The Cultural Potential of Keyboards

Neville Holmes, University of Tasmania

Culture has many facets and, with modern digital technology, many opportunities.

Human culture has been dependent on digital systems for a very long time, a digital system being one that “uses discrete (discontinuous) values” (tinyurl.com/WkPdDg).

Human language, the main enabler of our cultural development, has developed digitally. Spoken language is a combination of analog intonation and digital phonemes. Written language is notionally digital, and printed language is thoroughly so. Alphabetic language uses letters to indicate phonemes, punctuation to occasionally suggest intonation, and logograms such as & and 7 to represent words.

Musical culture also has a digital basis. For example, the keyboard was invented for a water organ more than 2,000 years ago (tinyurl.com/WpHdrls). Such keyboards are linear, suited to selection from a wide row of pipes or strings, but the keyboards developed much later for typewriting and typesetting were bidimensional because they were used for building narrow columns of words.

The typesetting keyboard arranged the letters of the alphabet in sequence by their frequency of use, so the first two rows, etaoin and shrdlu, became famous in the days of hot metal printing (tinyurl.com/WpEtSh). The typewriter keyboard was laid out rather peculiarly to avoid mechanical malfunction, hence the qwerty sequence (at least for the English language) and the staggered placement (tinyurl.com/WpKbLo).

In the days of 6-bit bytes with letters only uppercase, simple teletype keyboards were used with computers. When 8-bit bytes—EBCDIC and ASCII—were introduced during the 1960s, fairly straightforward typewriters were used as consoles and terminals.

Below, I include links to equipment I used myself, although quite a variety was available.

KEYPADS AND NUMBERS

Early PCs were expensive and often seen as glorified calculators. This was probably why many of them had a numeric keypad added to the alphabetic keyboard. An example was the IBM 5100 released in 1975 that had a very complex keyboard, including a numeric keypad, and a variety of control keys. It had a rather small screen but nonetheless was useful when lecturing because it had a video outlet for large-screen display.

The numeric keypad on the alphabetic keyboard was very popular because early commercial PCs were mainly used in offices, either for word processing or for spreadsheets and other calculations.

The keypad, usually of a 4 × 4 or similar design, with the rightmost column used to incite simple arithmetic, was much more suited to calculation than placing the row of numeric keys above the alphabetic keys, with its 0 strangely following the 9. Keypad use is still seen in shops with PCs coupled to cash drawers.

More widespread use of PCs and their descendants has reduced the need for simple calculation and increased the need for smaller size. Thus, the keypad is often superimposed over the right-hand side of the alphabetic keyboard with a (rarely used) NumLock key to bring it into play. Curiously, my cordless phone has gone the opposite way for compactness and lays the alphabet over a 4 × 3 numeric keypad.

Of course, the numeric subculture is unpopular today, but its keypad is a model of simplicity. Numbers can be keyed in with minimal hand movement, and touch-keying is relatively easy to learn.

KEYBOARDS AND PROGRAMS

Before PCs, fairly straightforward typewriters were coupled to comput-
ers, both as consoles used to interact with the operating system, and as terminals used to interact with user programs (tinyurl.com/Wp2741). A little later, a display screen replaced the printer, but the simple typewriter keyboard was still used (tinyurl.com/Chu2260). Interaction was by keying in a command or response and using the carrier return key, nowadays usually labeled Enter, to signal the end of the command or response.

As computers became more complex, in particular in allowing several programs to run at the same time, so did their keyboards. The old laptop PC I'm writing this essay on has 37 text keys and six text control keys (such as Backspace and Shift) that are descended directly from the typewriter keyboard. But there are also 33 other control keys that have their effect on or through the operating system. Most of these other control keys have alternative markings that are a mystery to me.

The idea of a control or function key made sense with a PC, as the user doubles as a system operator. This is probably why a distinct pad with 10 general-purpose control keys to the left of the alphabetic keyboard was introduced with the IBM 5150, as the IBM PC was known at first (tinyurl.com/WpPcKb). These keys were labeled F1 to F10 and had no fixed purpose. Certain conventions were adopted, however, and their general acceptance was followed by their expansion to F12 and their repositioning to a row above the numeric keys on the main part of the keyboard.

Since the function keys were almost always used independently of the numeric keypad, it's a little hard to understand why they weren't overlaid on the numeric keypad with a FnLk key to activate them. There, they would have been easier to use, and there could have been up to 16 on a 4 × 4 numeric keypad.

However, there's a drawback to the function keys: they're arbitrary— their numbering doesn't suggest their significance. In any case, there's a bewildering variety of other control keys and indeed of keyboards generally (tinyurl.com/CmpKbd). What's needed is a standard for keyboards with emphasis on symbolic control keys so that software will work better across cultures.

There's a hint of the possible in the arrow keys that are nowadays found in a ↓ shaped arrangement usually tucked in near the bottom right of the keyboard. These are cross-cultural ideographs that let the user select a direction of movement: up ↑, down ↓, right →, left ←.

That there are four of these arrow keys suggests that they could be laid over rows of the numeric keypad. With chording, this would allow 15 different functions per row. For example, ↑ could select expansion, and →← could select compression. The keypad's four rows could signal different scales or strengths, the higher the stronger, thus giving a total of 60 different functions without using the Ctrl, Alt, or Shift modifier keys. For the down arrow when used to move the cursor in a text file, the four strengths might move the cursor to the next line, the next paragraph, the bottom of the window, and the end of the file, respectively.

The arrows of different strengths would most simply and significantly be shown with progressive feathering, as Figure 1 shows. The Unicode arrow symbols are bewildering in their variety, but feathering seems lacking, and the feathered arrows would need to be added to Unicode for use in documentation (tinyurl.com/WpArSm).

The typical PC user switches between interacting with an individual program, such as a browser in a window, and interacting with the operating system. Therefore, if a SysLk and a PrLk key were provided, presumably in a row of Lk keys above the 4 × 4 keypad proper, 60 functions would be independently available for interacting with the operating system and with the current program before modifier keys would need to be used. This assumes that inter-row chording isn't supported on the grounds of its awkwardness and complexity.

The most important aspect of such an interactive keypad is that it's cross-cultural. The ideographic symbols could be used on a keypad alongside the text keyboard for any writing system, alphabetic or other, and their use could be consistent and memorable across languages.

The allocation of actions to signals would need to be carefully designed and standards agreed upon, but each arrow is capable of various implications. For instance, up and out, down and in, right and forward, and left and back could be accepted couplings. Agreement on a standard would facilitate use of software across cultures, and would simplify and partially unify the design of keyboards. It might also bring some consistency to the multiplicity of interactive devices currently available, and some order to the use of keyboards in cooperation with devices like the mouse and touch screen.

**KEYBOARDS AND TEXT**

Popular digital technology emphasizes rapid interaction with programs like videogames and browsers. Nevertheless, many people still use it for writing documents of various kinds. This is an important use culturally.
Unfortunately, the digital technology that enables people to interact recreationally with programs has been discouraging them from culturally interacting with text. Of course, text messages over mobile phones and the Internet are flourishing, but this is more superficial and social than cultural.

The problem is the restriction by technology of natural expressiveness. For example, the traditional input methods for Chinese characters involve keying in the pronunciation of a character using an alphabet, then selecting the required character from a list of homophones. However, the typical Chinese character has two parts: a radical that suggests the meaning and a phonetic that suggests the pronunciation. A creative writer might wish to combine any radical with any phonetic, but this isn’t possible when the user must select from a list of accepted characters.

For writers of English wishing to be innovative, there’s no problem if it’s simply a matter of putting together a word or name from the letters available on the keyboard, though a lurking spell-checker might try to interfere. But there’s a problem with traditional representations such as résumé, façade, zoology, N², or CO₂. I can do it in Word, but it isn’t easy or obvious for naive users like me. And I can’t think how I might put a right dipole (>) over the t in accenture.

There are several reasons for wanting to be able to easily modify English text typographically.

With English becoming more widely used, the use of diacritics and other modifications that make its pronunciation more evident has obvious merit. Being able to better show the pronunciation of foreign names, such as Phœbe, Zhou Enlái, and La Coruña, has merit as well. There’s also great potential for the teaching of English (The Profession, June 2006, pp. 102-104).

But the most compelling reason is simply to release imagination and creativity. This reason is becoming culturally more important with the threatened disappearance of handwriting (tinyurl.com/AgJ9w).

If children are to be taught to type rather than to write, they need to be able to type creatively, and not just to reduce boredom. The more expressively text can be used, the more appealing literacy becomes.

The obvious way to do this is to lay typographical functions over the numeric keypad with a TyLk key above it to enable typographic control while keying in text. The arrow symbols could be well suited for this, but they would need to be used in conjunction with the plaintext keyboard. Much could be done—in particular, belated restoration of the very useful backspace/overtype capability of traditional typewriters that the computing profession did away with.

The possibilities go way beyond simply the 60 keypad functions available. For example, by holding down a key of the alphabetic keyboard, the keypad could be used to modify the letter being keyed in, making it italic and wider, and rotating and reversing it. By holding down a typographic key, a sequence of letters could be keyed in to combine them, for example, marking diacritically and overlaying or ligating (tinyurl.com/WpTyLg).

These examples are relevant to cultures that use the Latin alphabet. Cultures that use other writing systems would need different enhancements for rich text, but the SysLk and PrLk controlled interaction would be able to cross writing systems.

The point of all this is that digital technology could greatly enhance both the interactive and textual use of computers by developing and standardizing keyboard technology. The computing profession has a social responsibility to press for this to be done and to see that the software exploits the extra capability thus provided, especially for schools. C

Neville Holmes is an honorary research associate at the University of Tasmania’s School of Computing and Information Systems. Contact him at neville.holmes@utas.edu.au.

Selected CS articles and columns are available for free at http://ComputingNow.computer.org.