

Paper Summary 3: Flat Volume Control: Improving Usability by Hiding the Volume Control Hierarchy in the User Interface

Adam Mikeal

7 March 2005

CPSC 671-600

- Title:** Flat Volume Control: Improving Usability by Hiding the Volume Control Hierarchy in the User Interface
- Source:** Proceedings of the 2004 conference on human factors in computing systems
Vienna, Austria
April 24—29, 2004
Year of Publication: 2004
ISBN: 1-58113-702-8
- Authors:** Patrick Baudish, Microsoft Research, Redmond, WA
John Pruitt, Microsoft MSX, Redmond, WA
Steve Ball, Microsoft eHome, Redmond, WA
- Sponsors:** SIGCHI: ACM Special Interest Group on Computer-Human Interaction
SIGWEB: ACM Special Interest Group on Hypertext, Hypermedia, and Web
ACM: Association for Computing Machinery
SIGCAPH: ACM Special Interest Group on Computers and the Physically Handicapped
SIGGRAPH: ACM Special Interest Group on Computer Graphics and Interactive Techniques
SIGGROUP: ACM Special Interest Group on Supporting Group Work
- Publisher:** ACM Press, New York, NY, USA

In this paper by authors at Microsoft Research, we are given the following scenario: a user is attempting to play a video in a media player during a presentation (not too uncommon a task, as recent experience in class will attest). Upon clicking the “Play” button, however, there is video but no audio. The user then turns up the volume on the media player. No change. Undaunted, the user browses over to the global “system” volume slider usually present in the Windows taskbar. That is raised all the way up—still no change. Next the user makes sure that the “Mute” checkbox on both the media player and the global system volume slider is unchecked; it is. Next stop is the “Volume Control” panel available from the system settings page (the Control Panel in Windows). The “Wave” slider is somewhat low, and that is adjusted up in an attempt to obtain sound—still nothing. Finally, the user notices that the “Mute” checkbox is selected, muting all output on this channel. The user unchecks this box, and is deafened by the cacophony that ensues, as he had turned up all the sliders to their maximum positions during the troubleshooting process.

This fictional scenario is unfortunately all too realistic, and is largely due to an overly complicated mapping between the volume control interface presented to the user and the underlying sound card. According to the authors, the system is unnecessarily presenting the low-level controls of the hardware, when a more appropriate interface from a user’s perspective would simply allow them to adjust the *loudness* of a particular application, not necessarily its *volume*.

My initial reaction to reading this scenario was predictable to those who know me: “Get a Mac”. However, the problem that the authors are trying to address exists also on modern Macintoshes running OS X, albeit to a much lesser degree. While the Mac long ago simplified the volume interface by abstracting (and hiding) the sound card channels, there is still a “master” system volume level that interacts with, and affects the individual volume settings on any particular application. If the “master” volume is set to a low level, the individual app becomes impossible to adjust to the desired volume without affecting the system volume first, and in the process perhaps altering the volume of other applications to an undesirable level (although Apple’s inclusion in it’s interface of hardware-based volume and mute controls also greatly aids in the entire process, and perhaps makes the following improvements only negligible).

The authors’ proposed solution is to “flatten” the hierarchy of controls through which a user must traverse to adjust the loudness of a particular application, and make affecting this property a single, direct action. They describe a sample interface they designed which contains sliders for all applications now playing and recently playing sounds. Each slider is directly mapped to the associated application; moving the slider will affect how loud the application sounds to the user. While the concept of a “master” system volume is still present, the direct control of this property is hidden from the user, and the volume interface is responsible for calculating how high this property should be set at any given moment based on the currently playing applications, and their individual loudness levels.

The redesigned interface includes a “Mute All” button, which when clicked instantly moves all the slider down to zero (the lowest level). If clicked again, the sliders will return to their previous positions. The designers deliberately chose not to represent this action with a checkbox, because unlike the mute features of existing systems, this button doesn’t describe a binary state that can interact *with* a volume slider, but more like an scripted set of actions: “drag all the sliders down to their lowest settings”.

Also included is a “master” volume control, but again it is implemented in a dramatically different manner. In the redesigned interface, this control appears not as another slider (which the user might erroneously assume to interact with the other, application-level sliders), but as a thumbwheel labeled “All Applications”. Moving the thumbwheel to the right or left *simultaneously* moves all the application-level sliders up or down, while maintaining their relative distance with each other. Because the human perception of *loudness* doesn’t map exactly to the concrete measurements of amplitude or decibel, the interface uses an algorithm to *proportionally* adjust the sliders to maintain their relative loudness. When at least one of the active sliders has maxed out, then the “master” control has maxed out as well.

Other features were also experimented with, but abandoned for the current prototype for the sake of simplicity, such as “flood marks” to indicate a level beyond which raising the current application’s volume would also raise other application’s volumes.

The authors performed a small-scale user study with seven participants to gauge the degree to which this redesigned interface aided in everyday tasks, showing a significant difference in the time it took the users to complete the tasks (up to 27%). Additionally,

five of the seven users indicated a preference for the redesigned interface over the original, system default.

One area that was lacking in the discussion was the impact that hardware-based controls (such as those available on Macs) would have had in the operation and completion time of the same tasks. The discussion of the interface problems was entirely centered around software-based controls, and showed strong evidence of constraints in their thinking by a single operating system (not entirely surprising or blameworthy, I suppose, since the author all work for Microsoft). Nevertheless, in terms of the current volume interface available in the Windows operating system, this proposal certainly addresses many shortcomings and would be a welcome improvement.

REFERENCES

1. Baudish, P., Pruitt, J., and Ball, S Flat Volume Control: Improving Usability by Hiding the Volume Control Hierarchy in the User Interface. In *Proceedings of CHI 2004*, 255—262. ACM, 2004.