DOI:10.1145/2063176.2063192

Kai A. Olsen and Alessio Malizia

Viewpoint

Interfaces for the Ordinary User: Can We Hide Too Much?

Increasing the visibility and access to underlying file structure on consumer devices can vastly improve the user experience.

There is a tendency today, in the operating systems for PCs, tablets, and cellphones, to hide the underlying file structure from users. The idea is to simplify by hiding "technical details." We challenge this view, show there are situations where the user needs to work on the file level, and offer arguments that controlling the folder structure will simplify many operations.

Modern computer-based devices, including PCs, phones, and tablets, are designed for wide markets. Since these devices are acquired to make our lives simpler and perhaps more interesting, we do not expect to spend a lot of time learning how to use them. Interfaces should therefore be intuitive. The techniques applied to attain this goal go in two opposite directions: hiding and visualizing. By hiding technical details we can reduce the number of objects and processes an ordinary user needs to know. With visualization we can represent the more important objects in such a way that these can be handled by the user. A car is a good example. In earlier generations of automobiles, a driver would have to open the hood to pump gas, check the oil, and perhaps clean the gas filter and carburetor. Today, with automation, drivers do not need to see the engine at all. Technical details are hidden under the hood, while lamps on the dashboard give a warning if there is a problem. That is, the panel visualizes what we need to know in order to drive the car.

The complexity, extendibility, and flexibility of modern electronic devices make it impossible to hide files and folders.

Manufacturers of computer devices, operating systems, and application software have used these principles of hiding and visualizing for decades. Low-level technical parts, ASCII codes, disk track and sector numbers, program code, and system files are hidden. At the same time care has been taken to visualize more high-level objects such as programs, files, and folders. Based on ideas from Xerox PARC in the late 1970s, the user is allowed to start a program by a click on an icon or to copy a file by "dragging" the file icon from one folder to the other, that is, to operate directly on the visual representations of the underlying objects. The aim is not only to make intuitive interfaces but also to empower the user. With control of the data users can apply their devices to all sorts of tasks.

Now an explorer program—a file manager—becomes crucial. With this the user is offered an overview of all data resources on the PC, often with folders visualized as a tree structure. However, on modern devices, from cellphones to tablets, we see a tendency to hide files and folders. The idea is perhaps to make things simpler by taking hiding a step further, using application-dependent file locations. For example, when the user takes a picture the operating system of the cellphone will store this in a default location. Most users will not be aware of where. When the user wants to view pictures this is done through a photo viewer that shows the contents of the hidden folders. Some devices even come without an explorer program. Where folder and file names are created by the system, an explorer view will be of little or no use anyway.

We argue that this development is counterproductive—that the complexity, extendibility, and flexibility of modern electronic devices make it impossible to hide files and folders. As we shall see, there are many situations were an explorer program is necessary to offer the user full control over the device.

When an Overview Is Needed

The user-oriented interfaces we build on top of the underlying hardware and software break down whenever a device error occurs. It may be an unreadable disk, a corrupted USB device, a change in the underlying file structure, or run-

ning out of disk space. That is, the command, "open document" may result in a "file not found" message. Since we cannot build hardware and software that never fails, it is better to require a basic understanding of important concepts up front than confusing the user whenever an error occurs. Knowledge of the basic technology is also required to comprehend why a backup is needed. Even when backup is automated, for example over a wireless network, the user would still need to understand that no backup will be performed if the network is not available.

A user may have several devices, such as a phone, tablet, camera, and a USB flash storage device. At some time in the future she may also buy a new type of device X. All can be connected to a PC. The traditional strategy has been to present these devices as generic disks, with similar folder structures as for internal devices. The user can then view this structure through the explorer program, and may use standard desktop operations to move or copy files between the devices. This will also work for device type X. We see that the advantage of letting a user work on low-level objects such as files and folders is that on this level one may use a large set of "generic" functions, such as copying, moving, and deleting. Copying an image is of course similar to copying a document or a spreadsheet or any other data type. By viewing external devices as generic disks instead of cameras, phones, or music players we can perform similar operations on these, for example moving files, without learning new specialized operations. Of course, any new type of device, a GPS navigator, an eBook reader, or the aforementioned device X, can be handled in the same way.

In order to simplify and hide details the user's aims must be predictable.

ILLUSTRATION BY HARRY CAMPBELL

The alternative, that many modern operating systems and drivers apply, is to let an application pertinent to the device type take care of the connection. For example, when a user connects a digital camera to a PC running Windows, the following alternatives are presented as defaults: view, edit, and print. But most users would want to copy the pictures to a folder on the PC at this time. On the iPad, Apple uses synchronization as the default mechanism. When an iPad is connected to a PC the software will automatically synchronize data such as photos or documents between the two devices. But synchronization is a powerful mechanism that can have dramatic effects. The user may not want to transfer all the images from the PC to the tablet; there may not even be storage space on the tablet for everything. Synchronization may therefore easily result in an "out of memory" error, leaving the average user in a situation that she cannot recover from. If the user has more than one PC, for example an office and a home computer, she may get into real trouble. At home all photos from the home computer will be synchronized with the iPad. But if she later connects the iPad to her office computer, for example to synchronize documents, all pictures on the iPad that are not on the office PC will be deleted.

In order to simplify and hide details the user's aims must be predictable. This will be the case for a car, but not for a flexible computer system where users can download new applications, modify software, add apps, and perform other tasks. Then

it becomes important that the user controls the data objects. For example, the owner of a small bakery that offers home delivery may get orders by SMS. The default sorting of messages that his phone offers, with the most recent message at the top, is not what he needs. Instead he may want to have messages ordered on delivery date, or by address. He could have developed an app to move the data to an order entry system, but this will also require access to the data objects. The manufacturers of the cellphone would probably never have envisioned such a use of their device. However, with an overview of the data—in this case, the SMS messages—our baker can use the device for his special purpose.

Even the ordinary user may get into problems when the data structure is hidden. Most cellphones offer one folder only for storing messages. This works fine with a few messages, but navigation becomes difficult when these run into high numbers. Users will encounter problems when they employ their phone as a camera. With just a few images sequential storing works fine, but when the number of images runs into the hundreds or thousands user-controlled structures are required.

Conclusion

A user can benefit when her PC has noted the calendar event she entered on the cellphone, when the phone can provide her with a weather forecast for the current location at any time, or when she can "walk" down and view a street scene using Google. Many of these application-oriented tasks can be automated and the data hidden with little cost to the average user. In other cases we must take the "magic" away. Simplification is important, but should not be taken to the point where we make users less effective. Jakob Nielsen tells us that "Using the system changes the users, and as they change they will use the system in new ways."[1] That is, users evolve with the system. In fact we can talk about a coevolution of users and systems. Systems may be used in ways that not even the designers dreamed of. Thus it can be a good investment for a computer user to learn some fundamentals about the technology. Giving the user access to data through an explorer program will result in much more powerful user capabilities, while hiding the data may stymie innovation (see Santini[2]).

Simplification is important, but should not be taken to the point where we make users less effective.

We can ask why companies that have excelled in usability try to hide files and folders even when the arguments for visibility are so compelling. The professional answer may be they expect that most customers will only use the devices in the most straightforward ways (as planned, restraining the need for flexibility); that they have limited amounts of data (so that memory or disk overflow do not occur); that tablets and phones are connected to one PC only (to avoid synchronization problems); that backup routines can be performed by automatic systems and that users will contact service personnel whenever an error occurs. But there may be another reason. With access to the underlying data structure users can exploit standard formats for images, documents, and for all other data types. It does not matter which camera was used to capture images or on which computer system the images are stored. Users are free to move the data to their next PC or cellphone. If files are hidden, however, they can be stored in proprietary formats, thus making it more convenient for the user to utilize devices from one manufacturer only. C

References

1. Nielsen, J. *Usability Engineering*. Academic Press, San Diego, CA, 1993.
2. Santini, S. Is your phone killing the Internet? *IEEE Computer* (Dec. 2010).

Kai A. Olsen (kai.olsen@himolde.no) is a professor in informatics at Molde College and at the University of Bergen, Norway.

Alessio Malizia (alessio.malizia@gmail.com) is an associate professor in the computer science department at the Universidad Carlos III de Madrid, Spain.