In italiano, per favore

En español, por favor

Dr. Gabriele Florìa DDS
VJO editor

English translation by
Dr. Andrea Marinelli

Original Article
Published on 15-12-00

Evaluation of computer software in an orthodontic office

INTRODUCTION

The use of a computer in an orthodontic office is nowadays very common, video-writing, filing of bookkeeping and personal data, management of radiographic analyses, in fact, find in it an essential technological support.
Therefore as any other instrument in our office, its cost to benefit ratio must be evaluated, but the nature of the tool requires knowledge of hardware and software to obtain an efficient and advantageous binomial in function of specific needs.
The financial investment made in order to prepare managerial and operative procedures, risks to suffer heavy losses if you decide to change the operating system or the information platform after a few years. Then a careful reflection becomes necessary to make conscious, reasoned choices, unconditioned by the marketing, which could come out as a disadvantage in the medium-long period.
Inexperienced users often neglect the choice of the personal computer's operating system in the orthodontic office. In reality, considering the choice of the application program as a priority and adapting ourselves to specific requests, we often risk to lose sight of the binomial processor-operating system. This represents the real "engine" of our computer, and many other characteristics that will influence all future hardware and software.

Definition of operating system

The computer's operating system is that particular software the machine loads when it starts and which allows the interaction between various hardware units and the CPU (Central Processing Unit), moreover it supplies services and support to application programs. (3)
Schematising we can distinguish 3 parts in it:

1. **il kernel**
2. **la shell**
3. **the utility programs**

This operating system fits for "power user", i.e. that ones who have high-level technical knowledge, supposing the necessity to use the basic characteristics of the operating system, like the kernel recompilation and compiling of self-made application programs. But it can also fit the common users that want an operating system compiled to measure for their hardware and that interacts with the computer through a specific application program, like a cephalometric program or a management program for patients or images. Moreover it fits anyone who has to manage a medium to large sized computer network due not only to its free licensing policies that reduce operating costs, but also offers a very good product with an high degree of stability and security. In the orthodontic field however, the biggest disadvantages is the availability of related software, because in recent years, most software development has been concentrated in developing programs for other operating systems.

UTILITY PROGRAMS OR SOFTWARE

Utility programs are a group of little application programs, which are primarily used for the system management. In theory they could be in a shell, but in reality that is not the case.
Traditional shell programs contain very commonly used commands and functions. It's often difficult to distinguish between the shell's functions and some utility programs. So the operating system provides functions to control data flow, locating sources, organizing and storing data location, maintaining the right level of security, and error checking. These very important functions are typically invisible to the user. Users commonly interface with application programs, which allows users to save clinical folders, cephalometric analysis or anything that they choose for their own functionality.
Sometimes we blame the programmer for application program's lacks, but in reality the problems are mostly due to hardware, operating

Kernel

The term "kernel" comes from the German word "Kern" which means core (even soul in technical sense). In fact the Kernel is the heart of the system and represents the lower interacting level with every program for every kind of operation. It controls inputs and outputs (data flow) to and from peripherals. It locates the resources in the required functions, and provides space to the management for the storing of data. So it has a central role and its correct operation affects almost every hardware and software system. Moreover different kernels with compatible functions promote that characteristic known as "**portability**" or rather the possibility to transfer the same programs on different platforms (1,4).

Shell

Kernel was often described as a "pearl" so the "shell" cannot separate it. It surrounds the kernel allowing the users to interact with the system. The shell can have different shapes, from the simple command line with a prompt, such as DOS, to a graphic interface with icons or a voice controlled system. They are all shells.

THE CHOICE OF THE OPERATING SYSTEM

MS DOS

It's nowadays an old operating system, but it is still able to run on Intel 286 or superior processors and needs 640 KB of free memory. It's not multi-user (it doesn't support more than one user on the same machine), and it's not multitasking (it can't execute different programs at the same time) and is not multithreading (it can't execute different portions of the same program at the same time). Moreover MS DOS isn't much able to interact with other operating systems, it has great limits in the memory management, it doesn't include networking programs, or utility programs for development (like editors and debuggers).

WINDOWS 95/98

It's preinstalled (so it's prepaid) on the majority of computers on the market, it's known by many users (so it's easy to find help when there are some problems) it's used by many programmers (so it's easy to have programs for specific needs). It has an intuitive graphic interface. It allows networking with other PCs which have the same operating system, even if; according to the author; it adapts itself better on the single PC. Computer viruses, which can cause of loss of data and destruction of operating system, can infect it, so it needs an updated protection system (Anti-virus software). The differences between the two systems, 95 and 98, look scanty since the user perceives only few performance improvements, but in reality the kernel system or to the unperfected co-operation between hardware and software.

EVALUATION OF SOFTWARE'S QUALITY

An orthodontic office needs many programs: a managerial program used for managing the financial aspects of a practice as well as managing staff, office policies and the day to day scheduling; a cephalometric software; a database for patients' clinical folders; archives of images and digitalized radiographs. If the entire needs of an office's computing needs are not met at once, the office may encounter bigger problems when the various products try to co-exist and integrate with one another. It may seem that exchanging data between a few systems is a trivial problem, but it becomes enormous when the size of the data grows. So, in my opinion, we have to choose the platform (Microsoft, Apple Macintosh, Unix, BeOS etc…) we want to use to accommodate our needs and then search for products available on the market that can satisfy our requirements. If we are not able to find what we are looking for we can ask a programmer to custom develop software that will meet our needs. In both cases we have to be able to evaluate the software's quality. Some of the problems are innate (such as scalability) while others can be measured by the end user:

1. **Reliability**
2. **Accuracy**
3. **Strength**
4. **Usability**
5. **Verifiability**
6. **Maintenance**
7. **Reusability**
8. **Understanding**
9. **Interoperability**

RELIABILITY

It's the agreement between the program and its requisites. Every machine, even a computer, is reliable if it works as expected. It may seem banal, but a real reliability in software programming requires a diligent team to work between the end users and the programmers to identify the user's needs. A debugging phase would lead to the final release of the program. This phase includes checking and eradicating the program of run time errors of preliminary versions.

ACCURACY

It could be considered as a part of the reliability, but this term exactly identifies the agreement between the program and its specifications. To clarify this, we would define a requirement document that outlines what the user wants and specifies what the programmers believe the user wants. It 's a very important point, because only those who are both a programmer and an end user are able to do very good specification and requirements document. In fact, often the user thinks about operating needs, while the programmer thinks about the logical algorithms to develop the program. Rarely does conversation between the two parties bring the best compromise.

STRENGTH

It's the ability of the program to be reliable in abnormal situations, such as human errors or other external factors. This shouldn't be confused

of Windows 98, evolution of Windows 95, was improved for stability and memory management.

WINDOWS NT 4

Windows NT is available on Digital Alpha and x86 Intel processors only. It's multi-user, has multitasking and multithreading skills. Its graphic interface is very similar to W 95-98 and this makes easier passing from one to another. It was designed for networking, i.e. NT means network; but it is not so easily configurable and lacks the plug and play interface of Win 95/98.

APPLE MACINTOSH

Apple's operative system for Macintosh works on Macintosh only. Moreover it suffers from the lack of development tolls, poor interoperability with other operating systems, primitive multitasking, and the absence of multithreading. The latest System 9 versions solved many problems by improving the multitasking ability and the whole system's reliability. The Mac's networking ability was also improved, making this platform probably the best graphic and desktop publishing workstation for not professional users.
Unfortunately, application programs for this operating system are poor in the orthodontic field, even if recently many American programmers presented themselves again by making very good management software, exploiting the ease of using of these computers.

SOLARIS 2.6

Solaris is a very good Unix operating system, according to many experts it's the best on the market for stability and reliability. It's rather expensive and needs Sparc and Intel platforms. It's multitasking, multi-user, multi-platform and multithreading. It's preinstalled in Sun computers which in general are used for mathematical-statistical calculations or for scientific research, but it' even separately purchasable. It requires a high level technical knowledge, and has an essential graphic interface. Not so long ago its source codes were available free.

OS/2

It was the first very popular operating system having a graphic interface based on icons, anticipating, in the 2X version, some characteristics, which become popular with Windows 95. Even if it has a fairly good number of fans, it's not so popular. Its greatest problem is that its producer, IBM, abandoned it in favour of Windows NT.

BeOS 4.5

It's the latest operating system on the market, and since it came out (1995) was planned to manage great audio, video and bi/three-dimensional data flows. It came out with the purpose to set free as many sources as possible for application programs. Moreover it's able to manage many processors on the same computer and has a 64-bit file system (nowadays the only one for PC). These prerogatives make it very powerful in multithreading

with the concept of "user-friendliness" or with usability, because it isn't only the interface. A good program has to prevent incorrect inputs and the entry of wrong data in the database. A good program will correct the entry at the time when it is first input.

USABILITY

It's the ability of a program to be immediately understood by the end users. This is very important, as the users are not experts. It's not surprising that many power users prefer a text interface instead of sophisticated icons. Programmers say that the interface's intelligence, if we can so define it, has to be inversely proportional to the end user's one. This just says that inexperienced users have to correspond to "error proof" programs.

VERIFIABILITY

It's the possibility to define an algorithm, which can reveal a program's defect. Different kinds of programs are available: mathematical ones, for example, can easily verify anything that does not relate to the databases. This leads to the distribution of completely unit tested applications to the end user.

MAINTENANCE

Often software has to be updated for accommodate for the continuous changes in the customer's needs. This term defines the ease with which these changes are implemented.

REUSABILITY

It's the ability to use parts of the program within another application program.

UNDERSTANDING

It's the reading grade of the program's code and documentation: it indicates how much it could be understood by someone who has not programmed it.

INTEROPERABILITY

It's the ability of a program to interact with software and hardware that surrounds it. It monitors the exchange of data and it is very important for the end user.

SOFTWARE'S LIFE CYCLE

Just like other products, the software has a life cycle, or rather it's characterized by a series of events that identifies its beginning, use and its end (5). There are various models, which define a product's life cycle. The one that stands out, for its logic and its clarity is called "like a Waterfall". It forecasts a series of consecutive events that provide at the end of every phase an output than represents the input for the next phase.
In detail these phases are:

REQUIREMENTS GATHERING: it's a list of functions that will be improved, if the programmer and the user are not the same person. This phase consists of a series of interviews with the users to discover what the program should be able to do. As we have seen, the correspondence and a partial knowledge of the two fields of work, in our case orthodontics and computing, represents the "bridge" element which makes possible the development of a good product. It's not by chance that software houses that dedicated themselves to a specific

and multitasking and it's also able to manage different file systems (MacOS, HFS, FAT16/32, VFat, Linux ext2 etc...) and it has got all the instruments for intranet and internet, but it seems not very suitable in multi-user mode. Its most important deficiency seems to be youth and relative lack of application programs, but having very good characteristics, seems to be the rising star in world-wide computing scene.

Linux

Every operating system has its own kernel, but this statement is not true at all for this operating system. In fact Linux allows kernel's recompiling (1) according to hardware characteristics or user's specific needs. We can state that every single machine has got its own made to measure kernel which gives to it dynamism, and represents, in author's opinion, a valid answer to the proliferation of peripherals and new standards which make calculation processes heavy and force the static (not modifiable) operating systems to have got enormous machine codes. It's necessary to say something else about the concept of dynamism. In recompiling process it's possible to choose between a monolithic kernel and a modular one. The monolithic kernel is entirely loaded on the memory when the computer starts, the modular one, instead, accesses to modules, portions of kernel ready to be used, only when it's necessary. This gives fluency to the system with good results for hardware sources management.

The early versions were entirely developed by Linus Torvalds at the University of Helsinki in Finland, but Linux is so different because it was developed by many heterogeneous groups of Unix programmers and operating programs experts who put at disposal their products' codes for free. This heterogeneity refers to technical competence, kind of job and state of origin. In order that these programmers' communities could work together, they needed an efficient media. Internet was the media so, being Linux the operating system chosen by these people, instruments and utility programs necessary to use the web were developed early. Many application programs, besides being specially developed for Unix, were "imported" from the best operating system available on the market in that times. (2)

Linux's: characteristics: it's multitasking, multi-user, multi-platform. It has a characteristic called "shared copy-on-write pages among executables" or rather multiple processes are able to use the same memory to work.

When one of these tries to write on the shared memory, that page (4 KB of memory) is copied elsewhere, Copy-on-write has two advantages: better performance and reduction of memory use. It has a management of virtual memory which uses the pagination (i.e. it doesn't have to load all the process) on the disk, on a separate partition, on a file in the file system, or both, with the possibility to add in every moment, if it's necessary, other swap areas (without restarting the machine or stopping the process). 16 swap areas of 128 MB in total can be used in the same time for a theoretical available

industrial or professional fields are more able to satisfy end users by offering very good "niche" products.

FEASIBILITY STUDY: in this phase the project's times and costs are analysed. If a product requires to be developed over a very long period of time, or costs a lot of money, it will be automatically a commercial failure because it will be out of date by the time it is available for the market. This phase is often called "make or buy", because sometimes it's better to buy what will be too expensive to make. Of course it's necessary to verify the rate of correspondence between user's needs and specifications of the product for sale on the market.

REQUISITES' ANALYSIS: it's formalized by flow diagrams, or entity-relation schemes to describe step by step what the application should do. In this phase it's not necessary to plan how it would be done in reality, so as not to prejudice the final goal. This has to be done only as the document known as "requirement's specification".

PLANNING: in the software's planning phase, it has to be seen how it possible to satisfy the requirements. Without describing technical details we can say that a series of software modules are defined, making a list of functions of each one. The whole of these modules and their relationship makes the application's architecture. The more the modules are independent of each other, the simpler it will be to modify the software in the future.

IMPROVING: the code's weight is made. If the previous phases were correct there should be little to no modifications. This phase will ensure that programming time is not lost time. This will reduce the overall cost of programming, and the software can be easily modified.

MANTEINANCE: it consists of all the operations made after the software's delivery and installation. The request for some modifications is a function of a specific use of the application and so it will have variable costs and times. Often maintenance costs are 70% of the total costs of the application.

CONCLUSIONS

The choice of software for the office, cephalometric analysis, acquirement of images, or for other uses has to be secondary to the 9 characteristics previously described. It has to be strictly related to the operating needs of the single professional for example, choosing an operating system that is designed more for graphics or for networking. The total cost/benefit ratio must also be considered. Last but not least, the future direction of computers and technology in the next few years must be considered as they relate to the orthodontist's investment and technical knowledge.

References:

1. Cavaliere F. La compilazione del kernel. Inter-punto-net 1999; 48:73-78.
2. Floria G, Vergari A, Xenakis D. The first on-line orthodontic journal: an international experience. In: Atti del II World Congress on Biomedical Communications; Academic Medical Center University of Amsterdam The Netherlands; AMC 1999. P.51.
3. Parsons J., Oja D. Computer concepts. International Thomson Publishing company; USA 1996.
4. Rubini A, Zanetti G. Il sistema operativo Linux. Login: Building the information higway 1996;1:46-52.
5. Vandoni L. Ingegneria del software. Dev Infomedia1998; 57:81-83

space of 2 GB.
It uses a unified memory pool for software and cache memory in order to use all the free memory as a cache, and at the same time the cache can be reduced in case you have to run a big program.
It has full availability of the sources, included kernel and all drivers, system programs and all user programs.
Besides it can be distributed for free (4). It has sections and multiple virtual consoles: it's possible to have different, independent logins from the same station and to shift from one to another with a simple button combination (often ALT-F1 - ALT-F2...). It's possible to have as many as 64 parallel sessions, independent and contemporaneous. It's multi-platform for many file systems: minix-1, XENIX System V, MS-DOS, HPFS (OS/2 2.x), VFAT (Windows 95/98), NT, HFS (Apple Mac), FFS (Amiga), CD-ROM, NFS...
Its resident file system supports partitions up to 4 TB and file names up to 256 characters. It has a resident networked TCP/IP (including ftp, telnet, NFS, etc.)
It's able to work as a server for AppleTalk, as a server for Microsoft nets (emulating LAN Manager, NT...) and as a client (WfWg, W95, and NT).
It's able to work as a client or as a server in a Novell Netware net. Linux is based on Posix standard for operating systems, which was originally taken from the functions of Unix world.
It has to be noticed that Unix is Linux compatible for queries, i.e. many programs written for other Unix's versions or for Linux can be compiled and can work even on other systems, with few or no modifications. As a rule Linux run quickly than other Unix versions on the same hardware.
Linux architecture is already the second most common platform as web server in U.S.A., second to Solaris only (both use Apache Web Server). All the systems based on Unix's architecture (also Linux) are free from viruses because even if it's possible to compile them (the concept of virus started on Unix) the grant management on single files doesn't allow, if the net has a good structure, the telnet, and so if the worst comes to the worst it could destroy data of the single imprudent or unlucky user.
It could be interesting to speak about the concept of "free software" that is behind this operating system and many other good quality software, but that could bring to philosophical, economical and political considerations that will require too much space and a different collocation.

To cite this article please write:

Floria G. Evaluation of computer software in an orthodontic office. Virtual Journal of Orthodontics [serial online] 2000 Dec 15; 3(3):[5 screens] Available from URL: http://www.vjo.it/033/compen.htm