ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THSH101

USING CONTROL SURFACES TO OPERATE CS-STUDIO OPIs

C. Rosati*, European Spallation Source ERIC, Lund, Sweden

Abstract

Modern control software has given us virtually unlimited possibilities for monitoring and controlling EPICS [1] systems, but sacrifices the organic feel of faders and knobs at our fingertips. This article will show how to reclaim that experience without losing the power of software through control surfaces commonly used with DAWs (Digital Audio Workstations) to manipulate audio, demonstrating how real motorised touch-sensitive faders, buttons and assignable V-pots will improve and speed up the control experience.

INTRODUCTION

There are interesting similarities between accelerator facilities and audio/video recording studios control rooms. Besides the high costs, from single cables and devices up to the buildings themselves, and the highly specialized technicians required to run the activities, both require a control room to operate their machines.

As can be seen in Fig. 1, physically big knobs, faders, buttons, indicator lamps, and meters were the tools at the operator's disposal to control the facility in the early times.

The computer era brought substantial changes in control rooms (see Fig. 2), substituting meters and indicator lamps with computer monitors, and buttons, knobs and faders with mouse and keyboard. Previous big cabinets plenty of physical controls are now substituted by screens[1] to be operated by means of computer mouse and keyboard (see Fig. 3).

In the audio/video production studios, recording consoles (in the early times, see Fig. 1b) were the command center of the recording studio. It was where all the audio was processed, sent out to dynamics and time-based effects units, headphone cues, the speakers, and final mixes were balanced and summed there as the audio passed from a tracking tape to a mixing tape. All of that now takes place within the computer in the digital audio workstation [11].

The new computer-based DAW's (Digital Audio Workstation) edit window is the old tape machine, and the mix window functions as the old console. While the edit window is great and offers endless improvements in terms of cutting and pasting, fading in/out, overdubs, rearranging, organizing, having playlists, etc., the mix window falls far short of an improvement over mixing with a console. It is incredibly small, relies on the mouse in order to make mixing moves and automation, confines the engineer to change only one control at a time, and forces him/her into a visual-based mixing experience [11]. So, while modern DAW software has given virtually unlimited possibilities for manipulating audio, it sacrifices the organic feel of faders and knobs at our fingertips [12].

This is where the benefits of a DAW *control surface* come to light. The control surface allows the engineer to ascend beyond the limitations of the mix window and get his/her head out of the computer and into the speakers. It relies on sound and touch, rather than numerical values on the fader. When you're limited to moving a little visual representation of a fader with the mouse, it gets between you and the sound. A control surface allows the digital domain of plug-ins, editing, and visual aspects to become secondary to the sound of the mix in itself [11]. Moreover, using a control surface rather than a mouse inevitably means performing a range of different arm movements, rather than the same small movements over and over again, which significantly reduces the risk of acquiring a repetitive strain injury (RSI) or carpal-tunnel syndrome. The potential health benefits alone, therefore, might be reason enough to invest in a control surface and its improved ergonomics [13].

In this paper I'll describe using a control surface to operate the Display Builder tool [14] of Control System Studio (CS-Studio, see [15]), the application widely used to control accelerators based on the EPICS [1] technology. This will add a further bit to the similarities between accelerator facilities and audio/video recording studios control rooms, borrowing something from the latter into the former.

While others already tried to use some control devices to improve the ergonomics of the control room (see, for example, [16]), the approach of directly control the PVs (Process Variables) fails in giving the user a proper feedback, causing him/her to make errors and, at the end, to be unsatisfied of the device and its integration. So, instead of directly operating the PVs, my approach is to control the *widgets* of Display Builder OPIs, leading to a real-time visual feedback much more satisfying for the operators.

MIDI CONTROLLERS

There are a lot of control surfaces nowadays available on the market [13, 17–19], from low-priced ones (see Fig. 4) to middle-priced (see Fig. 5), up to high and very high-priced (see Fig. 6). All of them communicate with the DAW application exchanging MIDI messages [20], and for this reason they are generally referred as MIDI Controllers [21].

Generally speaking, a MIDI controller is any hardware – or software – that generates and transmits Musical Instrument Digital Interface (MIDI, see [22]) data to electronic or digital MIDI-enabled devices, typically to trigger sounds and control parameters of an electronic music performance.

MIDI communication is achieved through multi-byte "messages" consisting of one *Status* byte followed by one

* claudio.rosati@esss.se

[1] In the audio recording industry, plug-in interfaces tend to closely mimic the hardware devices whose functionality they are replicating (skeuomorphic graphical user interface [7], see Fig. 3b). This tendency doesn't seem to be followed by the authors of OPI screens in accelerator facilities, where operator's screens are often poorly layed out, cluttered, and not professionally designed (see Fig. 3a, and [8]).

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THSH101

(a) Physicist Angelina Galtieri consults log with operator in control room of the Bevatron in 1963 [2, 3].

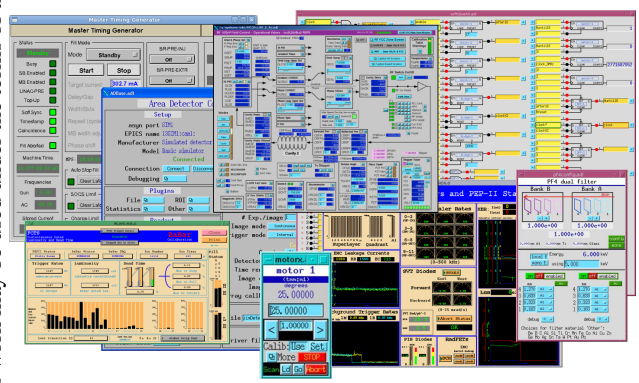
(b) George Martin working with REDD.51 mixing console at EMI's Studio Two, Abbey Road Studios, in 1964 [4–6].

Figure 1: Accelerator facilities and audio recording studios in the early times.

(a) Fermilab's main control room [9].

(b) Post production at Studio A, California Road Studios [10].

Figure 2: Modern control rooms.



(a) Accelerator OPI (OPerator Interfaces) screens.

(b) DAW (Digital Audio Workstation) plug-in screens.

Figure 3: User interfaces for control room operators.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THSH101

Figure 4: Examples of low-priced control surfaces: Behringer X-Touch Mini and Korg nanoKONTROL2 [18].

Figure 5: Examples of middle-priced control surfaces: Behringer X-TOUCH COMPACT and X-TOUCH [18].

or two *Data* bytes. *Real-Time* and *Exclusive* messages are exceptions. MIDI messages are sent over any of 16 channels which are used for a variety of performance information. There are five major types of MIDI messages: *Channel Voice*, *Channel Mode*, *System Common*, *System Real-Time* and *System Exclusive*. A MIDI *event* is transmitted as a "message" and consists of one or more bytes [22].

The Java™ programming language [23], the one used to build CS-Studio [15] and its tools (e.g. Display Builder [14]), support MIDI since version 1.3 through the `javax.sound.midi` package [24].

CONTROLLED KNOBS

A few months ago I reworked the wonderful little library that Gerrit Grunwald built to provide JavaFX [25] regulator controls [26]. I've published it on GitHub [27] and integrated inside Display Builder (see Fig. 7). This widget appeared to me to be the best one to be controlled by a rotary encoder device, like the DJ Techtools MIDI Fighter Twister [28], a low-priced MIDI controller available for less than 300€ (see Fig. 8).

By means of the MIDI Fighter Utility application (see Fig. 9), the device was programmed to:

Figure 6: Example of an high-priced control surface: Mackie Control Universal Pro [12, 17].

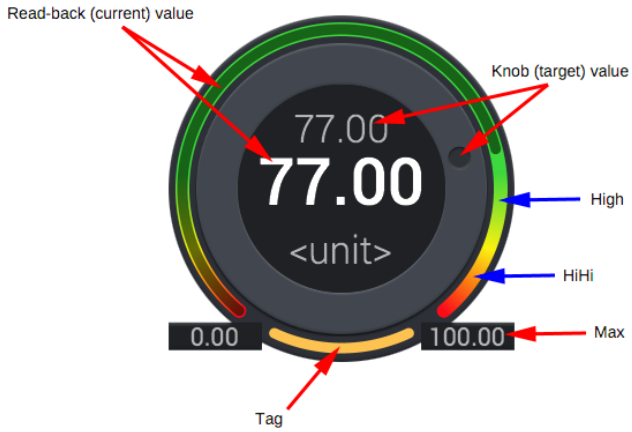


Figure 7: Knob widget in Display Builder [14, 27].

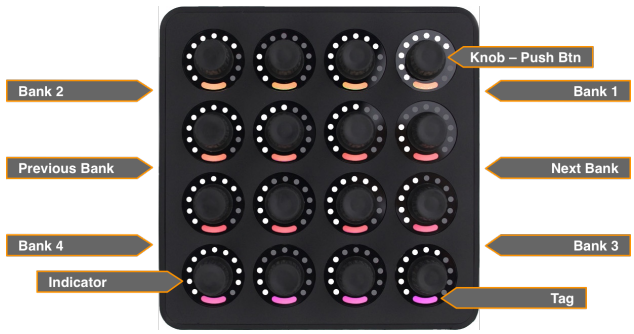


Figure 8: DJ Techtools MIDI Fighter Twister [28].

- use the 6 side buttons as depicted in Fig. 8;
- initially have the tag LED black (not set);
- have the 64 knobs (16 in 4 different banks) numbered from 0 to 63;
- have high resolution sensitivity for each knob;
- have the indicator blending the last LED according to the current value;
- send increment/decrement MIDI events each time a knob is rotated;
- send pressed/released MIDI events each time a knob is pushed.

The original KNOBS library [27] was extended to communicate with the MIDI Fighter Twister device, published on GitHub [29], and integrated into the ESS site-specific version of CS-Studio [30].

A *Controller* communicates with a physical control device and has a set of *Controllable* objects to control (see Fig. 11). Each time an event is received from the device, the Controller notifies the Controllable objects to update their GUI (Graphical User's Interface) representation (widget). Vice versa, when a Controllable object is graphically manipulated (usually through mouse and/or keyboard), it notifies back the Controller to update the device hardware (if

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THSH101

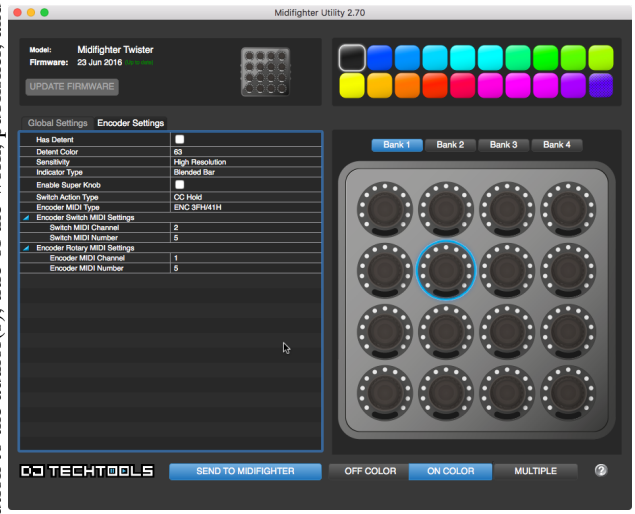


Figure 9: DJ Techtools MIDI Fighter Utility [28].

Figure 10: Controlled-Knob evaluator program and the connected MIDI Fighter Twister device.

and when possible). Controllers are discovered at run-time through the Java™ Service Provider Interface (SPI, see [31]) service discovery mechanism.

As can be seen from Fig. 11:

- *ControlledKnob* is just (i.e. extends) a *Knob*, implementing/exposing the methods required by the Controllable interface;
- other Display Builder's widgets could be easily controlled by the MIDI Fighter Twister device just implementing the Controllable interface;
- other MIDI controllers could be easily supported by implementing the few abstract methods inherited by the *AbstractMIDIController* class;
- Control devices not using the MIDI protocol (for example, the one used by Lang [16]) can be used too, by providing a Controller implementation that inherits from *AbstractController*, and communicate with the device hardware properly.

Three different *operating modes* are defined inside the Controllable interface to specify the interaction between the knob and its widget.

CONTINUOUS: Rotating the knob will coarsely update the widget target and current values. Pressing and rotating the knob will do the same in fine increments/decrements.

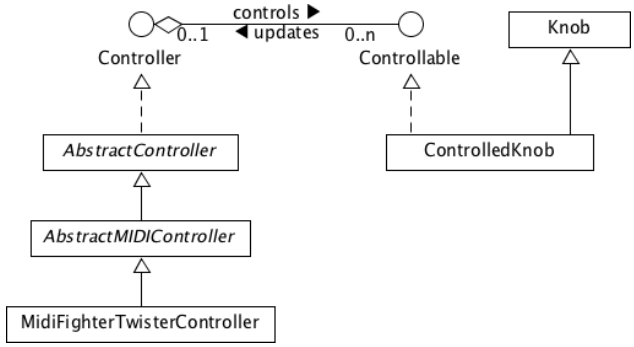


Figure 11: Controlled-KNOBS class diagram [29].

SET_AND_CLICK: Rotating the knob will coarsely change the widget target value. Clicking (pressing and suddenly releasing) the knob will force the update of the widget current value. Pressing and rotating the knob will do the same in fine increments/decrements. A further click will trigger the update of the widget current value.

CLICK_SET_AND_RELEASE: Rotating the knob will do nothing. Pressing and rotating the knob will coarsely change the widget target value. Releasing the knob will trigger the update of the widget current value.

The *ControlledKnobEvaluator* application provided with the library (see Fig. 10) allow to test the ControlledKnob implementation, allowing to:

- assign 4 widgets to different knobs in the MIDI Fighter Twister device (0, 1, 4, and 5 in Fig. 10);
- set the coarse and fine increment/decrement values for the selected widget;
- set the operating mode for the selected widget;
- select the controller from the list of discovered ones (currently only MIDI Fighter Twister is supported).

CONCLUSION

Despite the similarities between accelerator facilities and audio/video recording studio control rooms, in the former ones operators depend only on computer mouse and keyboard to operate the various OPIs controlling the machine, while a sound engineer can rely on the organic feel of faders, knobs, and buttons of a control surface to improve and speed up the control experience.

At the European Spallation Source I've experimented the adoption of a MIDI controller to enhance the user experience in operating Display Builder OPIs. The integrated MIDI Fighter Twister device can control in real-time up to 64 different knob widgets, providing multiple, mouse-free manipulations of corresponding PVs.

The preliminary tests show how incredibly easy, simple and natural is operating an OPI using the MIDI controller. More tests must be performed to get measured performances,

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THSH101

and to establish which is the best operating mode for the operators, nevertheless the initial results are encouraging.

Future developments will:

- improve the reliability of the library, allowing the user to safely connect/disconnect the control device without requiring the application to be restarted;
- increase the number of Display Builder widgets being controllable;
- support more MIDI controllers, especially in the low-priced range, like the Behringer X-Touch Mini;
- automatically use any discovered controller if the one originally used is not available;
- experiment with some very interesting alternative control devices, like Palette [32], VMeter [33], and QuNeo [34].

REFERENCES

[1] Experimental Physics and Industrial Control System. [Online]. Available: http://www.aps.anl.gov/epics/

[2] Bevatron. [Online]. Available: https://en.wikipedia.org/wiki/Bevatron

[3] Bevatron control room. [Online]. Available: https://www.flickr.com/photos/berkeleylab/4096230088

[4] A Brief History of The Studio As An Instrument: Part 1 – Early Reflections. [Online]. Available: https://www.ableton.com/en/blog/studio-as-an-instrument-part-1/

[5] A Brief History of The Studio As An Instrument: Part 2 – Tomorrow Never Knows. [Online]. Available: https://www.ableton.com/en/blog/studio-as-an-instrument-part-2/

[6] A Brief History of The Studio As An Instrument: Part 3 – Echoes From The Future. [Online]. Available: https://www.ableton.com/en/blog/studio-as-an-instrument-part-3/

[7] Skeuomorph. [Online]. Available: https://en.wikipedia.org/wiki/Skeuomorph

[8] C. Rosati, "Building Uncluttered OPIs," European Spallation Source ERIC, Lund, Sweden, Tech. Rep., 3 2017, internal course material.

[9] Fermilab's Main Control Room. [Online]. Available: https://www.flickr.com/photos/berkeleylab/4096230088

[10] Studio A. [Online]. Available: http://www.californiaroadstudios.com/studio-a

[11] V. Garay. DAW Control Surface – Pt I: Today's Studio Command Center. [Online]. Available: http://www.valgaray.com/daw-control-surface-pt-i-todays-studio-command-center/

[12] Mackie Control Universal Pro. [Online]. Available: http://mackie.com/products/mcu-pro-and-xt-pro

[13] DAW Control Surfaces – Roundup. [Online]. Available: https://www.soundonsound.com/reviews/daw-control-surfaces-roundup

[14] K. Kasemir, M. Grodowitz, A. Carpenter, and C. Rosati, "CS-Studio Display Builder Update," in *Spring 2017 EPICS Collaboration Meeting*. Research Reactor Institute, Kyoto University (KURRI), May 2017. [Online]. Available: http://www.rri.kyoto-u.ac.jp/EPICS/materials/DisplayBuilderUpdate_2017_05.pptx

[15] J. Hatje, M. Clausen, C. Gerke, M. Moeller, and H. Rickens, "CONTROL SYSTEM STUDIO (CSS)," in *ICALEPCS 2007*. Oak Ridge National Laboratory, October 2007. [Online]. Available: http://accelconf.web.cern.ch/AccelConf/ica07/PAPERS/MOPB03.PDF

[16] K. C. Lang, "USB Human Interface Devices and EPICS," in *Spring 2017 EPICS Collaboration Meeting*. Research Reactor Institute, Kyoto University (KURRI), May 2017. [Online]. Available: http://www.rri.kyoto-u.ac.jp/EPICS/materials/presentation.ppt

[17] The 7 Best Control Surfaces for Home Recording. [Online]. Available: https://ehomerecordingstudio.com/daw-control-surfaces/

[18] Best DAW Control Surface. [Online]. Available: https://protoolstutorial.org/best-daw-control-surface/

[19] Tweak's Guide to Control Surfaces. [Online]. Available: http://tweakheadz.com/control-surfaces/

[20] THE OFFICIAL MIDI SPECIFICATIONS. [Online]. Available: https://www.midi.org/specifications

[21] MIDI controller. [Online]. Available: https://en.wikipedia.org/wiki/MIDI_controller

[22] *The Complete MIDI 1.0 Detailed Specification*, 3rd ed. The MIDI Manufacturers Association, 2014.

[23] Java (programming language). [Online]. Available: https://en.wikipedia.org/wiki/Java_(programming_language)

[24] The Java™ Tutorials – Overview of the MIDI Package. [Online]. Available: http://www.midifighter.com/#twister-features

[25] H. Schildt, *Introducing JavaFX™ 8 Programming*, 1st ed. McGraw-Hill Education, 2015.

[26] G. Grunwald. Regulators. [Online]. Available: https://github.com/HanSolo/regulators

[27] C. Rosati. KNOBS. [Online]. Available: https://github.com/ESSICS/KNOBS

[28] MIDI FIGHTER TWISTER. [Online]. Available: http://www.midifighter.com/#twister-features

[29] C. Rosati. Controlled-KNOBS. [Online]. Available: https://github.com/ESSICS/Controlled-KNOBS

[30] CS-Studio ESS Product. [Online]. Available: https://github.com/ESSICS/org.csstudio.ess.product

[31] Introduction to the Service Provider Interfaces. [Online]. Available: https://docs.oracle.com/javase/tutorial/sound/SPI-intro.html

[32] PALETTE. [Online]. Available: https://palettegear.com

[33] VMeter. [Online]. Available: http://www.vmeter.net

[34] QuNeo. [Online]. Available: https://www.keithmcmillen.com/products/quneo/