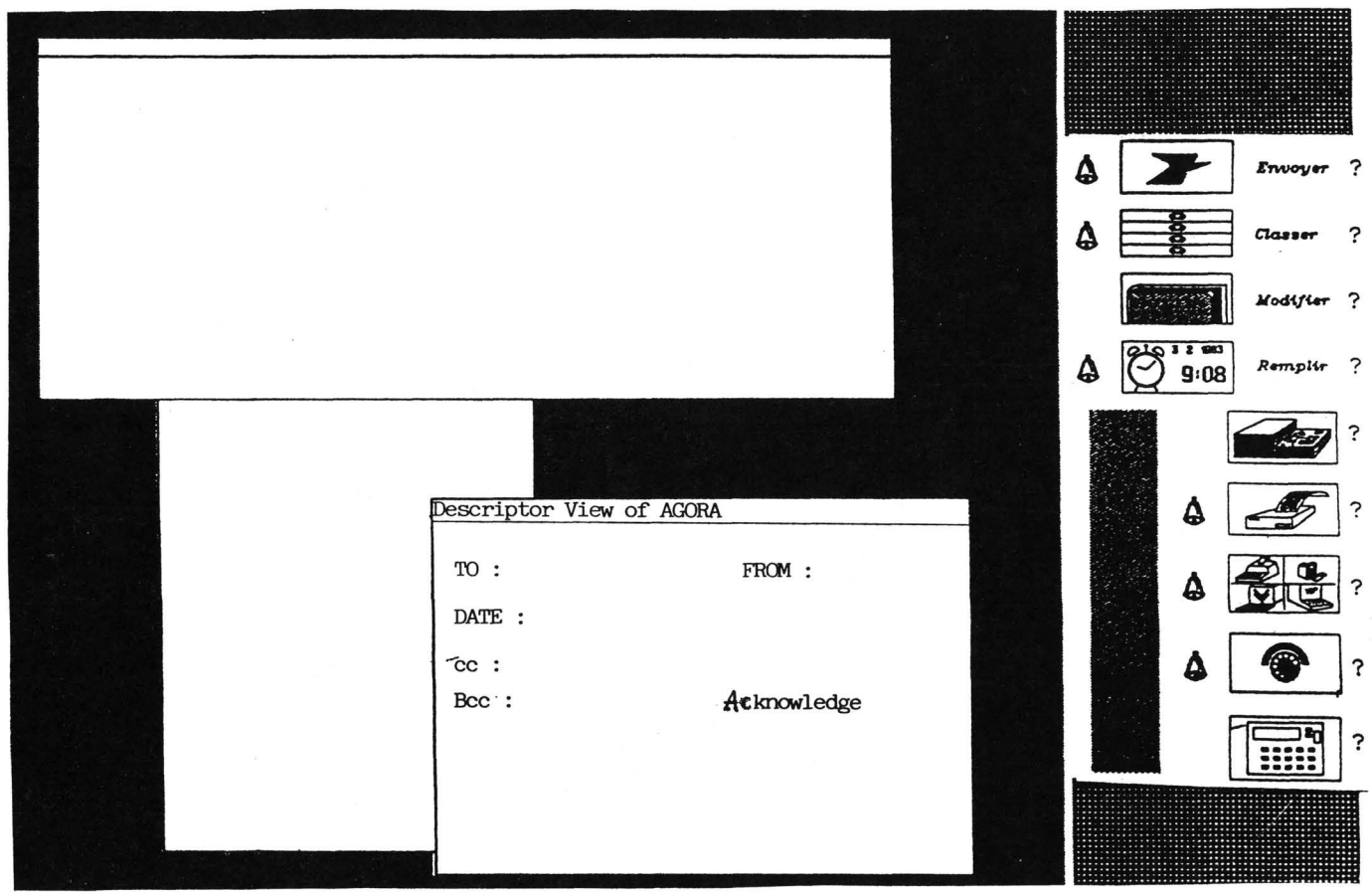
ACTIVE DOCUMENT ZONE
SERVICE ZONE
Descriptor View of AGORA
TO :
FROM :
DATE :
cc :
Bcc :
Acknowledge
Envoyer ?
Classer ?
Modifier ?
Remplir ?
9:08

Figure 2.

4.1.2 Presentation of services.

4.1.2.1 Implicit Action.

Most of the services store an information space that can be visualized for the user upon his request (e.g. Mail).
We assign to each of these information spaces a symbol which represents the information. Pointing to this symbol is implicitely translated by LEA as a READ-request for this information space. This way of thinking applies to :

AGORA : incoming and outgoing electronic mail ;

VISION : finding a document in the file system or in the private or shared databases.

BOTTIN : consulting the particular directory database

CALENDAR : read the tasks of a specific day.

4.1.2.2 The notion of "signal".

Certain types of information space are in an active state. This means that they accept the arrival of an event (external and internal) and translate this to the user in a meaningful way, for instance, in the case of a mail-box that receives a message at a given time. Another typical case is the one that corresponds to a stored documement which should be processed at a specific date. When this constraint is satisfied, the signal is activated.

In VISION, many files can be stored with this automatic rewakening feature. The same automatic remembering applies to the AGORA service. That's why it is interesting to introduce the concept of SIGNAL, that, on the screen, appears as a little

bell close to the information object to which it is attached, that rings eventually, with speech synthesis, or moves visually when the signal is activated.

Likewise in AGORA, the BAL-symbol (mailbox = Boite Aux Lettres) is equipped with a signal, the "Armoires"[*] in VISION own a signal each. And the same is true for CALENDAR to manage a TO-DO event.

[*] English translation: "Drawer"

4.1.2.3 Explicit actions/activities.

Another class of services, for which the visibility of the stored data is less important, resorts to direct activation. Let us list some examples that correspond to already mentioned services :

- For AGORA, activating the "SEND"[-] necessitates a direct action of pointing to an abject. This object presents itself in the form of an electronic document already to be found within the active document zone.

[-] French Original: "ENVOYER"

- In VISION, activating the "STORE"[+] results in the action of filing an object that has to be pointed to, or in the action of directory manipulation ("change of archiving space"), where a directory node is created or destroyed.

[+] French Original: "CLASSER"

As far as the other services are concerned (PLUME editor, printer, terminal emulation, telephone, professional tools), action is immediately started when pointing to the corresponding box in each service placeholder.

In PLUME the user chooses between blank sheet mode (i.e.

formatless editing) or specialized document mode (i.e. editing with predefined page properties).

In IMPRIMANTE (the print service), after having activated the service, the user will point to the object to be printed among those in the active document zone, or he will input the name of the document to be printed in a suitable window (cf. later : "descriptor window"). A SIGNAL is highlighted at the printer placeholder in order to indicate to the user whether work (i.e. printout) is already terminated.

In the EMULATION service, some boxes which contain the different terminal types (that are emulated) represent the visible interface. Activation can take place by pointing to one or more (terminal placeholder) boxes, or by pointing to a single box several times to open multilinks to the same service. The emulated terminal interfaces are :

- teletype
- videotex
- 2741
- 3270

A signal is highlighted by each of these boxes in the case of an outside call.

In ATELO ("telephone" activities), pointing to the placeholder launches dialing action in either real time or deferred time mode. In both cases the user enters appropriate parameters in the descriptor window. This service again will highlight a signal box upon arrival of an outside call.

The professional tools (like VISICALC) are activated by pointing to the respective boxes.

4.1.2.4 Administration.

)* i.e., the "?"-Symbol

For each service presented in Figure 1, there exists a box that contains a "query point")* which means a "HELP" trigger. Pointing to this symbol will result in a state change, whereupon the user can inquire about the usage of the service, or can change current parameters, or can ask the service about the status of the ongoing computation.

The detailed semantics of this HELP -command will depend on the particular service being explained. Here we give some examples to help in understanding the concept.

In AGORA (electronic mail) : it is possible to perform dialogues with the mail administrator program, in order to obtain information on members and managers of lists, on the state of sent messages, or on general usage hints. Composition of these different requests is done via a descriptor window.

In VISION (file system) : we can obtain statistics on retrieval frequency, on creation of new nodes in the directory tree, on the volume of stored documents, on remaining free disk space.

In CALENDAR : the administrator program gives usage hints for the calendar as well as statistics on held meetings, on conferences to particular topics, etc.

In PLUME (text editor) : current usage is indicated as well as the different specialized ducument types.

The IMPRIMANTE (printing service) : gives information on the stored formats, their properties and characteristics.

4.2 Generic commands.

The philosophy behind the generic commands, is that they apply to all information objects managed by any type of application, and provide necessary control mechanism which is useful in any application context. Thus we define a certain number of common commands, that appear in all sessions with the above mentioned services.

LEA executes these generic commands on behalf of the user. It is important to point out, that certain commands act on electronic documents within the user's service information space as opposed to the data space belonging to LEA.

The commands acting on service data are :

* COPY Copying a source object to a target position.
* MOVE Placing a source object into a new target position. This command keeps the name of the object.
* DELETE Making an object invisible in the active document zone without tampering with its existence in the file system or work space.

The commands not directly effecting information service are :

* STOP/RESUME : Interrupt command execution for later continuation (without state loss).
* CANCEL : Break off an ongoing command execution.

The interface consists of a menu owned by LEA that will be found in the service zone (cf. Fig. 2). Screen placement of these commands can be initiated by the user. The commands reappear at the same position whenever the LEA menu is activated by a special mouse key combination.

The COPY-function needs to be illustrated in somewhat more detail, because it allows for a first inter-service-communication level, under coordination of the user :

- An electronic document can be copied as a whole. It suffices to point to: the COPY command, the document name, and the target position.
- Above that, an elementary object within an electronic document that is in VISIBLE mode can be copied as well. To achieve that, we "point to" COPY, then we delimit the object to be copied (i.e.: two positions for a textual paragraph, one "mouseclick" for a graphical object). Finally we indicate the position of the target place.

The user is then invited to give a name to that object being copied, which becomes an identifiable electronic document in the LEA workspace. This name is input into the upper window frame which is maintained by VITRAIL, the buroviseur multi-window manager [11]. On this new object the same actions can be performed as on any other document in the active document zone.

5. First Level Command Control.

LEA keeps a first level control of the coherent execution of the different commands which were activated by the user with each of the services or with the common command set, that belongs to LEA. Towards that goal, we apply a certain rule that is presently evaluated on the ergonomy level, and that consists of adhering to the following logical steps.

1. Choice of placeholder symbol in the service zone.
2. Proceed to step 4, if the service belongs to the class of direct actions.
3. Mark the object and eventually delimit its position endpoints.
4. Skip step 5, if parameter set is complete.
5. Fill in the descriptor (mask in the descriptor window)
6. Confirm or Cancel, using the A-key (right-most) on the mouse.

The new element among these steps is the filling in of descriptors in step 5. We propose to keep available a "mobile" window that pops up at the same place anytime the situation demands it. This situation is determined by the "nature" of the activated command.

Despite the tight connection between the contents of this window and the application type, we can imagine that LEA maintains a certain "command context" in order to display that (descriptor) window according to the command that the user has activated.

This window contains fixed and variable fields for utilization by the user. We call this window the DESCRIPTOR VIEW (fig. 2). Here are some examples of descriptor windows.

For the AGORA command SEND, the Descriptor View corresponds to the message header as it is shown in Fig. 2.

For the AGORA implicit action of viewing a MAILBOX content, the Descriptor corresponds to a table describing Attribute-value pairs describing the message (SOURCE-NAME, DATE-OF-SUBMISSION, SUBJECT,...)

The user launches a query on the mail-items by filling in the variable fields, leading to the display of a subset of the messages. After this display, which happens in the active document zone, the user can choose to activate more detailed commands of the message system such as :

- Forward to a Destination or to a Distribution List.
- Reply.
- Circulate according to a fixed pattern.
- etc..

All these commands are displayed in a separate menu in the lower part of the descriptor. Common commands can be applied to the objects as said before, as well as the commands belonging to the VITRAIL window-system (e.g. scrolling).

Another example is the interaction with the VISION file system, where the descriptor serves to either specify a direct search based on specific terms, or a "navigational search" by displaying different horizontal cuts of the hierarchical tree of documents in the database.

In PLUME blank sheet mode there is no need for a descriptor. in fixed format mode, a descriptor is generated that lists the Specialized Document types available in the system [10]. The user chooses by pointing to a list item.

In the PRINTER service, the descriptor shows a list of Templates or Formats (page attributes, line attributes, font sets) that can be applied to generic documents.

6. Implementation Tools.

The basic building blocks for such an interface are the available tools in the buroviseur for managing bidimensional objects (e.g. data windows, commands menus, ...). The first tool, called VITRAIL or Multiwindow manager [11], allows for displaying many windows for the variety of activated applications. Windows may overlap according to user wishes. VITRAIL keeps the context of each window and redisplays hidden objects whenever a window or a part of a window is moved. Every window can be expanded or reduced to its label. This is very useful for operation on the limited physical space of the screen in multidocument context.

Scroll bars surround the window, and are used to move the viewport in the four directions on the information space. Figure 2 shows a hardcopy of the screen produced during VITRAIL implementation. Another low level tool, called RASTER-OP [13], is worth mentioning here. It constitutes the most elementary building block for manipulating objects on the screen. VITRAIL and all other application use heavily RASTER-OP that consists in executing boolean operations between two rectangles of pixels having the same dimensions. A combination with a predifined MASK of pattern can be done to produce more significant images. RASTER-OP has been implemented on the Buroviseur and is described in [14].

CONCLUSION.

We estimate that this view of the end-user-interface to LEA renders the system more flexible to use, and sufficiently coherent on the level of "Step systematisation during command formulation". Above that, a minimal tool set of basic actions can be chosen, by the application which the end-user works in an optimized environment.

The work presented here has taken benefit from discussions with many people in the KAYAK team (MAZOYER, LARGILLIERE, SCHEURER, BERBER, WEGMANN, KARMOUCH, SZTAJNKRYCER and from pionnering work at XEROX PARC [15]. I acknowledge here in particular the work of D.BERBER, A. WEGMANN and B.MAZOYER on VITRAIL and RASTER-OP.

It should be clear that these specifications will not be finished before the completion of an implementation and ergonomic evaluation in a real operation field.

BIBLIOGRAPHY

[1] N. NAFFAH
Distributed Office Systems in Practice.
ONLINE, May 1981 - London U.K.

[2] N. NAFFAH
Poste de Travail à Interface Universelle.
Convention Informatique - Paris 1979.

[3] B. SCHEURER
Office Workstation Design.
OIS, St. Maximin 1981 - North Holland Publishing Co.

[4] N. NAFFAH, A. KARMOUCH and G. FRANTZ
The AGORA Message Systems Architecture.
COMNET 1981 - Budapest HONGRIE.

[5] N. NAFFAH
Editing Multitype Document.
OIS, St. Maximin 1981 - North Holland Publishing Co.

[6] F. FERNANDEZ
Le Classement et la Recherche Manuel de documents Electroniques.
KAYAK's Internal Publication. November 1982

[7] S. RADIA
KAYAK's Filing System.
KAYAK's Internal Publication. January 1983.

[8] N. NAFFAH, P. CHARBONNEAU
ATELO : un service de gestion des activités téléphoniques.
KAYAK's Internal Publications. February 1983.

[9] N. NAFFAH
KAYAK : Les Espoirs de la Recherche Française.
Science et Avenir, Numéro Spécial 40 - Bureautique 1982.

[10] N. NAFFAH
Description du Document Electronique Intégré.
KAYAK's Internal Publication. February 1983.

[11] A. WEGMANN
VITRAIL : un outil pour la manipulation interactive de documents.
To be published. February 1983.

[12] R. PURVY, J. FARRELL and P. KLOSE
The Design of STAR's Records Processing : Data Processing for the Noncomputer Professional.
ACM. Trans. on OIS - January 1983, Vol. 1, n°1.

[13] W. NEWMAN, B. SPROULL
Principles of Interactive Graphics.
Mc Graw Hill - Second Edition, 1979.

[14] B. MAZOYER
Primitives d'utilisation de RASTEROP dans le Buroviseur.
KAYAK's Internal Publications. February 1983.

[15] A. KAY
Microelectronics and the Personal Computer.
Scientific American Offprints n° 384 - Sept. 1977.