Keyboard layouts: lessons from the Me'phaa and Sochiapam Chinantec designs

Hugh Paterson III
SIL International and University of North Dakota[i]
Hugh.Paterson@sil.org

Successful writing systems today depend on electronic input methods which can be easily used for producing printed or electronic material. This paper explores keyboard design issues involved in designing two keyboards for two different established orthographies. Both orthographies are based on Latin scripts and cover a total of five minority languages in Mexico (four languages in the Me'phaa genus and Sochiapam Chinantec [cso]). The design issues considered are:

- Technical differences encountered across major computer operating systems (OS X and Windows)
- Computer culture issues like the keyboard layout of the national language
- Key stroke frequency of language specific segments
- Unicode/non-Unicode issues related to character composition

Designing a Unicode keyboard for data input allowed native speakers of Me'phaa to have a greater involvement in the data collected by feeding the documentation team typed data and texts in addition to providing oral data. Early adaption of digital input methods may prove to better meet the needs of both language community and researcher. By giving the language community a keyboard for their orthography the minority language speakers were given the opportunity to enter into, and use, their language in new technological media and the language domains associated with communicating in those mediums.

With the arrival of new technological mediums in the personal communications arena, it is important for linguists and language documenters to consider their effects on the societies of minority language users (Eisenlohr 2004: 21). By and large, new digital technologies have come to sit between conversational interlocutors. Sometimes, as with the advent of the cellphone, minority language users can bypass obstacles presented by longer standing methods of encoding language used in distance communication, methods like orthographical representation. Now with the event of the smartphone and the tablet, we see video conferencing where not only are reactions to the oral channel considered in the communication but also the visual channel. This is certainly a change from merely writing or typing which was the primary means of encoding language nearly 100 years ago, and certainly the more popular way of communicating just five years ago.

Access to modern digital personal communication technologies is not the novelty it was a decade ago and it appears that the pace of cultural technologization is increasing (Holton 2011: 373, 93-4). Financial considerations may prevent minority language writers or media producers from acquiring a $3,000 USD computer (though laptops are in the offices of teachers in remote mountain villages of Mexico). These are not generally the price class of devices making it into (formerly isolated and low-income) minority language communities. It is far more common for these minority language speakers to be using local computer centers, (smart)phones, netbooks and tablets. All of these devices still require some sort of keyboard layout, be it soft or hard. The concern of language planners and users is the language of use in these digital mediums.

Notwithstanding these great digital advances or their social acceptance rates, oral and oral-visual communication is not always fitting, nor is it always sufficient. Modes of communication like text messaging, e-mail, web-surfing, letter writing, certificate printing, and a host of others require text based communication. These constraints also apply in the minority language context. Prompting language development planers (Diki-Kidiri 2011: 231; France, Zhozhikov, Aleksandrov & Varlamov 2011: 251; Russia), educators (Galla 2009, Silva & Donaghy 2004; Hawai'i), governments (Bailey 2007; South Africa), and speakers (Bernard 1992; Mexico) alike to acknowledge the need for keyboarding solutions. Often these communities or institutions call upon organizations and companies such as SIL International (McLendon 2011: 98-9) and Tavultesoft (2013). Encoding a language via text is not only needed by minority language writers but also of interest to language documenters. Best practice calls for the capture of language events and use in oral and visual modes as primary data (Bird & Simons 2003: 574) as well as text based documentation and analysis of primary data (Seifart 2006: 286) such as transcriptions, annotations and translations (Himmelmann 1998: 162-3). It is often the linguists who are first aware of the need to keyboard minority language text and rise to the occasion (Harvey 2013). When keyboard layouts are designed and distributed by linguists they are not always centrally or transparently available to communities nor are they always designed with the intent for use beyond the immediate project. While the issue of textual encoding is of great concern to both language documenter and community, the challenge to the minority language writer/typist is often not the reading of orthographies, but on the production of literature in these orthographies. Some of these challenges are social, some revolve around user experience issues, yet others are technological.

- Socially the challenges may include the typical domains in which languages are used; the acceptance and usage of (digital) written literature within the community; and balancing design for a particular minority language with design requirements for a poly-lingual environment.
- User experience challenges include visual stimulus and feedback to the user; key positioning based on frequency of character occurrence; and the psychological ordering of keystrokes to produce intended characters.
- Technologically, the challenges often center around the underlying encoding processes and the ability of the Operating System to natively process the input from input devices.

Discussion of keyboard layout design is often missing (Eisenlohr 2004), though not completely absent (Holton 2011: 372) from the language documentation technology literature. Much more technology related ink is spilt discussing archiving platforms and formats, internet usage, and analysis assisting software tools. The challenges arising from difficult to type orthographies are more often acknowledged in language documentation literature which deals with orthography design (Csató & Nathan 2007, Guérin 2008: 57, Jany 2010, Lüpke 2011: 333-4, Seifart 2006: 285-6). Treatments of the challenges faced by minority language writers generally focus on the development of orthographies, including their social appeal and readability, and therefore leave relatively little in terms of guiding principles for designers of keyboard layouts. This is not to say that nothing has been written on implementing keyboarding solutions in minority languages (Hosken 2001), but only that it has not been well treated with regard to current technologies in the language documentation and orthography design discussions. This is not completely unexpected because human-computer interaction such as keyboarding is often treated and discussed as an sub-discipline of computer science or psychology (Krishna et al. 2005) rather than linguistics.

Within the field of human-computer interaction there have been some advances in the last decade with respect to the keyboard. For many years the keyboard as it was popularly conceived by general users, was a device with a fixed arrangement. With touchscreen technology, the keyboard is no longer bound to a fixed position - including key location within a layout and which characters can be displayed on the surface of the keys, or visually returned to the user as output. Since there is no hard keyboard in these devices there is no limitation to designing layouts with graphemes which are tied to the majority language (Hinkle, Lezcano &

Kalita 2010: 191). Even with this great amount of flexibility, issues of key location on the keyboard layout are yet to be solved for minority language orthographies and their users. The challenge with keyboards for minority languages on devices with virtual keyboards is that the vendor must support keyboard layouts or app developers must create keyboards on a per app bases. Again this is not a linguist's concern; this is the software designer and product manufacturer's concern. Yet it remains the community's concern.

The problem space of an appropriate keyboard layout for a given orthography comes to the minority language writer partially un-invited, at the whims of the larger global society and heavily impacted by manufacturers of keyboards and communicative devices. Therefore, the challenge of how to access characters on a keyboard is not truly a linguist's problem. It is a social approach challenge. Nobody has told the community that they have to use technology. It is something that they pull from the majority culture around them. Linguists are called upon to solve the problem because they are often the ones representing technical expertise and are trusted by the minority language community. Language documenters attempt to solve the problem because they want to enable the documentation of the language. More broadly speaking, the challenge is really in the hands of human interface interaction and human interface designers. The delivery of the solution is in the hands of those manufacturing and marketing digital devices. Nevertheless, some practical guidance is beneficial for those new to designing keyboard layouts.

As language documenters and linguists, when we build digital solutions, like keyboard layouts, we need to consider the lasting effect on the communities to whom we are providing these products. It is our ethical and professional obligation not just to seek out solutions but to seek out great solutions. In the manufacturing industry, manufactures are often held accountable for the effects of their products on the users of their products. When we offer our linguistic and technical expertise to communities of minority language speakers and writers, we need to not just design solutions; we need to offer well designed solutions. Just because we create something which is usable and useful does not mean we have created something desirable. And when the community does not want to use that created input method, our answer should not to simply say: “well they do not have enough desire”. The interesting thing about keyboard layouts is that they are not just products, they are also experiences. Each keystroke in its place is a pattern created in an attempt to implement the orthography. It creates an experience that writers’ fingers will potentially encounter multiple times a day. This physical interaction is only part of the user experience and should not be overlooked in the design process. Other parts of the user experience deal with the keyboard layout as software. So the keyboard layout should be considered and designed as software as well as an experience.

The words usability and design each suffer from a very unfortunate ambiguity. Usability in a very raw sense means, “is a tool usable?” Just because every tool can be used as a hammer, does not mean that every tool should be shaped like a hammer. Nor does it mean that every tool should be used as a hammer. Just because a keyboard layout can be used does not mean that it is a good layout. The term design in computing also suffers a similar fate to the term usability. If some computer tool does something, it does so because it was designed to do so. The software was not generated by accident. Mere existence does not mean that the computer tool is aesthetically pleasing or that it creates a sexy or desirable impression on its user (Anderson 2006, 2009, 2011a, b). As designers we strive to make our software create an impression on the user, so that the user wants to re-use the software again and again, all the while the software is actually meeting the functional needs of its user. It might even be said that we hope to create a symbiotic and addictive relationship between the user and the software.

There are several marks of good design. Dieter Rams has put forward ten principals of good design (Vitsœ 2012). Here I will relate keyboard layout considerations to four of them.

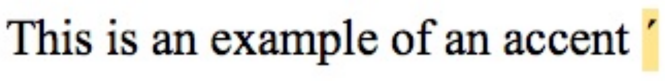
Good design:

- Makes a product useful
- Makes a product understandable
- Is unobtrusive
- Is thorough down to the last detail

Rams suggests that a product is not useful if it does not also meet certain criteria: aesthetically, functionally, and psychologically. Because we are talking about keyboard layouts, aesthetics are physically dictated by the physical keyboard, or the manufacturer of the OS on tablets. However, criteria for function and the psychological relationship with the keyboard layout are both available to keyboard layout designers (linguists and language documenters).

Some of the functional criteria are obvious, the keyboard layout must be able to implement the orthography of the target language. Another very useful thing for the keyboard layout to accomplish is to type in the majority language's orthography. In the cases of Me'phaa and Chinantec this would mean being able to type Spanish as well. It is important to notice the directionality of composition. Typing a document in Spanish and adding a few words or sentences in Me'phaa is drastically different from typing a Me'phaa document and adding a few words or sentences in Spanish. Even if the orthographies are "similar" in that they both use Latin scripts and try and show social affinity by "looking similar" there is still a difference in terms of the user experience when composing the document. For instance two semantically equivalent texts[ii] were analyzed for the purpose of evaluating the efficiency of the Me'phaa keyboard layout, the Me'phaa glyph ⟨ á ⟩ is used 880 times whereas in the same content written in Spanish the same glyph is only used 59 times. A keyboard which takes into account the complexity of inputing a complex glyph should also take into account the frequency that that glyph is accessed. To input 59 ⟨ á ⟩ glyphs in Spanish on the Spanish ISO keyboard requires one to make 118 keystrokes. Alternatively, on a standard OS X ANSI U.S. QWERTY keyboard, one would need to make 177 keystrokes to form the same 59 ⟨ á ⟩ glyphs. Using the Me'phaa layout we created it still only takes 118 keystrokes to produce the 59 ⟨ á ⟩ glyph in Spanish. However, if one were writing the same content in the Me'phaa language one would need to use 880 ⟨ á ⟩ glyphs, this would require 1,760 keystrokes. At this point the layout designer needs to ask: Is the Spanish layout an efficient option for typing Me'phaa? Concerning psychological factors, we need to consider how much work it is to produce each character and how that impacts a person's desire to type in a given language. Psychological factors also include user experience and the psychological composition of characters represented by complex characters like accented characters representing tone and stress. In terms of user experience, we must also consider placement of a frequently typed character on the keyboard layout. In the Me'phaa text the LATIN SMALL LETTER SALTILLO U+A78C is used 1,189 times. This accounts for almost 8% of all characters used in the text and is the second most common noncomplex character. This character on the Meꞌphaa layout is placed at one of the farthest places on the keyboard for the little finger to reach[iii]. This distance can have an effect on a typists speed and rhythm. The Saltillo is also a character which is not in the Mexican-Spanish orthography[iv]. The combination of these two factors makes it more compelling to type in Spanish rather than in Meꞌphaa. The ordering of input on the diacritic marks is another user experience factor pointed out by Hosken (2001 section 5.2). It turns out that this is significant for both the Meꞌphaa and and Chinantec layouts. Both use dead keys to assign diacritic marks to base characters. The important design question becomes: How should the ordering of the diacritics be conducted? Should the dead key (or what might be socially viewed by users as a "tone mark key") be struck first and then the base (known as the dead key method) or should the base be struck first and then the modifier key (known as the operator key method)? While both may be valid ways to consider input there are several issues related to user experience which need to be considered.

Hosken (2001) points out that using the dead key method does not provide the user with any visual feedback, whereas there is always a visual change for every keystroke with the operator key method. It would appear that this has be addressed in OS X by rendering the diacritic without a base prior to the base being struck. Additionally, if the diacritic is one which can become part of a composed character, it is backgrounded in yellow. However, as of


This is an example of an accent ʹ

Windows 7, the behavior of Windows is still to not show the diacritic before the key for the base is struck. Therefore on the Window’s OS, Hosken’s concern obtains.

In his design principles, Rams says that “good design makes a product understandable. [Good design] clarifies the product’s structure. Better still, it can make the product talk. At best, it is self-explanatory.” (Vitsœ 2012) In today’s lingo we might say that the product needs to be intuitive. For keyboards, this can apply in several ways.

It used to be the norm that most minority language writers were also new computer users (or had no exposure at all to computers). With globalization and the digitization of communication, there are many more endangered language speakers who use or have access to computers, tablets and cell phones. One challenge of using a virtual layout, and even more so for those who are new to typing, is the mismatch between what is printed on the keys of a physical keyboard and what the virtual keyboard layout returns as output to the screen. As designers, one thing to consider is the effect of visual feed back to the user. Both what the user sees on the screen and what they are looking for on the keyboard. While the pedagogy of typing generally calls for hiding the keys from the eyes to increase speed of typing, the universal benefit of this is questioned (Byers, Brovey & Zahner 2004). In the case of the minority language writers, there is at least two considerations for needing to see the correct characters on the keys:

- how typists learn what buttons to press to get specific results
- the only way to complete the visual spectrum feed back loop.

From a user experience perspective I know that if I touch said button that it means I will get said results from the keys on the screen. So the experience is about setting the users expectations and then meeting them. Foundational to this is the visual confirmation for that user. One practical and inexpensive way to approach this is to have a plastic overlay custom made, which can be done for under $10 USD. However, it is also important to note that visual feedback could also be confusing to users. In the Mephaa text there was situation where the intent of the typists was supposed to use Guillemets but the typist used only greater than and less than signs. So in that case the visual presence on the keyboard simulated a character association which was not the intended character. This kind of error is not generally encountered, or specifically acknowledged as a typing error in the relevant literature for typing (Kano 2008) (for overview see: Kano & Read 2009: 294). However, it could possibly be considered a "hardware" mistake as classified by (Kano et al. 2007) and (Read, MacFarlane & Casey 2001) as they follow Norman (1981) in classifying other kinds of mistakes made while typing.

Figure 1: Users rely on visual continuity between what they see printed on the surface of the keyboard and the output on the screen.

If this kind of error is common among typists in a given community, one approach designers can take is to make the angle bracket key when hit twice in a row output a Guillemet. However, this method is more divergent from the Spanish ISO keyboard (the national layout), and begs the question: What will a Me'phaa writer do when they are writing a Me'phaa text on a Spanish keyboard? To some challenges, the only good solution is education. An education where the writers using a particular keyboard understand the differences between two languages, their orthographies and capabilities of a given keyboard solution, but then also an education for the designer to be able to take the community feedback and consider possible alternatives.

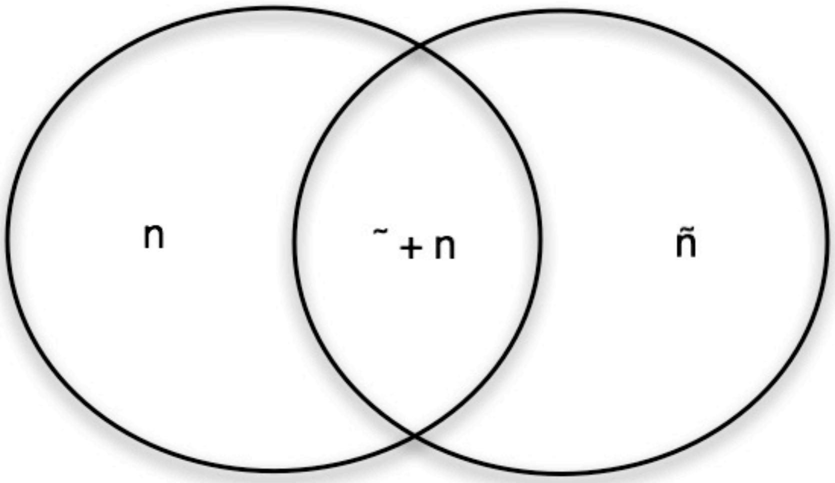
Another way to make layouts more intuitive is to design them to behave the same way across multiple operating systems. A typists should not need to relearn how to type on each new device. This provides continuity to users when they switch computers or operating systems and maximizes opportunities for social, peer based learning. Both continuity (the status of previously understood analogies in graphical user interface design) and learnability are important factors which affect the adoption of software. When language use in the digital medium is in question, adoption of software is essential to the success of language revitalization.

In implementing the Me'phaa keyboard layout one of the questions which was asked was: Could we use the vendor key as a dead key? Our hope was to avoid the sacrifice of one key which is primarily in the grapheme production area of keyboard to the sole purpose of becoming a dead key. The result was: “no, we could not use the vendor key”. The design motivation for this is to respect and comply with device and platform oriented user interface guidelines. It is typical for applications running on OS X to use the vendor key, also known as the command (⌘) key, as an application level shortcut key. This behavior is also not uncommon on Windows and Linux machines where the control key is used in leu of the vendor key. On Windows based machines, the vendor key is used to bring up the Windows menu. Therefore using the vendor key as a modifier key becomes problematic because it changes the overall way the machine behaves rather than just modifying the orthographic characters available to the typist. As designers of keyboarding experiences, the experiences need to transparently fit into the overall computing experience for any given platform.

A third way that a keyboard needs to explain itself is through the cognitive associations it invokes on its users. Should all characters be accessed the same way?[v] Not all orthographies use the same characters to represent the same sounds or ideas (phonetic or phonological representation as understood by the orthography users). Kutsch Lojenga (2011) offers an example from Yaka [axk] and Sango [sag] in the Central African Republic:

> Occasionally, different accents are used, e.g. when the circumflex is used for H tone, as is done in YAKA (Bantu C.10, spoken in C.A.R.), where the choice of tone marks had to conform to the system used in the widely-known lingua franca Sango, by using a circumflex for H tone. It may not be elegant for a linguist, but it works.

In Chinantec and Spanish the use of ⟨ ñ ⟩ is in a similar relationship as Lojenga describes above. In Chinantec ⟨ ñ ⟩ represents a velar nasal; whereas in Spanish it represents a palatal nasal. In Me'phaa the acute accent ⟨ ´ ⟩ represents tone, where as in Spanish and Chinantec it represents stress. However, in terms of the character composition, and tactile input of that character the question becomes: How is the ⟨ ˜ ⟩ diacritic related to the base ⟨ n ⟩ and does that relationship parallel the semantically salient ideas about the phonemes these glyphs represent?



How do speakers conceptualize the graphical element and relate it back to the sound it represents?

To put it another way, How do the speakers conceptualize the graphical elements of the glyph? Do indigenous writers think of ⟨ ñ ⟩ as a separate character from ⟨ n ⟩? or do they think of it as an altered ⟨ n ⟩? As layout developers, we must be asking are we dealing with two separate ideas or the modification of one idea?

One option in designing the Me'phaa and Chinantec layouts was to remove the ⟨ ñ ⟩ from having its own dedicated key and make the ⟨ ˜ ⟩ a diacritic which was then accessed through a dead key or even the same dead key other diacritics in the language were accessed through.

Unicode composite and base characters with consonants	
ñ	n + ˜
LATIN SMALL LETTER N WITH TILDE U+00F1	LATIN SMALL LETTER N U+006E + COMBINING TILDE U+0303

This would serve to free up a key in the layout for a more common character, and also serve to bring consistency to the input of characters with diacritics. In both cases it was decided to leave the ⟨ ñ ⟩ key as it appears on the Spanish ISO keyboard layout. But this example serves to point out that such considerations should be made on a language by language bases.

If a keyboard layout is to be intuitive to its users then there should be a parallel between the graphical representation of sounds and the way the glyphs are generated through the fingers. There should also be some internal cohesion regarding how composite characters are created by a given keyboard layout. This is part of designing the tactile aspect of the user experience element of a keyboard layout.

An example of internal cohesion can be seen in the Me'phaa layout in the way that tone is marked. Me'phaa has three levels of tone which are indicated in the orthography. High tones are marked with an acute accent above the vowel ⟨ ´ ⟩, mid tones are unmarked, and low tones are marked with a COMBINING MACRON BELOW U+0331 ⟨ _ ⟩. The use of the macon below gives the visual effect of an underline below the vowel. The Me'phaa keyboard layout dedicates one dead key for high tone and another dead key for low tone. By giving each tone mark its own dead key the keyboard layout creates a symmetry in user experience for how a tone can be marked on each vowel.

The symmetry in the Chinantec keyboard was not as simple to implement. This is due to some limitations in one of the operating systems the keyboard was being implemented on and the way that characters are coded in Unicode. Understanding the how Unicode allows for the target characters to be created will help us see where there is symmetry and where there is asymmetry. The Chinantec asymmetry is more clearly understood when it is compared with the Me'phaa case described below:

In Me'phaa the letter ⟨ a ⟩ can be used by itself, with a low tone mark or with a high tone mark. In every case that ⟨ a ⟩ is combined with a low tone mark two Unicode characters are needed: the base character ⟨ a ⟩ and the combining macron below diacritic ⟨ _ ⟩. However, when a high tone is used there are several ways these could be encoded: as ⟨ a ⟩ plus ⟨ ´ ⟩ or as a single character ⟨ á ⟩.

Unicode composite and base characters with vowels		
a	a + _	a + ´ or á
LATIN SMALL LETTER A U+0061	LATIN SMALL LETTER A U+0061 + COMBINING MACRON BELOW U+0331	LATIN SMALL LETTER A U+0061 + COMBINING ACUTE ACCENT U+0301
		LATIN SMALL LETTER A WITH ACUTE U+00E1

In the Me'phaa case the available options in Unicode do not make a difference for the implementation of a symmetrical input method. However, in the Chinantec layout there is an important difference. Chinantec, like Me'phaa is a tonal language. However, the orthography does not mark the tone on the vowel, but rather with numbers at the end of the syllable (Fortis 2000, Unknown 2009). It marks a type of stress on the vowels with an acute accent (Mugele 1982). Symmetry does not become a problem until we try and implement a stressed barred ɨ ⟨ ɨ́ ⟩. Unicode does not have a composite character for LATIN SMALL LETTER I WITH STROKE AND ACUTE. This means that the character needs to be a series of at least two Unicode code points and it could potentially be coded as three code points.

The conceptual construction of a Character		
ɨ	í + ̵	ı + ̵ + ́
LATIN SMALL LETTER I WITH STROKE U+0268	LATIN SMALL LETTER I WITH ACUTE U +00ED	LATIN SMALL LETTER DOTLESS I U+0131
	COMBINING SHORT STROKE OVERLAY U+0335	COMBINING SHORT STROKE OVERLAY U+0335
		COMBINING ACUTE ACCENT U+0301

The challenge comes because the keyboard layout editor from Microsoft for Windows, MSKLC, will not allow the building of keyboard layouts which provide the stringing of input with one keystroke. That is, if I want to input three or more characters with one keystroke, it can not be done.[vi] This behavior is desirable for using decomposed characters, i.e., if I have a base and two diacritics then I would have three unicode characters. Microsoft based keyboard solutions can only input one unicode character per keystroke and do not allow triple character input as is required by some orthographies (Holton 2011: 372) With OS X, a dead key can be used to enter another state of the keyboard, wherein, when the correct key is struck the desired series of Unicode characters is input. However, with MSKLC it is impossible to replicate the multi-character input behavior. Rather, the dead key must be used to insert the combining diacritic and then the next key is used to insert the base. In this manner all of the necessary diacritics for Me'phaa were addressed. However, for the Chinantec no solution was found for accented barred I ⟨ ɨ ⟩. This is because by nature this is a composed character. If we were to represent symmetry with the other characters in the orthography, we would have to move from barred i ⟨ ɨ ⟩, LATIN SMALL LETTER I WITH STROKE U+0268 to barred i with acute (accent). This is not possible with Unicode because the barred i would need to be dotless, rather than combining above the dot. Alternatively, one could add the diacritic COMBINING SHORT STROKE OVERLAY U+0335 to the base character ⟨ í ⟩ LATIN SMALL LETTER I WITH ACUTE U +00ED but this method on Windows would require a fourth dead key for the stroke overlay (a dead key already exits for acute, dieresis, and tone) and this dead key would not match the behavior of the layout for adding a stress mark to the other vowels, if implemented as a stressed ⟨ í ⟩ plus a stroke overlay. This implementation would also not fit the way that indigenous writers think about the vowel; as being barred i ⟨ ɨ ⟩ plus stress. Rather it forces writers to think about the glyph as stressed i ⟨ í ⟩ plus stroke overlay.

Rams' third point about design which is applicable to keyboard layouts is Good design is unobtrusive. The limitation of MSKLC to be able to produce keyboards which are flexible and meet the needs of writers has opened the door of opportunity for third party application developers to come up with creative solutions. These solutions add complexity to the user experience and also add complexity to the deployment of layout files. Up to this point the discussion has focused on using tools which create files which work and are installed within the framework of the OS not needing third party software. Some of these third party solutions include:

- Keyman (Durdin & Durdin 2011)
- Inkey (InKey 2012)
- AutoHotKey (Mallet 2012)
- KeyTweak (Krumsick 2009)
- Sharp Keys (Santossio 2011)
- Map Keyboard (InchWest 2012)

The first two of these solutions allow for the editing of custom keyboard layout files but require their software to be active and running on the computer to use the layouts. They are also created with the multilingual typist and minority language typist in mind. The third program listed above has wider audience and works via scripts and can be configured so that a script converts each keystroke to the desired character(s). The last three programs are essentially graphical interfaces on registry editors for the windows based keyboard registries. They are best viewed as a MSKLC alternatives with one exception: If a user edits a registry file then the changes are global, whereas if they create a keyboard the user can choose when to use a given keyboard layout (on a per program bases).

Each of the above applications require the installation of additional software and the particular keyboard desired for by a minority language typist, whereas a MSKLC based solution only requires the installation of the actual keyboard file via an .exe script. Therefore in terms of design, and creating a solution which can readily be adopted and used by a minority language community these above third party options represent non-optimal solutions.

As language documentation practitioners and language use advocates, when we introduce a solution to a community one of our considerations should be that solution’s longevity and sustainability. For software we need to ask what is the future capacity of the community to develop or modify this solution further? Does this software have any dependencies other than the OS which will depreciate the solution and disrupt communication for this particular community? Closed source, third party solutions, and third party solutions which have only one maintainer, have a greater chance for more rapidly becoming depreciated. Solutions with a significant financial return on time invested have a greater chance at not becoming depreciated.

Rams’ fourth point that Good design is thorough down to the last detail has several implications for language documentation partitioners and advocates for minority language use. As already mentioned there are elements of the solutions we bring to communities which are not specifically just one product, but rather part of a larger ecosystem of how that community will interact with acquiring or implementing that product. So, as we consider what we need to design in terms of layouts we also need to consider: Do we want the use of these digital products created through our efforts to spread through-out the language community? What is the impact we as catalysts in language vitality situations want to see? What is the level of complexity we want to bring to digital interactions in a given community’s language?

If as practitioners, who hope to improve the digital interface for minority language writers and to see our recommended (and collaboratively developed) digital solutions become embraced by communities, then we should pay attention to even the last details. There are details like:

- issues of congruency of character positioning in the keyboard layout with the national language keyboard layout;
- how is the character composed in the mind of the writer? - what is the relationship between a diacritic and its base? in the case of tone: Is it a toneme or is it a vowel with a tone? or is it a high tone vowel? is it a grammatical tone attached to the tone

bearing unit of the word? or is it a tonal melody (Snider 1999) super-imposed on a word or morpheme?;
- the actual composition of characters and the input of unicode code points of the data.

However, we must consider two final details. We should keep free and open source software (FOSS) principals in mind when we license and release our products; and we should have a stable, reliable and sustainable distribution mechanism. FOSS principals (Wong & Sayo 2004) allow for digital products to be legally accessed and shared freely throughout the community. They also leave a path for sustainability by allowing anyone to alter and redistribute the software.

FOSS principals:

- The freedom to run a program, for any purpose
- The freedom to study how a program works and adapt it to a person's needs
- The freedom to redistribute copies so that you can help your neighbor
- The freedom to improve a program and release your improvements to the public, so that the whole community benefits

By adding to the FOSS principles a reliable distribution point, perhaps ScriptSource (Raymond 2012), we build trust with the community. Trust is built as community members have a place to consistently obtain the keyboard layout product and a place to obtain a trusted copy of the source code. These need to be accessible to those in the local geographic area and to minority language writers in diaspora.

Good design is not designing a usable keyboard (physical). Good design is creating a keyboard layout which ergonomically, psychologically and transparently meets the needs of native speaker-writers.

In keyboard layout design, the goal is to make the orthography work with the technology. While some may view the process developing writing options for language communities as being one where both the layout and the orthography are variables. It is far easier to establish the orthography first and then develop technology to meet the demands of the orthography. Some ask, why not just use a majority language keyboard layout? Why not restrict the characters used by the minority language to those used by the majority? For the Chuxnabán Mixe [pxm] Jany (2010: 235-36) presents this very position.

> … [An] important non-linguistic factor in the development of an orthography for an oral language is ease of use with computers and new media. With the world-wide web reaching even the remotest areas of the world and expanding in use, it becomes clear that a new orthography should be designed in a way so that its graphemes are readily available on standard keyboards. This will not only facilitate the language documentation process, it will also encourage its use with new media and possibly in new domains.

What is perhaps unique about this Mixe example is that the keyboard is viewed as a potential repository for orthographic characters. This view is not universally accepted. In South Africa, rather than being subject to mechanical imperialism, translate.org.za helped to create the South African Keyboard (Bailey 2007). The designers chose to not be subjected to the confines of technology and used the opportunity to create a keyboard layout which addressed the specific needs of one language, Venda [ven] and still served the needs of multiple languages in South Africa. In this way they brought the characters needed by typists to keys under their fingertips.

In dealing with layout design, the task is the implementation of the orthography. It is not the creation of the orthography. That said, if orthography developers were approaching the task with the mindset to also create a keyboard layout then these are relevant questions. Relevant, not because they are designing keyboard layouts, but because they are designing orthographies. Addressing the larger, and more complex question about the orthography, and maximal differentiation; others have said that designing orthographies, is a task of matching the orthography to the social attitudes of the language speakers (Cahill & Karan 2008: 10). In many respects this is very much like corporate logo design or typeface design. As keyboard designers, we have to follow the orthography design not lead it. In many cases, as Lüpke (2011:316-317) describes about Baïnouk [bcz] the orthographies change overnight in order for the language to receive recognition status. As these standardizations take place, linguists watching the tug-of-war for social and political clout hope that everybody involved feels like they win something; even if it means re-printing language education materials. This has long-term implications on minority language groups establishing a culture of writing. If the orthography is changed every five to ten years, then old materials, and potentially skills, are redundant.

The advancement of technology changes the dynamics of communicative setting. And humans are more than willing to adopt these advancements - as long as they embrace the benefits. In a way, this is why we have the current language death crisis. Well designed keyboard layouts are not some technological silver bullet. By themselves they will not reverse the status of moribund languages (Dauenhauer & Dauenhauer 1998: 70, Holton 2011: 397). Well designed layouts do give the power to the community to use their language in the medium of choice in the 21st century. Just because the community has the ability to type in their language does not mean that they will. However, having the ability gives the community a choice. Nikolai Pavlov (2011: 241) says, "The problem [online Yakut use] is aggravated by the paucity of native speakers generally and the already mentioned fact that young people are used to socialising in cyberspace in the majority, functionally stronger, language." As part of a strategy for encouraging the use of Yakut Pavlov goes on to say that a keyboard layout is necessary.

Minority language writers will only be successful when there is desire, ability, and opportunity to function in a digital medium. Technology, is in and of itself, is not the savior of a language vitality situation. There still have to be speakers and users of the language. Even with a keyboard layout and the technology to support language in a digital medium, various social pressures persist in influencing minority language writers to not use their language.

There are a variety of factors which affect the input mechanism other than the users and their language. As we have seen, these include: the product manufacturers, industrial standards affecting keyboard layouts, the characters on the surface of the computer keys, and the keyboard layout of the majority language of the region. However, a chief concern for designers of keyboard layouts should be the users' experience and the way that human-computer interaction is approached for a given digital medium.

References

Anderson, Stephen P. 2006. Creating Pleasurable Interfaces: Getting From Tasks to Experiences. Paper presented at Refresh '06. <Accessed: 27 January 2012>. http://www.slideshare.net/stephenpa/creating-pleasurable-interfaces-getting-from-tasks-to-experiences

Anderson, Stephen P. 2009. Seductive Interactions. Paper presented at Idea '09. <Accessed: 27 January 2012>. http://www.slideshare.net/stephenpa/seductive-interactions-idea-09-version

Anderson, Stephen P. 2011a. Long After the Thrill: Sustaining Passionate Users. Paper presented at #IxD11. <Accessed: 27 January 2012>. http://www.slideshare.net/stephenpa/long-after-the-thrill-sustaining-passionate-users

Anderson, Stephen P. 2011b. *Seductive interaction design: creating playful, fun, and effective user experiences*. Berkeley, CA: New Riders.

Bailey, Dwayne. 2007. Creating a single South African keyboard layout to promote language. *Lexikos* 17.1: 212-25. http://www.ajol.info/index.php/lex/article/view/51533

Bernard, H. Russell. 1992. Preserving Language Diversity: Computers can be a tool for making the survival of languages possible. In *Cultural Survival Quarterly*, p. 15. Cambridge, United States, Cambridge: Cultural Survival, Inc.

Bird, Steven & Gary Simons. 2003. Seven dimension of portability for language documentation and description. *Language* 79.3: 557-82.

Byers, Emily M., Andrew Brovey & Jane Zahner. 2004. Teaching and Learning Keyboarding. *Action Research Exchange* 3.1: <Accessed: 23 Jaunuary 2013>. http://teach.valdosta.edu/are/vol3no1/pdf/embyers-article.pdf

Cahill, Michael & Elke Karan. 2008. Factors in designing effective orthographies for unwritten languages. *SIL Electronic Working Papers* 001. Dallas, Tx.: SIL International. <Accessed: 15 May 2012>. http://www.sil.org/silewp/abstract.asp?ref=2008-001

Csató, Éva Á. & David Nathan. 2007. Multiliteracy, past and present, in the Karaim communities. In Peter K. Austin (ed.), *Language Documentation and Description* (4), 207-30. London: The Hans Rausing Endangered Languages Project, School of Oriental and African Studies.

Dauenhauer, Nora Marks & Richard Dauenhauer. 1998. Technical, emotional, and ideological issues in reversing language shift: examples from Southeast Alaska. In Lenore A.

Grenoble & Lindsay J. Whaley (eds.), *Endangered languages: language loss and community response*, 57-98. Cambridge: Cambridge university press.

Diki-Kidiri, Marcel. 2011. How to Guarantee the Presence and the Life of a Language in Cyberspace. In Evgeny Kuzmin, Ekaterina Plys & Anastasia Parshakova (eds.), *Linguistic and Cultural Diversity in Cyberspace. Proceedings of the International Conference (Yakutsk, Russian Federation, 2-4 July, 2008)*, 230-2. Moscow: Interregional Library Cooperation Centre.

Durdin, Marc & John Durdin. 2011. Tavultesoft Keyman. Version: 7. Computer program. <Accessed: 15 May 2012>. http://www.tavultesoft.com/

Eisenlohr, Patrick. 2004. Language Revitalization and New Technologies: Cultures of Electronic Mediation and the Refiguring of Communities. *Annual Review of Anthropology* 33.1: 21-45. http://www.jstor.org/stable/25064844

Fortis, David. 2000. Developing a Sochiapan Chinantec orthography: linguistic factors and sociolinguistic results. Paper presented at Bilingualism at the Ends of the Earth, Depatment of General and Applied Linguistics, University of Waikato, Hamilton, New Zealand, (24th -26th November 2000).

Galla, Candace K. 2009. Indigenous Language Revitalization and Technology From Traditional to Contemporary Domains. In Jon Allan Reyhner & Louise Lockard (eds.), *Indigenous Language Revitalization: Encouragement, Guidance & Lessons Learned*, 167-82. Flagstaff, AZ: Northern Arizona University.

Guérin, Valérie. 2008. Writing an endangered language. *Language Documentation & Conservation* 2.1: 47–67. http://hdl.handle.net/10125/1804

Harvey, Chris. 2013. Languagegeek.com. <Accessed: 23 January 2013>. http://www.languagegeek.com/keyboard_general/all_keyboards.html

Himmelmann, Nikolaus P. 1998. Documentary and Descriptive Linguistics. *Linguistics* 36:161-95. <Accessed: 2 January 2011>. http://www.uni-muenster.de/imperia/md/content/allgemeine_sprachwissenschaft/dozenten-unterlagen/himmelmann/linguistics98.pdf

Hinkle, Lauren, Miguel Lezcano & Jugal Kalita. 2010. Designing Soft Keyboards for Brahmic Scripts. Paper presented at ICON-2010: 8th International Conference on Natural Language Processing, Kharagpur, India. <Accessed: 23 January 2013>. http://ltrc.iiit.ac.in/icon_archives/ICON2010/10Dec2010/Paper11-File39-Paper186.pdf

Holton, Gary. 2011. The role of information technology in supporting minority and endangered languages. In Peter K. Austin & Julia Sallabank (eds.), *The Cambridge Handbook of Endangered Languages*, 371-99. Cambridge: Cambridge University Press.

Hosken, Martin. 2001. An introduction to keyboard design theory: What goes where? In Melinda Lyons (ed.), *Implimenting Writting Systems: an Introduction*, Preliminary edn, 123-37. Dallas: SIL International.

InchWest. 2012. MapKeyboard. Version: 1.5. Computer program. <Accessed: 28 December 2012>. http://www.inchwest.com/mapkeyboard

Jany, Carmen. 2010. Orthography Design for Chuxnabán Mixe. *Language DocumentatIon & ConservatIon* 4.1: 231-53. <Accessed: 12 June 2012>. http://hdl.handle.net/10125/4481

Kano, Akiyo. 2008. MECE Method For Categorising Typing Errors. In David England (ed.), *Proceedings of the 22nd British HCI Group Annual Conference on People and Computers: Culture, Creativity, Interaction*, vol. 2, 249-50.

Kano, Akiyo & Janet C. Read. 2009. Text input error categorisation: solving character level insertion ambiguities using Zero Time analysis. *People and Computers XXIII Celebrating People and Technology: Proceedings of the 2009 British Computer Society*

Conference on Human-Computer Interaction, BCS-HCI 2009, Cambridge, United Kingdom, 1-5 September 2009, 293-302. ACM.

Kano, Akiyo, Janet C. Read, Alan Dix & I. Scott MacKenzie. 2007. ExpECT: An Expanded Error Categorisation Method for Text Input. In Linden J. Ball et al. (eds.), *People and Computers XXI – HCI… but not as we know it: Proceedings of HCI 2007*, vol. 1, 147-56. ACM Press.

Krishna, Ashish, Rahul Ajmera, Sandesh Halarnkar & Prashant Pandit. 2005. Gesture Keyboard - User centered design of a unique input device for Indic Scripts. Paper presented at 11th international conference on Human-Computer Interaction, HCI International 2005, 22-27 July 2005, Las Vegas, Nevada, USA. ＜Accessed: 23 January 2013＞. www.hpl.hp.com/india/documents/papers/gkbuserdesign.pdf

Krumsick, Travis. 2009. KeyTweak. Version: 2.3.0. Computer program. ＜Accessed: 28 December 2012＞. http://webpages.charter.net/krumsick/

Kutsch Lojenga, Constance. 2011. Orthography and tone. Paper presented at Linguistic Society of America Annual Meeting, Pittsburgh, Jan. 6-9, 2011, Symposium on Developing Orthographies for Unwritten Languages. ＜Accessed: 30 June 2012＞. http://www.sil.org/linguistics/2011LSASymposium/KutschLojenga.html

Lüpke, Friederike. 2011. Orthography development. In Peter Austin & Julia Sallabank (eds.), *Handbook of endangered languages*, 312-36. Cambridge: Cambridge University Press.

Mallet, Chris. 2012. AutoHotKey. Version: 1.1.09.02. Computer program. <Accessed: 28 December 2012>. http://www.autohotkey.com/

McLendon, William W. Jr. 2011. SIL Technology for Multilingualism in Cyberspace. In Evgeny Kuzmin, Ekaterina Plys & Anastasia Parshakova (eds.), *Linguistic and Cultural Diversity in Cyberspace. Proceedings of the International Conference (Yakutsk, Russian Federation, 2-4 July, 2008)*, 97-104. Moscow: Interregional Library Cooperation Centre.

Mugele, Robert Louis. 1982. *Tone and Ballistic Syllable in Lalana Chinantec*. Ph.D. dissertation, The University of Texas at Austin, United States -- Texas.

Norman, Donald A. 1981. Categorization of action slips. *Psychological Review* 88.1: 1-15. http://psycnet.apa.org/journals/rev/88/1/1/

Open Source Project. 2012. Inkey Keyboard Creator. http://code.google.com/p/inkey-keyboard-creator/

Pavlov, Nikolai. 2011. Increasing the Numbers of Sakha-Speaking Internet Users. In Evgeny Kuzmin, Ekaterina Plys & Anastasia Parshakova (eds.), *Linguistic and Cultural*

Diversity in Cyberspace. Proceedings of the International Conference (Yakutsk, Russian Federation, 2-4 July, 2008), 238-42. Moscow: Interregional Library Cooperation Centre.

Raymond, Martin. 2012. ScriptSource: Making information on the world's scripts and languages accessible. Paper presented at Charting Vanishing Voices, 29-30 June 2012, Cambridge, UK.

Read, Janet, Stuart MacFarlane & Chris Casey. 2001. Measuring the Usability of Text Input Methods for Children. In Ann Blandford, Jean Vanderdonckt & Phil Gray (eds.), *People and Computers XV—Interaction without Frontiers*, 559-72. London: Springer.

Santossio, Randy. 2011. SharpKeys. Version: 3.5. Computer program. <Accessed: 28 December 2012>. http://www.randyrants.com/2011/12/sharpkeys_35.html

Seifart, Frank. 2006. Orthography development. In Jost Gippert, Nikolaus P. Himmelmann & Ulrike Mosel (eds.), *Essentials of language documentation* (Trends in Linguistics Studies and Monographs 178), 275-99. Berlin: Mouton de Gruyter.

Silva, Kalena & Keola Donaghy. 2004. Ke A‘o Ho‘okeleka‘a‘ike: Hawaiian Language Instruction On The Internet. In Yoshiko Saito-Abbott, Richard Donovan & Thomas Abbott (eds.), *Language on the Edge: Implications for Teaching Foreign Languages and*

Cultures, Proceedings of Digital Stream 2003 (Emerging Technologies in Teaching Language and Culture IV). San Diego: Montezuma Publishing.

Snider, Keith L. 1999. *The geometry and features of tone*. Dallas: Summer Institute of Linguistics.

Tavultesoft. 2013. Keyboard Search. <Accessed: 23 January 2013>. http://keymankeyboards.com/

Vitsœ. 2012. Dieter Rams: ten principles for good design. <Accessed: 27 June 2012>. http://www.vitsoe.com/en/gb/about/dieterrams/gooddesign

Wong, Kenneth & Phet Sayo. 2004. Free / Open Source Software. United Nations Development Programme's Asia-Pacific Development Information Programme (UNDP-APDIP). <Accessed: 28 December 2012>. http://www.iosn.net/downloads/foss_primer_current.pdf

Zhozhikov, Anatoli, Yakov Aleksandrov & Alexander Varlamov. 2011. The Type Fonts of the Yakut Alphabet and Those of the Minority Peoples Residing in the Republic of Sakha (Yakutia): Challenges of Applying in Operating Systems. In Evgeny Kuzmin, Ekaterina Plys & Anastasia Parshakova (eds.), *Linguistic and Cultural Diversity in Cyberspace.*

Proceedings of the International Conference (Yakutsk, Russian Federation, 2-4 July, 2008), 250-3. Moscow: Interregional Library Cooperation Centre.

[i]An earlier version of this paper was presented as: Hugh Paterson III. 2012. Keyboard layout as part of language documentation: the case of the Me'phaa and Chinantec keyboards. CRASSH Conference Language Endangerment: Methodologies and New Challenges, July 6th 2012, Cambridge, UK.

[i] I am grateful to Rebecca Paterson and a host of SIL colleagues for the development of my thought and expression on these matters. Of course the errors are mine. Please direct all comments to the me at Hugh@thejourneyler.org or hugh.paterson@sil.org

[ii] The texts were both translations of the New Testament book James. The Spanish word count was 2165, while the Me'phaa word count was 2856 for the same set of verses.

[iii] On a physical ANSI keyboard, it is a little bit further away than on a physical ISO keyboard.

[iv] The Saltillo also does not appear graphically on the physical keyboard. This is not a major challenge to implement, but it means that the user will have to learn to strike a key which does not return an input corresponding to the key top.

[v] So should all characters be accessed the same way in keyboard layouts? More importantly if they are not accessed in the same ways then how should they be?

[vi] This truly seems counterintuitive to the potential of the modern operating system. If a reader has a comment on accomplishing this within MSKLC please do share. But to the best of my ability I have not found a way to accomplish this behavior.