VIETNAMESE CHARACTER ENCODING STANDARDIZATION REPORT

VISCII and VIQR 1.1
Character Encoding Specifications

Viet-Std, September 1992

Contents

Preface

The Vietnamese Standardization Group (Viet-Std) was formed in the fall of 1989 to promote the standardization of Vietnamese character encoding and to monitor ongoing work of international bodies in this regard. The group has been working on designing and implementing a code table that can be integrated into existing computing environments on many platforms. In addition, the group has contacted the two committees on multilingual character encoding—the Unicode Consortium and the International Standardizations Organization (ISO)—to request that the Vietnamese characters be encoded in a precomposed form in the same manner as other Latin-based European languages. All these efforts will be reported fully in a special issue to be published in the near future. This special report collects only works concerning the 7- and 8-bit encodings of the Vietnamese alphabet.

The first article is a cover letter in Vietnamese that summarizes the Viet-Std Group's work in the area of 7- and 8-bit Vietnamese character encoding.

The second article covers the Vietnamese character encoding in 7 and 8 bits. It reviews the pros and cons of current encoding schemes and discusses the vital need to integrate into existing computing environments that a standard must address. It then presents the Viet-Std's 8-bit proposal for the Vietnamese Standard Code for Information Interchange (known as VISCII) and a Vietnamese Quoted-Readable convention (VIQR) to represent Vietnamese characters in 7-bit ASCII. The article also examines some guidelines and conventions in handling Vietnamese electronic mail over 7-bit channnels, Vietnamese keyboarding, and adapting existing Vietnamese applications. It includes two appendices listing both VISCII and VIQR for reference purposes.

The third article is the Vietnamese translation of the above to serve the general Vietnamese community.

The last item included in this report is the announcement of the third release of public-domain Vietnamese software by the TriChlor Group. It summarizes software packages either developed or enhanced by TriChlor members and other independent developers to run on the Unix, X-Windows, DOS, and MS-Windows platforms.

Since the release of version 1.0 in January 1992, VISCII has undergone only one change—swapping the two characters ạ (`a dot-below`) and Õ (`O tilde`)— to accommodate MS-Windows. To reflect this change, the versions of VISCII and VIQR published in this report are called VISCII 1.1 and VIQR 1.1 although VIQR was unchanged. This report therefore supercedes all previous publications concerning VISCII and VIQR.

A brief history of the Vietnamese Standardization Group is in order. The group was born out of the Viet-Net electronic mailing list, which comprised members at companies and universities throughout the computing world. At the time, Vietnamese software for publishing existed commercially only on the personal computer platform, but each package was limited to its designed function and could not be easily intermixed with others. No standard encoding existed. Developers designed their own schemes to perform the job they needed. Even Viet-Net itself had invented a 7-bit mnemonic writing style for Vietnamese for use in e-mail. For example, "`Nước Việt Nam`" would be typed as "`Nu+o+´c Vie^.t Nam`." Everyone agreed that some form of standardization was necessary so as to promote availability of and access to Vietnamese computing, and Viet-Net provided the forum that electronically brought together for the first time a large number of individuals committed to this goal. With this specific interest in mind, Viet-Std was formed by Thành Văn Nguyễn, and standardization discussions moved to the mailing list "Viet-Std@images.Sun.COM" and later to "Viet-Std@Haydn.Stanford.EDU." Early contributors included Cường Tấn Nguyễn, Tâm Nguyễn, Nhân Trần, Randall Atkinson, Khoa Tôn, Khiêm Hồ, Tước Lương, and many others too numerous to acknowledge properly. Later joining the group were Cương Minh Bùi, Học Đình Ngô, and others. This special report is a testimony to the success of the group. In addition, the group has contributed its expertise to international organizations on matters relevant to Vietnamese encoding; the details will be reported in a separate issue.

It should be mentioned that TriChlor Software (a non-profit experimental group) was formed by Cường Tấn Nguyễn, Cương Minh Bùi, and Tín Lê to independently explore and implement any encoding scheme for Vietnamese to gain real experience with the pros and cons in each scheme. It has helped integrate VISCII-compliant Vietnamese into popular public domain software products.

It is our dream one day to be able to read, write, and exchange Vietnamese data of a common format on any machine, any platform, and to take advantage of all the processing tools that have been produced by the computing world. That dream, once a pure exercise in imagination, has today come many steps closer to realization.

Viet-Std Vietnamese Standardization Group

California, USA
September 1992

Thư Ngỏ Gửi Quí-Vị Trong Ngành Điện-Toán
Cùng Tất-Cả Quí-Vị Quan-Tâm
Đến Việc Mã-Hóa Chữ Việt-Nam

Kính thưa quí vị:

Chúng tôi là một nhóm chuyên viên Việt Nam ở hải ngoại cộng tác trong tinh thần vô vụ lợi để theo dõi và đóng góp ý kiến chuyên môn về chữ Việt Nam với các Viện Định Chuẩn Tin Học hoặc các công ty điện toán có tầm vóc quốc tế. Trong thời gian qua chúng tôi đã vận động tích cực với tổ chức Unicode và Viện Định Chuẩn ISO để họ tiêu chuẩn hóa bộ Việt-tự-mã (*Vietnamese character encoding*) trong khuôn khổ bộ mã chữ quốc tế 16-bit và 32-bit trong chiều hướng có lợi nhất. Ngoài ra, chúng tôi đã nghiên cứu và đề nghị bộ Việt-tự-mã 7-bit và 8-bit để giải quyết nhu cầu phát triển và trao đổi nhu liệu hiện nay. Bộ tự-mã 8-bit có tên Anh ngữ là *Vietnamese Standard Code for Information Interchange* hay gọi tắt là VISCII để phân biệt với các bộ tự-mã khác. Tiêu chuẩn 7-bit được gọi là Quy-tắc Đọc-Được-Trong-Ngoặc (*Vietnamese Quoted-Readable Specification*) hay gọi tắt là VIQR. Đây là đề tài chính mà chúng tôi muốn thảo luận với quí vị qua lá thơ này.

Trước hết chúng tôi xin trình bày về bộ tự-mã 8-bit VISCII. Điểm qua quá trình phát triển ngành điện toán dùng chữ Việt ở hải ngoại, chúng tôi nhận thấy hầu như mỗi công ty hay mỗi nhóm tự đặt ra cho mình một bộ Việt-tự-mã, với hệ quả tất nhiên là nhu liệu không thể trao đổi dễ dàng với nhau. Do đó chúng tôi đã quyết định nghiên cứu một bộ Việt-tự-mã tiêu chuẩn (*Vietnamese character encoding standard*) nhằm giải quyết tình trạng này.

Như quí vị đã rõ, đa số nhu liệu hiện nay được viết dựa trên nền tảng mỗi mẫu tự được mã hóa bằng 8-bit (1 byte). Với 8-bit chúng ta có thể mã hóa được 256 mẫu tự hoặc tín hiệu khác nhau. Vì những lý do có tính cách lịch sử, bộ tự-mã ASCII của Hoa kỳ dùng 128 mã số (*code point or code value*) đầu tiên đã trở thành tiêu chuẩn quốc tế. Bộ tự-mã này bao gồm 32 tín hiệu điều khiển (*control character*) có mã số từ 0 đến 31, và 96 mã số còn lại dành cho một số mẫu tự La tinh, dấu chấm câu hoặc ký hiệu. (Để tiện việc thảo luận, chúng tôi tạm dịch *control character* là kiểm-tự, các chữ còn lại là ký-tự. Đứng trên quan điểm điện toán, chúng ta phải định nghĩa thêm mã-tự là bất cứ cái gì được tượng trưng bằng một mã số; mỗi mã-tự tương ứng với một mã số và ngược lại.) Phần còn lại (128 mã số từ 128 đến 255) thường được quy định và sử dụng tùy theo nhu cầu của mỗi quốc gia hoặc từng bộ nhu liệu. Như vậy chúng ta có thể tiêu chuẩn hóa 128 mã số này cho các mẫu tự Việt Nam không nằm trong danh sách 128 chữ ASCII.

Trên thực tế việc mã hóa chữ Việt là cả một vấn đề phức tạp. Ngoài các phụ âm, chúng ta có 12 nguyên âm chính (A Ă Â E Ê I O Ô Ơ U Ư Y) và 60 nguyên âm khác được tạo thành do các nguyên âm chính kết hợp với 5 dấu giọng (sắc, huyền, hỏi, ngã, nặng). Như vậy chữ Việt có tất cả 144 nguyên âm vừa thường vừa hoa. Chúng ta có thể kể ra hai phương pháp chính để mã hóa nguyên âm Việt Nam như sau:

1. Mỗi nguyên âm chính và mỗi dấu giọng được xem như các mã-tự riêng biệt, nghĩa là chỉ cần 29 mã số để mã hóa 12 nguyên âm chữ thường, 12 nguyên âm chữ hoa và 5 dấu giọng. Thí dụ, mẫu tự **À** được xem như gồm có hai mã-tự `"A"` và mã-tự `"`"`.
2. Mỗi nguyên âm, và dấu giọng nếu có, được xem như một mẫu tự duy nhất (mã-tự), nghĩa là phải cần đến 144 mã số để mã hóa tất cả nguyên âm Việt.

Phương pháp đầu tiên, còn được gọi là phương pháp dấu rời, đã được áp dụng ở nhiều quốc gia dùng chữ La-tinh có dấu phụ (*diacritical mark*). Nhưng quá trình thử nghiệm cho thấy phương pháp này có khuyết điểm lớn lao về nhiều mặt như tốc độ xử lý giảm sút, nhu cầu về sức chứa (*storage*) và bộ nhớ (*memory*) gia tăng, việc thảo chương phức tạp, không thể tích hợp vào môi trường nhu liệu và cương

liệu hiện hữu ... Vì những lý do này, phương pháp dấu rời hầu như không còn được sử dụng trong kỹ nghệ điện toán hiện nay.

Để có thể hội nhập vào nền kỹ nghệ điện toán của thế giới, chúng ta phải chấp nhận giải pháp thứ hai, nghĩa là phải mã hóa một số lượng rất lớn mẫu tự Việt. Trừ một số chữ Việt đã có sẵn trong bộ tự-mã ASCII, chúng ta có tất cả 134 mẫu tự cần phải mã hóa trong khi chỉ có 128 mã số còn trống mà thôi. Cách giải quyết thông thường là tìm cách thay thế 6 mã-tự ASCII nào đó bằng 6 mẫu tự Việt. Có rất nhiều cách nhưng cách nào cũng vấp phải một số khuyết điểm riêng.

Sau khi phân tích kỹ càng vấn đề và cứu xét cả chiều hướng phát triển cương liệu và nhu liệu trong tương lai, nhóm Viet-Std đã giải quyết bài toán này dựa trên căn bản triệt để bảo toàn 96 ký tự ASCII trong vùng G0 (mang mã số 32 đến 127). Quyết định này được trình bày kỹ càng trong bài Anh Ngữ in lại trong tập này [xem tr. 6–22], nhưng trong phạm vi lá thơ này có thể được tóm tắt như sau: bằng cách bảo toàn 96 ký tự ASCII chúng ta có thể sử dụng hầu hết nhu liệu và cương liệu sản xuất khắp nơi trên thế giới mà không phải đầu tư nhân lực và tài nguyên vào việc điều chỉnh hoặc biến đổi cho thích hợp với chữ Việt Nam, cụ thể như bộ dịch (*compiler*), khiển hệ (*operating system*), khung X (*X windows*), v.v... Chỉ có cách giải quyết này mới giúp chúng ta sử dụng được những thành quả kỹ thuật mới mẻ nhất trên thế giới.

Với phương châm trên, chúng tôi đã thay thế 6 kiểm-tự ASCII trong vùng C0 bằng 6 mẫu tự Việt Nam Ả, Ã, Ẫ, Ỷ, Ỹ và Ỵ. Ngoài ra, dựa trên bộ tự-mã tiêu chuẩn 8859/Latin-1 dành cho các nước Tây Âu, chúng tôi cũng quyết định duy trì mã số của tất cả chữ Việt đã có sẵn trong tiêu chuẩn này. Tất cả chi tiết về bộ tự-mã 8-bit VISCII được trình bày rõ ràng trong chương 3 của bài Anh ngữ [xem tr. 8-11] và bản dịch Việt ngữ [xem tr. 26–29].

Sau đây chúng tôi xin sơ lược về Quy-định Đọc-Được-Trong-Ngoặc, VIQR. Đây là quy luật viết chữ Việt Nam bằng bộ tự-mã 7-bit ASCII của Hoa-Kỳ. Đã từ lâu cộng đồng người Việt ở hải ngoại thường hay sử dụng hình thức này để trao đổi điện thư bằng tiếng Việt trên các máy vi tính không có chữ Việt Nam. Theo quy định này, các dấu được đánh sau các nguyên âm; dấu sắc, huyền, hỏi, ngã, nặng được thay thế bằng các ký hiệu ASCII Hoa-Kỳ có dạng tương tự là `"´"`, `"`"`, `"?"`, `"~"`, `"."`, dấu trăng (˘) được thay bằng `"("`, dấu mũ bằng `"^"`, dấu móc (˒) bằng `"+"`, và chữ đ được thay thế bằng `"dd"`. Chẳng hạn, hai câu thơ Kiều của cụ Nguyễn Du:

```
Trăm năm trong cõi người ta
Chữ tài chữ mệnh khéo là ghét nhau
```

sẽ hiện ra như sau khi viết theo quy định VIQR:

```
Tra(m na(m trong co~i ngu+o+`i ta
Chu+~ ta`i chu+~ me^.nh khe´o la` ghe´t nhau
```

Tuy nhiên một số người lại thích dùng các ký hiệu khác, chẳng hạn như `"*"` tượng trưng cho dấu móc (˒), `"<"` thay cho dấu trăng (˘), `"\"` thay cho dấu huyền, v.v... Nhưng với vai trò là một tiêu chuẩn, VIQR phải ấn định một quy luật duy nhất và tối thiểu để làm cơ sở thống nhất cho việc trao đổi nhu liệu, nghĩa là mỗi dấu Việt Nam phải được tượng trưng bằng một và chỉ một ký tự ASCII mà thôi. Với sự sử dụng rộng rãi bàn đánh chữ ASCII trong giới điện toán Việt Nam, chúng tôi quyết định chọn quy định VIQR dựa trên nguyên tắc thực dụng là dễ đọc và dễ nhớ cho tuyệt đại đa số người dùng. Quy luật VIQR được mô tả chi tiết trong chương 4 của tài liệu Anh Ngữ và Việt ngữ đính kèm.

Tiện đây chúng tôi cũng xin nhấn mạnh là bộ Việt-tự-mã 8-bit VISCII và quy định 7-bit VIQR đã được phổ biến rộng rãi trên các mạng thông tin điện toán (*computer network*) ở Hoa Kỳ và các quốc gia tiên tiến khác trên toàn thế giới. Các chuyên viên điện toán Việt Nam ở hải ngoại đã dùng những tiêu chuẩn này để viết nhu liệu ứng-dụng (*software application*) hoặc nhu-liệu dụng-cụ (*software tool*) cho khiển hệ DOS và Unix. Những nhu liệu này đã được cả người viết lẫn người dùng trắc nghiệm trên các máy như PC, AT, 386/486, workstation, mainframe, v.v... trong một thời gian khá lâu dài và thực tế cho thấy tất cả đều vận hành tốt đẹp với dữ liệu Việt ngữ. Trong phạm vi lá thơ này, chúng tôi xin tóm tắt những sản phẩm sau đây:

- chương trình biến đổi dữ liệu ở dạng 7-bit VIQR sang dạng 8-bit VISCII và ngược lại.

- ứng dụng điều khiển bàn chữ Việt trong môi trường Unix, DOS, MS-Windows 3.1. Ứng dụng này cho phép người dùng đánh chữ Việt trên bàn chữ ASCII Hoa Kỳ nhưng màn ảnh sẽ hiện ra chữ Việt Nam.
- ứng dụng màn ảnh Việt chạy trong khung X (*X windows*) hoặc khung MS-DOS (*MS-DOS windows*).
- ứng dụng thư tín cho phép đọc và viết thư bằng 8-bit VISCII nhưng sẽ tự động biến đổi thư thành dạng 7-bit VIQR khi lưu trữ hoặc gửi đi.
- ứng dụng in dữ liệu 8-bit VISCII trên các máy in Laser, ma-trận-điểm (*dot matrix*), hoặc PostScript.
- ứng dụng viết bài (*editor*) dùng 8-bit VISCII.
- các nhu-liệu dụng-cụ và nhu liệu thư viện (*library*) thường dùng trên khiển hệ Unix, DOS.
- ứng dụng bảng tính (*speadsheet*) chạy trên Unix.
- ứng dụng sư phạm dùng để ra câu đố hoặc đề thi trắc nghiệm tiếng Việt.
- ứng dụng kiểm soát lỗi chính tả.
- ứng dụng trò chơi.
- các bộ phông chữ ở dạng điểm (*bitmap fonts*), TrueType, PostScript.

Ngoài ra còn nhiều sản phẩm nữa không tiện kể ra đây. Muốn biết chi tiết xin xem thông-báo về việc phát hành nhu-liệu Việt Nam đợt 3 [tr. 41]. Hiện chúng tôi đang phân công thiết kế thêm các bộ phông chữ (*fonts*) VISCII để sử dụng trên các máy in laser. Vì Viet-Std không có tính cách thương mại, tất cả các công trình nghiên cứu và sản phẩm nhu liệu của nhóm đều được phổ biến miễn phí. Một số công ty nhu liệu và tổ chức chuyên gia Việt Nam ở Hoa Kỳ đã tuyên bố ủng hộ bộ tự-mã 8-bit VISCII và quy-định 7-bit VIQR như VNU, TIẾN, Hội Chuyên Gia Việt Nam, v.v...

Bộ tự-mã VISCII và quy định VIQR là công trình nghiên cứu của đông đảo chuyên viên Việt Nam ở hải ngoại thuộc nhiều lãnh vực khác nhau và đã trải qua quá trình thử nghiệm khá lâu dài. Chúng tôi sẵn sàng gửi đến quí vị một số nhu liệu cần thiết để quí vị có thể tự mình chạy và quan sát ưu điểm của VISCII và VIQR một cách cụ thể. Xin quí vị nghiên cứu kỹ lưỡng tài liệu đính kèm và dành cho chúng tôi sự ủng hộ cần thiết để bộ Việt-tự-mã 8-bit VISCII và Quy định 7-bit VIQR trở thành tiêu chuẩn chính thức cho chữ Việt Nam.

Trân trọng kính chào quí vị và mong nhận được ý kiến của quí vị trong thời gian ngắn nhất.

Nhóm Nghiên Cứu Tiêu Chuẩn Tiếng Việt

California, USA
September 1992

A Unified Framework for Vietnamese Information Processing

Vietnamese Standardization Working Group[1]
September 1992[2]

ABSTRACT

Increasing demand for Vietnamese electronic information processing has seen answer in a wide array of Vietnamese-capable applications. The inevitable need for integration of Vietnamese into existing environments and the exchange of data among them point to the necessity of standardization. This paper presents the strategic and pragmatic technical considerations that must go into such a standard, and reviews existing conventions/proposals in these important contexts. A full description of the Viet-Std proposal is presented, including 1) an 8-bit, fully precomposed encoding table for Vietnamese Standard Code for Information Interchange (known as VISCII), 2) a 7-bit Vietnamese Quoted-Readable (known as VIQR) standard for data interchange over 7-bit channels, with a seamless interface to the 8-bit encoding, and 3) a keyboard user-interface specification that works transparently with both 1 and 2. Together, these provide a truly unified framework for a Vietnamese information processing environment with simplicity, efficiency, and straightforward integration. The real-world construction of this framework has proven quite successful in an array of compliant applications from a number of group and individual developers across a number of platforms, including Unix and its variants, the X window system, MS-DOS, Windows, and with ongoing work elsewhere.

1 INTRODUCTION

With the growing Vietnamese population abroad and the proliferation of computer usage within Viet Nam, the Vietnamese language has seen rapidly increasing representation in electronic information processing. The concomitant growth in demand for Vietnamese-capable software has resulted in successful launches of myriad vendors in the U.S. and elsewhere, mainly in the area of Vietnamese word processing. In addition, individual and group efforts have also been productive in providing Vietnamese-language users with high-quality public-domain applications. In Viet Nam, centers such as the Institute of Informatics have reported impressive progress on many fronts, among which is the Vietnamization of standard software packages [1].

All of the above illustrate two important points: 1) There are growing market demands for Vietnamese-capable processing engines, and 2) There is no shortage of technical talent to fulfill those demands. Unfortunately, therein lies a large problem: most existing Vietnamese applications have been designed to operate in the exclusive framework or environment of the developer, and all are incompatible with one another. As long as this trend continues, the application base for Vietnamese can never keep reasonable pace with demand. Users want to do more with Vietnamese than mere word processing, and to expect one single vendor to provide all potential applications across all platforms is to dream the impossible. Technicians providing these applications are limited to the Vietnamese tools they must themselves learn and develop from the ground up. Standardization is necessary. Anyone who has had to deal with the incompatibility between ASCII and EBCDIC can try to imagine a world where every machine is using a different character set, and appreciate how limited that world would be in its application base and how cumbersome in its data interchange. A uniform framework will greatly benefit both the user and the technician alike.

The proposal for any Vietnamese data standardization must take several important points in the proper contexts. First and foremost, since this discussion is geared toward existing 7- and 8-bit environments, the prime goal is straightforward and direct integration onto current platforms. The standard must work here and now. This implies the use of precomposed Vietnamese characters, because the handling of floating diacritics will never see full or simple support outside of specific contexts. The standard must be designed so as to take advantage of existing applications as much as possible. The

[1] Postal address: Viet-Std, 1212 Somerset Dr., San Jose, California 95132, USA. E-mail address: Viet-Std@Haydn.Stanford.EDU

[2] This version 1.1 supersedes version 1.0 of January 1992. The only significant difference is the exchange of the positions of ạ (`a dot-below`) and Õ (`O tilde`) in the 8-bit table.

familiar "don't reinvent the wheel" rule is not only an advantage—but a necessity—if a meaningful application base is to be established in any reasonable length of time. Furthermore, it is known that overall efficiency both in time and space is greater in processing precomposed character units when compared with the floating-diacritic approach [2]. Floating diacritics therefore must be limited to only where they are necessary and inevitable, such as in keyboard entry or 7-bit data transmission. There is no reason to require that all applications must deal with the complexities and inefficiencies of floating diacritics, for example, in 8-bit data processing, storage, transmission, screen rendering, or printing.

The second major context points to the pragmatic and vital consideration of existing precedents set in the Vietnamese software base. Standardization necessarily requires adaptation, but it makes little sense to propose to change the world so significantly that the inertia against large changes greatly delays adoption of the standard. The trend towards 16-bit and wider data standards for multinational character sets has gained momentum with the recent works of Unicode [3] and ISO 10646 [4]. However, the need for an 8-bit Vietnamese standard is irreplaceable until these new standards are fully supported and completely dominate the computing world. An 8-bit Vietnamese standard must not ignore existing software precedents so that it can gain speedy acceptance before it becomes obsolete.

Thirdly, the standard must address the issue of user interface; if not defining it, then at least consider its possible effects on the end-user. This relates primarily to the 7-bit keyboarding and representation of Vietnamese—in both instances diacritics are necessarily floating, and represented mnemonically by existing 7-bit characters with similar appearance. With keyboarding, one must preserve where possible existing practices such as that defined for the Viet-Net mailing list and the Usenet newsgroup Soc.Culture.Vietnamese, both with members worldwide. For 7-bit readable representation, the keyword is "readable." The goals here are to maintain a short learning time and to promote a uniform interface so that it is not necessary for a user to re-learn the particulars of every software installation before being able to use it effectively.

Finally, to every extent possible, the standard must stay within the framework of international standards, e.g., ISO-8859/x [5], in order to ensure compatibility with existing environments. For example, this goal means preservation of the ASCII encoding. It should extend also to the encoding into the same 8859/Latin-1 slots those Vietnamese characters that are already defined, thus ensuring that 8859/Latin-1 keyboards will work transparently for those Vietnamese characters. However, there are many standards requirements that are obsolete from a practical viewpoint. For example, in recent Unicode/ISO-10646 decisions, the prohibition from use of the available control character space—those with encodings between xx00h and xx1Fh, except for C0 itself—was discarded on the grounds that it was a waste of encoding space. As will be discussed later, the encoding of Vietnamese into the existing 8-bit space presents some well-known trade-offs. Where trade-offs are made, they must be justified with good reason—pragmatic preferred over theoretical.

These primary requirements are summarized as follows:

- R1. Straightforward and direct integration into existing platforms.
- R2. Ease of adaptation for existing software.
- R3. User-friendly mnemonic encoding scheme and interface.
- R4. Adherence to international standards.
- R5. Trade-offs made only on practical usage considerations and with good reason.

In the following section we present a brief review of the strengths and weaknesses of different approaches to Vietnamese encoding. Section 3 will describe the proposed 8-bit encoding table in detail. A quoted-readable encoding scheme encompassing 7-bit data streams, including electronic mail and keyboard input, is presented in Section 4. Finally, Section 5 outlines the particular rules and conventions relevant in some application-specific contexts.

2 REVIEW OF CURRENT CONVENTIONS

A review of current conventions used by software vendors reveals one distinct feature: virtually all realize the strengths of a precomposed encoding and adopt it as a primary requirement. The complications arise from a familiar fact: apart from the alphabetics already available in the ASCII standard, Vietnamese requires an additional 134 unique characters. Of these, 128 can be coded in the C1 and G1 areas. The allocation of the remaining 6 characters in the lower C0 and G0 space is handled with differing approaches:

- A1. Encode into 6 of the "least-used" G0 characters in the context of Vietnamese data processing.
- A2. Encode into 6 slots of the National Replacement Character[3] (NRC) set.

[3] This set contains 12 country-specific characters at code positions corresponding to ASCII characters #, $, @, [, \,], ^, `, {, |, }, ~.

A3. Drop 6 of the "least-used"[4] Vietnamese characters, typically accented capitals such as Å̉, Ã̉, Ẽ̉, Ỷ, Ỹ, and Ỵ.

A4. Map accented "y" combinations into corresponding "i" combinations, e.g., "kỹ sư" is replaced with "kĩ sư."

A5. Encode into the ASCII control space C0.

Approaches A1 and A2 both satisfy the typical needs of the word processing environments in which rarely used ASCII characters can be avoided, or employed by font shifting. However they both eliminate prospects for integration of Vietnamese into existing ASCII environments where all graphic characters in G0 are needed. A character that already serves one purpose cannot be re-used for another. First, it makes rendering of the needed G0 character incorrect, as it would now look like a Vietnamese character. The frequency of use of G0 characters in an integrated environment is far too high for this conflict to be tolerable. Second, while font shifting may be employed to remedy this in some situations, a more serious problem occurs when the Vietnamese character is needed. The environment would typically have assigned some specific meaning to the G0 character, particularly with those in the NRC set. Consider, for example, using the backslash character "\" for a Vietnamese character under Unix. The backslash is used for many escape mechanisms under Unix so that the Vietnamese character cannot simply be used but must be escaped in one way or another. This is more than just an inconvenience; it means data interchange is complicated by the fact that the escape mechanism will not be understood on another platform, and data integrity has thus not been preserved. A standard employing this approach fails at its basic mission: to provide cross-platform transparency. A similar case can be made for the other G0 characters.

Both A3 and A4 propose to limit Vietnamese language data in one way or another. Most agree that elimination of some Vietnamese characters are simply unacceptable; indeed, this point is so fundamental that we have in the foregoing chosen to assume it as a technical requirement without elaboration. However, it must be said that A4 is not a proposal without rationale. A school of thought exists that believes y's existing in words as a single vowel should be mapped to corresponding i's, as their pronunciations are indeed identical. The concept dates as far back as 1948 [6, 7]. However, it is not the function of an encoding standard to settle a linguistic issue, and hence A4 is also a bad choice.

The immediate objection to A5 is primarily in data communication channels where many C0 characters are used as data control. In addition, it also presents problems for integration into environments where some C0 characters are used in the keyboard interface and in data format controls, similar to the problem facing A1 and A2. However, as will be discussed further, judicious choice of the 6 C0 characters to be used has in practice been shown successfully to avoid characters that are significant in data communication. Furthermore, most data channels provide for clean transfer of binary data, and there is no reason to worry that arbitrary data bits cannot be employed over these binary routes.

With those particular cases where C0 is used in the keyboard interface, judicious choice as well as remapping of keys can minimize conflict. Data format control is application-specific but is typically scattered in C0 and C1. It is therefore a universal problem for integration because C1 is necessarily densely encoded, but, again, conflict can be avoided by studying significant applications. Finally, the choice can be made for 6 least-used Vietnamese characters so that the probability of conflict is greatly reduced.

It should be noted here that the foregoing discussion has subjected the alternatives to the requirements of integration into existing applications and platforms, as outlined in Section 1. The importance of this goal cannot be overstated, and it does present complications that result in the following Pragmatism Principle: it is obviously impossible to define a standard that would operate seamlessly with all existing applications, therefore pragmatic considerations must be made to make a standard workable in as many important applications and on as many platforms as possible, with emphasis on the word "workable."

3 VISCII: 8-BIT ENCODING SPECIFICATION FOR VIETNAMESE

3.1 MOTIVATION

The available body of evidence shows that alternative A5 described in the previous section, encoding into 6 of the C0 characters, has the greatest chance of success in fulfilling the requirements outlined in Section 1. The choice of the 6 C0 codes and the 6 least-used Vietnamese capital letters to encode, when made carefully, greatly reduces the probability of conflict for all practical purposes. Concerns regarding data communications are well addressed by avoiding C0 codes that are in fact often used for data control. Indeed, data communication concerns are more applicable to C1 and G1 encoding; a prominent example is electronic mail transfer through 7-bit gateways and mail agents. Communication failure here has in most cases been due to the use of the eighth bit and not because

[4] Least-used because they (a) rarely begin words and therefore do not often get capitalized, and (b) appear in fewer words.

of C0 encoding. In any event, the option exists for data to be sent in some "binary" mode, or to employ the Vietnamese Quoted-Readable format to be described in Section 4.

The overwhelming advantage of this approach is that it is readily and easily integrated into existing environments without many of the problems plaguing the other alternatives, if they can at all be integrated. As a testimony to the approach's successful application, this document itself was prepared using the TEX system under Unix. The text source was edited in an 8-bit X terminal window, using a minimally modified[5] version of Elvis, a public-domain 8-bit version of Unix's Vi text editor. Both TEX (a document preparation system) and Dvi2ps (a PostScript generator) readily accepted and processed Vietnamese (8-bit) data transparently. Many other applications including a spreadsheet, various text viewers, PostScript and dot-matrix printing, DOS's WordPerfect, Word, PC Tools, etc., have been tested and seen to operate well with Vietnamese text. Modifications, if any, were primarily in making these applications accept 8-bit data. An educational teaching tool for Vietnamese has also been produced using the C programming language with 8-bit Vietnamese strings embedded in the source code. With increasing system internationalization, applications and tools are being made 8-bit "clean," further facilitating integration of this Vietnamese encoding.

3.2 ENCODING RATIONALE

A basic requirement is to preserve the 7-bit ASCII graphic characters (G0) layout, since the emphasis is on integration. G0 was therefore left unchanged. For the 6 C0 characters, we first lay out the code space and consider typical usage, a sampler of which is in Table 1. The codes selected, STX (2), ENQ (5), ACK (6), DC4 (20), EM (25), and RS (30) present the least possible problems with data communication and significant applications considered. The use of ACK, for example, is actually context-dependent. In those protocols we have reviewed, it is only considered a "control" character outside of a data frame; within a data frame it is transfered without special interpretation. To reduce the probability of conflict even further, the 6 least-often used Vietnamese capital letters, Å, Ã, Ẽ, Ỷ, Ỹ, and Ỵ, are encoded into these slots.

The remaining task is to encode the other 128 Vietnamese characters into the extended ASCII space (C1 and G1). Since no unique international encoding standard exists in this region, the philosophy is to be as much conservative as possible so that in the worst case the user can still use all of the lower case Vietnamese letters.

[5] The modifications provided the keyboard interface described in later sections.

The encoding of C1 is less troublesome, although in application-specific contexts it has been found that some C1 characters are employed with special meanings. A review of ongoing work on 8-bit mail transport standardization indicates that C1 characters will be fully supported as graphic characters without special interpretation. Nevertheless, it is prudent to encode only uppercase characters into the C1 space.

For G1, the aim is to accommodate the popular PC character set (code page 850) and to adhere, if possible, to the 8859/Latin-1 mapping where Vietnamese-specific characters are already encoded.

Experience in development of this encoding on the MS-DOS platform motivates the consideration of line-drawing glyphs in the PC character set. In many situations where both Vietnamese and line-drawing characters are desirable but font switching is impossible, the best we can do is to preserve all the lower case Vietnamese characters and all the single- and double-line drawing characters. This means that code positions occupied by single- and double-line drawing characters must be populated with upper case letters. With this provision, the MS-DOS user can be supplied with either code pages containing all Vietnamese glyphs or code pages where a number of upper case Vietnamese characters are replaced by PC line-drawing characters. For existing applications, the user can choose the code page most appropriate for her purpose. Where the code page with line-drawing characters must be used, the penalty from missing Vietnamese characters has been minimized by the choice of the infrequently used ones. For new applications, code page switching can easily be done on the fly, if it is desired.

Compatibility with the 8859/Latin-1 standard is merely for user friendliness and is not mandatory. It is natural and reasonable for a user in France to expect that the same keystrokes producing "é" on the screen for French will do the same for Vietnamese. The motivation for this compatibility is the predominant and increasing availability of 8859/Latin-1 keyboards and font sets, e.g., Digital's VT-terminal series, Xterm keymaps, and Microsoft's Windows. Table 2 lists the subset of 8859/Latin-1 characters in G1 that are also Vietnamese.[6] It can be concluded that all 8859/Latin-1 text that contains characters mostly from G0 (ASCII) and this table, French text for example, is highly readable in the Vietnamese environment.

Finally, certain characters in G1 are not renderable in a number of applications such as character codes 160 (non-breaking space character in 8859/Latin-1), 202

[6] Note that the "đ" in Table 2 is actually a similar-looking Icelandic "edh" in 8859/Latin-1; the Vietnamese rendering form is better reflected in 8859/Latin-2.

Table 1: A sampler of possible C0 usage conflicts. Codes selected for this standard proposal are noted with a †.

CODE	COMM.	CTRL-	GENERAL	PRINTER (PC)	PC	UNIX	VI (Unix)
0	NUL	@	C string			strings	
1	SOH	A					
2†	STX	B					back screen
3	ETX	C	INTR		INTR	INTR	INTR
4	EOT	D	EOF			EOF	back tab
5†	ENQ	E					
6†	ACK	F					forw.screen
7	BEL	G	BEL	BEL	BEL	BEL	
8	BS	H	BS	BS	BS	BS	BS
9	HT	I	HT	HT	HT	HT	HT
10	LF	J	LF	LF	LF	LF	LF
11	VT	K		VT			
12	FF	L	FF	FF		FF	redraw
13	CR	M	CR	CR	CR	CR	CR
14	SO	N		wide on(IBM)			
15	SI	O		comp.on(IBM)			
16	DLE	P			Prt.on/off		
17	DC1	Q	XOFF	XOFF	XOFF	XOFF	
18	DC2	R		comp.off(IBM)		retype	
19	DC3	S	XON	XON	XON	XON	
20†	DC4	T		wide off(IBM)			forw.tab
21	NAK	U		clr buf(IBM)		kill	kill
22	SYN	V				literal	literal
23	ETB	W				werase	werase
24	CAN	X				kill	
25†	EM	Y				suspend	
26	SUB	Z			EOF	suspend	
27	ESC	[	ESC	ESC sequence	ESC	ESC	ESC
28	FS	\				quit	
29	GS	]	Telnet ESC				
30†	RS	^					
31	US	_			Windows		

Table 2: Vietnamese-specific characters already present in 8859/Latin-1.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Cx	À	Á	Â	Ã					È	É	Ê		Ì	Í		
Dx	Đ		Ò	Ó	Ô	Õ				Ù	Ú			Ý		
Ex	à	á	â	ã					è	é	ê		ì	í		
Fx	đ		ò	ó	ô	õ				ù	ú			ý		

(non-breaking space character on Macintosh), or 255. The list of potentially non-graphic characters in C1 and G1 can be quite large: nearly 30 characters in MS Windows 3.0 and roughly 25 characters in MS Windows 3.1. These positions must be populated with upper case characters in consistence with the above philosophy. In applications where font switching is allowed and upper case characters are blocked out, a solution is to supply fonts in pair: a normal font and a capital font. In the capital font all the positions that should be filled with lower case characters are actually filled with the corresponding upper case. When a capital letter in the normal font cannot be rendered, the user simply switches to the corresponding capital font and types in the corresponding lower case character.

With the above guidelines, the task is then to lay out the remaining Vietnamese characters in some fashion, perhaps even arbitrary. This has been done in such a way so as to provide some degree of symmetry simply for aesthetics. It turns out that all the above guidelines can be adhered to except for compatibility with the letter `Õ` (`O tilde`) in 8859/Latin-1. Note that the Vietnamese collating order cannot in any case be preserved, but this is not a major issue since collation for non-ASCII characters is well accepted to be a table-lookup problem.

The preceding guidelines have resulted in the VISCII 8-bit Vietnamese encoding proposal listed in Table 3. It is intended to be a single table that applies to Vietnamese data handling including storage, processing, transmission, and font encoding. This greatly simplifies the integration, implementation, and usage processes and is indeed one of the major strengths of the proposal.

4 VIQR: MNEMONIC ENCODING SPECIFICATION FOR VIETNAMESE

4.1 MOTIVATION

While the 8-bit specification attempts to standardize Vietnamese encoding in 8-bit environments, much remains to be addressed in important 7-bit environments such as electronic mail transport and other 7-bit data lines, as well as in keyboard entry applications where the interface for generating Vietnamese characters needs to be standardized.

Transporting more than 128 unique symbols over 7-bit data channels is not a problem specific to the Vietnamese language. Since its proposal in 1982, the Internet Simple Mail Transfer Protocol ("SMTP", [8]) has seen unrelenting efforts to extend it to accommodate 8-bit and wider-word data in European Latin scripts and Oriental ideographic characters (see, e.g., [9]). While clean 8-bit transport is highly desirable, all mail gateways are not going to be converted overnight. For the foreseeable future there is a need for unambiguous transport of Vietnamese text over existing 7-bit channels.

Indeed there is an *ad-hoc* standard in use on the Viet-Net mailing list and the Usenet newsgroup Soc.Culture.Vietnamese, where mnemonic use of appropriate characters to follow a vowel proves to be quite readable; for example, "Việt Nam" would be written as "Vie^.t Nam". However, this is troubled by the ambiguity in the multiple roles played by the mnemonic diacritical marks; for example, does "tha?" mean "tha?" or "thả"?

The Viet-Net convention is not far in concept from a quoted-readable format proposed by K. Simonsen [10, 11]. which disambiguates such texts by specifying text states at both the character and character set levels. Unfortunately, in its attempt to provide a universal solution to mnemonic encoding, the proposal does not provide a good answer for Vietnamese text. First, it restricts the

Table 3: VISCII 8-bit Encoding Standard Proposal for Vietnamese.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x	NUL	SOH	Ả	ETX	EOT	Ẵ	Ẫ	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1x	DLE	DC1	DC2	DC3	Ỷ	NAK	SYN	ETB	CAN	Ỹ	SUB	ESC	FS	GS	Ỵ	US
2x	SP	!	"	#	$	%	&	’	(	)	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL
8x	Ạ	Ắ	Ằ	Ặ	Ấ	Ầ	Ẩ	Ậ	Ẽ	Ẹ	Ế	Ề	Ể	Ễ	Ệ	Ố
9x	Ồ	Ổ	Ỗ	Ộ	Ợ	Ớ	Ờ	Ở	Ị	Ỏ	Ọ	Ỉ	Ủ	Ũ	Ụ	Ỳ
Ax	Õ	ắ	ằ	ặ	ấ	ầ	ẩ	ậ	ẽ	ẹ	ế	ề	ể	ễ	ệ	ố
Bx	ồ	ổ	ỗ	Ỡ	Ơ	ộ	ờ	ở	ị	Ự	Ứ	Ừ	Ử	ơ	ớ	Ư
Cx	À	Á	Â	Ã	Ả	Ă	ả	ã	È	É	Ê	Ẻ	Ì	Í	Ĩ	ỳ
Dx	Đ	ứ	Ò	Ó	Ô	ạ	ỷ	ừ	ử	Ù	Ú	ỹ	ỵ	Ý	ỡ	ư
Ex	à	á	â	ã	ả	ă	ữ	ẫ	è	é	ê	ẻ	ì	í	ĩ	ỉ
Fx	đ	ự	ò	ó	ô	õ	ỏ	ọ	ụ	ù	ú	ũ	ủ	ý	ợ	Ữ

use of mnemonics to the 83 invariant ISO-646 [12] graphic characters, which is a good idea in principle, but sacrifices readability in the process. For example, the counter-intuitive mnemonics for hook-above (dấu hỏi) and tilde (dấu ngã) are "2" and "?", respectively, in order to avoid "~" itself, which is not an invariant. The wide availability of ASCII keyboards to the great majority of Vietnamese users makes this too unreasonable a limitation in the context of Vietnamese processing. It should be noted that we are in fact arguing in favor of "readability for most" against "illegibility for all." Furthermore, with ongoing progress on keyboard and display internationalization, e.g., in graphical window environments where keyboard mapping and font switching are easily implemented, this availability is on the increase, further obsoleting the restriction.

The greater difficulty is that the two-character fixed-length encoding[7] cannot provide a readable or mnemonic representation of all Vietnamese characters, in particular those with 2 diacritical marks. The variable-length mnemonics[8] have been extended to include all Vietnamese characters, but this scheme is so cluttered with announcers and delimiters that readability and efficiency are near nil, keeping in mind that diacritics are heavily used in Vietnamese. While machine data translators will have little trouble with any "mnemonic" scheme, one that is directly accessible to human users, who are in many cases typing mail messages using 7-bit editors, needs to be more user-friendly. A Vietnamese user will not want to learn or remember among all possible combinations that, say, "a5" stands for "ắ", nor will she like typing sequences as long as "&_a(′_" for some letter in every word.

To satisfy the readability and flexibility requirements, a separate specification is necessary. It is better to adopt an approach like code-page switching under ISO-2022 [13] to switch the text into "Vietnamese" mode and optimize encoding according to the language state. Recently, van der Poel put forth a mnemonic proposal [14] which emphasizes language-specific conventions for these reasons. This proposal provides a means to specify the language state, each with its own (efficient) encoding method. Its strength lies in the flexible specification that conformant implementations "need not be able to display all of the character sets specified"; they have the option of stating messages such as "undisplayable Greek appeared here" for unsupported languages (for a more precise specification, see [14]). This allows networking communities to determine the best approach for encoding their own languages. The VIQR convention is compatible with this approach and should easily be incorporated into this framework.

The specification here encompasses all data streams including text transfer, file I/O, and keyboard entry. This principle has been the major reason for success in operating systems such as Unix, in which device-specific details are hidden as much as possible from the applications programmer, leaving a uniform interface above which tools such as common library routines can be shared. Indeed as the keyboard example above has implied, the characters actually typed by the user are often not different from the text data that is eventually stored or transmitted. It is therefore desirable to provide a common base on which to build data interpreters for all data streams, independent of the input source. In actual implementation, this has greatly facilitated development of the Vietnamese-capable software base.

In addition, the user stands to benefit tremendously from standardization of keyboard entry. One does not need to learn a different keyboard entry technique for each different Vietnamese application. If one standard keyboard model is fully supported by all Vietnamese software, a user familiar with the standard can sit down and start typing Vietnamese immediately. This standard defines the minimum expected behavior from compliant software; any additional input techniques can of course be incorporated as a superset of the standard behavior. This is discussed further in Section 5.2 on Vietnamese keyboarding.

4.2 QUOTED-READABLE SPECIFICATION (VIQR)

The mnemonic model from Viet-Net is fully employed in the specification. The Vietnamese QR comprises three major states: Literal, English, and Vietnamese. The Literal state is intended for completely transparent handling of literal data (except of course for the escape sequences into and out of Literal state). The English and Vietnamese states are designed for mixed use of English and Vietnamese, with each optimized in appearance as well as data size for texts containing mostly English and Vietnamese, respectively. In either state there exist methods for composing Vietnamese-specific characters, using a base vowel followed by one or two diacritics.

We first introduce the concept of implicit and explicit composition, then discuss how they are used in each of the states.

[7] The convention is "&xy", where x is a literal character and y represents some combining form.

[8] The convention is "&_xxxx_" where xxxx can be an arbitrary mnemonic sequence.

4.2.1 Implicit Composition

Implicit composition is useful for data containing a large percentage of Vietnamese characters.

With implicit composition, a sequence of a base vowel followed by one or two diacritical marks is combined into one Vietnamese letter as long as it is grammatically legal. This is best illustrated by examples:

```
a^       →  â
o+?      →  ở
ơ?       →  ở
Vie^.t   →  Việt
Viê.t    →  Việt
la´^n    →  lá^n (not lấn)
lá^n     →  lá^n (not lấn)
```

Note in the last two example that the sequence `"a^´"` is not grammatically equivalent to `"a´^"` or `"á^"`. In general a modifier (`"("`, `"^"`, `"+"`) must immediately follow the appropriate vowel in order to be combined.

The special sequence `"dd"` is composed into `"đ"`; `"DD"`, `"dD"`, and `"Dd"` all represent `"Đ"`.

The base vowels are: `a, ă, â, e, ê, i, o, ô, ơ, u, ư, y,` and their corresponding capitals. The encoding values are those listed in Table 3, the 8-bit VISCII proposed standard.

The diacritical marks are represented by ASCII characters having correspondingly similar appearances. Table 4 lists the 7 ASCII characters used as mnemonic replacements for the Vietnamese diacritics; the first three are modifiers, and the remaining five are tone marks.

Table 4: ASCII Mnemonics for Vietnamese Diacritics

Diacritic	Char	ASCII Code	Dấu
breve	(	0x28, left paren	trăng (˘)
circumflex	^	0x5E, caret	mũ (ˆ)
horn	+	0x2B, plus sign	móc (ʾ)
acute	´	0x27, apostrophe	sắc
grave	`	0x60, backquote	huyền
hook above	?	0x3F, question	hỏi
tilde	~	0x7E, tilde	ngã
dot below	.	0x2E, period	nặng

4.2.2 Explicit Composition

Explicit composition is associated with the concept of a leading character which explicitly announces the composition. The announcer character is the backslash (`"\"`, ASCII 0x5C), known here as `<COM>`. The subsequent combining characters are defined in the same way as those in implicit composition. Thus the examples given above would appear in explicit composition mode as:

```
\a^      →  â
\o+?     →  ở
Vi\e^.t  →  Việt
```

Explicit composition is useful for data containing mainly English text, as well as for maintaining real-time compatibility with keyboard character events, as will be discussed in Section 5.2 on Vietnamese keyboarding.

With the composition methods described, we are now ready to discuss how they are employed in each of the three states. The state of the data stream is specified by the two character sequence `<COM>x`, where `x` is specified below.

4.2.3 Literal State

The appearance of `<COM>L` or `<COM>l` in the data stream initiates the Literal state. This state is intended for near-perfect transparent literal data transfer. Neither implicit nor explicit composition is available here, nor is the `<COM>` character special, except when it is followed by one of the six characters `l, L, v, V, m` or `M` which initiates one of the three states.[9]

4.2.4 English State

The sequence `<COM>M` or `<COM>m` sets the data stream state to English. In English state, only explicit composition is supported. This means that in order to generate a Vietnamese letter, the announcer character `<COM>` must be used. A "composition" sequence not preceded by `<COM>` will be left uninterpreted. Examples:

```
\mD\u~ng, how are you?  →  Dũng, how are you?
\mKho\e?  kh\o^ng?      →  Khoẻ không?
```

As noted, the sequence `"you?"` above was not converted into `"youỉ"` because no composition was specified.

[9] To effect `<COM>L, <COM>M, and <COM>V` themselves, it is necessary to switch to either English or Vietnamese state and use the Character Literal feature available there.

4.2.5 Vietnamese State

The data stream state is set to Vietnamese when the sequence <COM>V or <COM>v is encountered. In Vietnamese mode, both explicit and implicit compositions are in effect. The following examples assume that the data stream is initially in English state:

```
\vCh\u+~ Vi\e^.t   →   Chữ Việt
\vChu+~ Vie^.t     →   Chữ Việt
Chu+~ \vVie^.t     →   Chu+~ Việt
```

The availability of implicit composition in Vietnamese state ensures that the text is not cluttered with unnecessary <COM>s, as would be the case in Vietnamese text using explicit composition. Explicit composition is included to maintain compatibility with the English state so that there is no need to define additional meanings for the <COM> sequences. Also, the real-time keyboard compatibility mentioned previously is also available in Vietnamese state through explicit composition.

4.2.6 Character Literals in English and Vietnamese States

Consider the following example:

```
\vDu~ng, how are you?   →   Dũng, how are yoủ
```

In this example, the sequence "you?" was interpreted as "yoủ" because the data stream was still in Vietnamese state. Thus it is sometimes desirable to suppress composition altogether without having to switch states. The *literal* property of the <COM> character conveniently accomplishes this. In either Vietnamese or English state, whenever <COM> is followed by a non-combining character *c* the result is the literal character *c* itself. The <COM> is discarded from the data stream. To get the <COM> character literally, use <COM><COM>. Consider the following examples:

```
\vddi dda^u?     →   đi đâủ
\vddi dda^u\?    →   đi đâu?
\vddi v\o^?      →   đi vổ
\vddi v\o^\?     →   đi vô?
\\               →   \
\\V              →   \V
\\M              →   \M
\\L              →   \L
```

4.2.7 Closure

The data stream supports another special character used to generate *explicit closure*. The closure character is CTRL-A (ASCII 0x01), known here as <CLS>. When <CLS> is encountered in the data stream, it immediately terminates any ongoing composition sequence. The <CLS> itself is always discarded, unless it appears in the literal sequence <COM><CLS>.

Explicit closure is useful in real-time character applications such as keyboard entry, when it is necessary to specify that a composition sequence has in fact ended and the input engine should not stay hanging and wait for more data.

5 SPECIFIC APPLICATIONS

This section outlines application-specific guidelines and conventions that have evolved in the software development community. It is intended to be a live and growing documentation of such discussions as more experience is gathered. Readers are welcome to participate in these discussions and contribute to the development of these guidelines in particular, and to the standards in general.

5.1 ELECTRONIC MAIL OVER 7-BIT CHANNELS

Many of the available channels for electronic mail currently still enforce the 7-bit limitation. The 8-bit character set defined in Section 3 cannot be transported verbatim over these channels. VIQR plays an important role here, as it provides for 7-bit transport of Vietnamese text without the ambiguity problem of deciding what to do with the double usage of a diacritical/punctuation mark, e.g., the hook-above or question mark, "?". Because of the 7-bit nature of these communications channels, mail agents will typically not encounter those Vietnamese-specific base vowels that are encoded in the G1 area, namely: ă, Ă, â, Â, ê, Ê, ô, Ô, ơ, Ơ, ư, and Ư. However, mail agents designed to work with 8-bit channels are still expected to handle the occurrence of these characters according to the complete VIQR, namely to combine base vowels and diacritical marks as appropriate, for example:

```
ă´   →   ắ
```

In order to be correctly interpreted, electronic mail messages must explicitly set the language state either in the headers or text body. One cannot assume what state the receiving input engine is in at the start of the message, since messages are not always read in message units, e.g., when a file containing multiple mail messages is scanned.

Furthermore, if a language state specification (\L, \V or \M) is present in a mail message, it is highly recommended that the message end in the Literal state. This helps applications reading multiple mail messages in one

data stream, such as a terminal application. It is useful because mail headers do not adhere to the VIQR, and they are more adversely affected when interpreted in non-Literal states.

5.2 VIETNAMESE KEYBOARDING

Keyboards are becoming increasingly internationalized. As mentioned in the 8-bit specification, this is the major reason for using the same code positions for those Vietnamese characters already present in ISO 8859/Latin-1. A Vietnamese keyboard driver designed to work in the 7-bit-only environment can assume that it will not encounter Vietnamese base vowels residing in G1. Keyboard drivers for the 8-bit environments, like 8-bit electronic mail agents (Section 5.1), must be prepared to accept any base vowel, including those encoded in G1.

The real-time echoing behavior of keyboard input during composition requires further specification. The options are to report the character only after the composition sequence has finished, or to report all intermediate forms and backspacing over them. Each has its own useful context as described below.

5.2.1 Immediate Echo for Implicit Composition

Implicit composition is designed to be convenient for a user processing data that is mostly Vietnamese. As such it is desirable for the keyboarding user to get immediate feedback on typed keys. With implicit composition, the keyboard works in *immediate-echo* mode. Keypresses immediately generate key events. If a character is subsequently composed with a diacritical mark, a *backspace* (typically `BS`, ASCII 0x08) is sent followed by the new composed character. This cycle continues as long as composition is possible. The sequence of events for the key sequence `"a^´n"` under immediate echo is:

1. user types `a`, `a` is sent to the application
2. user types `^`, `BS` and `â` are sent
3. user types `´`, `BS` and `ấ` are sent
4. user types `n`, the single key `n` is sent

The actual *backspace* character code may vary depending on the system, application, and user settings. The keyboard interface should use the appropriate code, and/or allow the user to specify the preferred *backspace* character.

5.2.2 Delayed Echo for Explicit Composition

When a composition sequence is started, the keyboard interface must not send any key events to the application expecting keyboard input until the sequence is terminated. Composition may end either naturally when the interface receives a character that cannot be composed into the sequence, or when the closure character `<CLS>` is received. A single key event for the composed character is then sent to the application above. Subsequent processing can proceed naturally. Consider what happens when the user types the sequence `"\a^´n"` under delayed echo:

1. user types \, no key is sent to the application
2. user types `a`, no key is sent
3. user types `^`, no key is sent
4. user types `´`, the single key `ấ` is sent
5. user types `n`, the single key `n` is sent

Or an example involving closure, `"t\o+<CLS>"`:

1. user types `t`, the key `t` is sent
2. user types \, no key is sent
3. user types `o`, no key is sent
4. user types `+`, no key is sent
5. user types `CTRL-A`, the single key `ơ` is sent

Note that without the closure key the keyboard interface would still be left hanging after the `"+"` key has been pressed, because the user can still enter a tone mark as part of the composition sequence.

This *delayed-echo* behavior for explicit composition is designed to ensure compatibility with applications expecting single key events for each character, particularly in the English state where only explicit composition is available.

While it is certainly possible to have immediate-echo in explicit composition or delayed-echo in implicit composition, these options are not useful and serve only to confuse the user learning how to use a Vietnamese keyboard. It is therefore simplest to associate delayed-echo with explicit composition, and immediate-echo with implicit composition. These options make natural sense.

This standard defines the minimal "look-and-feel" behavior a user can expect from a compliant Vietnamese software package. A standardized interface decreases the required learning time for each new application. This standard does not preclude other input mechanisms to improve user-friendliness, e.g., intelligent menu-driven diacritics, or to assist in speed typing, e.g., through the use

of CONTROL or FUNCTION keys. Any enhancement in compliant applications is a bonus for the user, so long as such enhancements do not adversely conflict with the minimum expected behavior described here.

5.3 ADAPTING EXISTING VIETNAMESE APPLICATIONS

A realistic approach to standardization provides for the inertia against change in existing software applications. While it is desirable that the standard 8-bit encoding described here be fully supported, an alternative exists which is more amenable to rapid adoption. All applications should provide a means for importing and exporting data encoded using the VISCII 8-bit encoding table. At the same time, the VIQR keyboard interface should be implemented, at least as an optional entry method. Such moves are highly desirable both for the user and the vendor alike. The user will be able to use the software immediately because of the uniform keyboard interface, as well as process the same data in different applications and on different platforms, with increased productivity and interactivity among users. This ease of use means greater acceptance and a correspondingly larger customer base for the vendor.

6 SUMMARY & CONCLUSIONS

This paper has presented a proposal for standardization of Vietnamese information processing. A case has been made for the necessity of standardization; we hope to have encouraged vendors and users of Vietnamese alike to work together toward this goal to benefit everyone involved. Various encoding approaches were discussed, leading to the choice of the VISCII 8-bit encoding proposal. A single encoding table was presented that has been shown in actual practice to work well for Vietnamese including editing, processing, storage, transfer, font encoding, and printing. Where 8-bit data handling was not available or reliable, e.g., electronic mail transport, the Vietnamese Quote-Readable specification (VIQR) was introduced to provide a seamless filtering gateway. VIQR was defined to be input-source-independent and hence has been designed to be applicable to Vietnamese keyboard input as well as machine data filters. All of this was shown to have been integrated into existing environments facilitating the use of existing tools and applications—a major strength of the encoding. Finally, these specifications have been linked together seamlessly to include every point in the input-process/transfer-output cycle of data handling and provide for a truly unified framework for Vietnamese information processing.

References

[1] Bạch Hưng Khang. "Institute of Informatics,". Hà Nội, Việt Nam, February 1991.

[2] B. Jerman-Blažič, "Will the Multi-octet Standard Character Set Code Solve the World Coding Problems for Information Interchange?," *Computer Standards & Interfaces*, vol. 8, pages 127–136, 1988.

[3] The Unicode Consortium. *The Unicode Standard: Worldwide Character Encoding Version 1.0.* Addison-Wesley, Reading, MA, first edition, October 1991.

[4] ISO Technical Committee, "Universal Multiple-Octet Coded Character Set (UCS), ISO/IEC DIS 10646-1.2," Draft standard, International Organization for Standardization, 1992.

[5] International Organization for Standardization. *ISO 8859/x: 8-bit International Code Sets.* ISO, 1977.

[6] Famjxuæn Thais. *Việt Ngữ Cải Cách.* Tứ Hải, Hà Nội, Việt Nam, March 1948.

[7] Phạm Xuân Thái. *Chữ Việt Hợp Lí.* Tín-Đức Thư-Xã, Sài Gòn, Việt Nam, April 1958.

[8] J. Postel, "Simple Mail Transfer Protocol," RFC 822, USC Information Sciences Institute, August 1982.

[9] J. C. Klensin et al., "SMTP Extensions for Transport of Text-Based Messages Containing 8-bit Characters," Internet draft, Massachusetts Institute of Technology, July 1991.

[10] K. Simonsen, "Character Mnemonics & Character Sets," Internet draft, Danish Unix Users Group, January 1992.

[11] K. Simonsen, "Mnemonic Text Format," Internet draft, Danish Unix Users Group, August 1991.

[12] International Organization for Standardization. *ISO 646: 7-bit Coded Character Set for Information Interchange.* ISO, third edition, 1991.

[13] International Organization for Standardization. *ISO 2022: 7-bit and 8-bit Coded Character Sets—Code Extension Techniques.* ISO, third edition, 1986.

[14] E. M. van der Poel, "Multilingual Character Encoding for Internet Messages," Internet draft, Software Research Associates, Japan, January 1992.

[15] IBM. *System/370 Reference Summary–GX20-1850-5*, sixth edition, 1984.

[16] C.E. Mackenzie. *Coded-Character Sets: History and Development.* Addison-Wesley, Reading, MA, 1980.

[17] D.E. Knuth. *The TEXbook.* Addison-Wesley, Reading, MA, 1984.

Glossary of Terms

Announcer: A character or sequence of characters appearing in the data that signifies the start of some special sequence. In this text, it announces a Vietnamese composition sequence.

ASCII: American Standard Code for Information Interchange, a 128-character code used almost universally by computers for representing and transmitting characters data, in which each character corresponds to a decimal number between 0 and 127. Eight- or nine-bit codes of which the first 128 characters correspond to ASCII are called Extended ASCII; the additional characters are used to provide graphic characters for roman alphabets with diacritics, non-roman alphabets, special screen effects, etc.

Base Vowel: In this text, the unaccented Vietnamese vowels: a ă â e ê i o ô ơ u ư y (and their capitals). Contrast this with *Vowel*.

C0 Space: "Control characters" at code positions with hex values 00 through 1F.

C1 Space: "Control characters" at code positions with hex values 80 through 9F.

Code: In data communication, the numeric or internal representation for a character, e.g., in ASCII.

Code Page: Name used to denote glyph sets on the IBM PC. Abbreviated as CP. CP 850 is the multilingual code page, CP860 is for Portugal, CP863 is for French Canada, CP865 is for Norway.

Control Character: An ASCII character in the range 0 to 31, plus ASCII character 127, contrasted with the printable, or graphic, characters in the range 32 to 126. It is produced on an ASCII terminal by holding down the CTRL key and typing the desired character.

EBCDIC: Extended Binary Coded Decimal Interchange Code. The character code used on IBM mainframes. Not covered by any formal standards but described definitively in [15] and discussed at length in [16].

Floating Diacritics: A multiple-unit encoding approach for Vietnamese that treats the vowel and its diacritics as separate units. The diacritics may either precede or follow the vowel, or even the word. Contrast this with *Precomposed Character*.

Glyph: The physical appearance of a character as displayed on the screen or printed on paper.

G0 Space: "Graphic characters" at code positions with hex values 20 through 7F.

G1 Space: "Graphic characters" at code positions with hex values A0 through FF.

ISO: International Organization for Standardization. A voluntary international group of national standards organizations that issues standards in all areas, including computers, information processing, and character sets.

ISO 646: The standard 7-bit code set, equivalent to ASCII [12].

ISO Standard 8859: An ISO standard specifying a series of 8-bit computer character sets that include characters from many languages. These include ISO Latin Alphabets 1-9, which cover most of the written languages based on Roman letters, plus special character sets for Cyrillic, Greek, Arabic, and Hebrew [5].

ISO 8859/1: ISO Standard 8859 Latin Alphabet Number 1. Supports at least the following languages: Latin, Danish, Dutch, English, Faeroese, Finnish, French, German, Icelandic, Irish, Italian, Norwegian, Portuguese, Spanish, and Swedish [5].

ISO 2022 and ISO 4873: ISO standards for switching code pages [13].

ISO DIS 10646: The prospective 16- and 32-bit Universal Coded Set, (Draft International Standard) [4].

Latin: Referring to the Latin, or Roman, alphabet, comprised of the letters A through Z, or to any alphabet based upon it.

MS-DOS: Microsoft's Disk Operating System for microcomputers based on the Intel 80x86 family of CPU chips.

Modifier: A phonetic diacritical mark. The Vietnamese modifiers are: breve (trăng, ˘), circumflex (mũ, ˆ), horn (móc, ʾ).

PC: Personal Computer. In this text, the term PC refers to the entire IBM PC and PS/2 families and compatibles, which includes the AT, 286, 386, and 486 PC's.

PostScript: A page description language with graphics capabilities designed for electronic printing. The description is high-level and device-independent. PostScript is a trademark of Adobe Systems Incorporated.

Precomposed Characters: An encoding approach for Vietnamese that treats all vowel combinations as single units. Contrast this with *Floating Diacritics*.

TEX: A computerized typesetting system developed by Donald Knuth [17], providing nearly everything needed for high-quality typesetting of mathematical notations as well as of ordinary text. TEX is a trademark of the American Mathematical Society.

Tone Mark: A tonal diacritical mark that indicates the tone/accent. The Vietnamese tone marks are: acute (sắc), grave (huyền), hook above (hỏi), tilde (ngã), dot below (nặng).

Unicode: A 16-bit multilingual character code proposed by the Unicode Consortium [3].

Unix: A popular operating system developed at AT&T Bell Laboratories and noted for its portability.

Usenet: A worldwide network available to users for sending messages (or "news articles") that can be read and responded to by other users. Participating in Usenet is

like subscribing to a collection of electronic magazines. These "magazines," called *newsgroups*, are devoted to particular topics. The "Soc.Culture.Vietnamese" newsgroup is very popular among both Vietnamese and non-Vietnamese worldwide.

Viet-Std: A non-profit group of overseas Vietnamese and other professionals working on software & hardware standards for the Vietnamese language. Members of the group exchange ideas via electronic mail and meetings.

Vowel: In this text, a generic term applying to all Vietnamese vowels and their various combining forms, e.g., a, ă, and ắ. See *Base Vowel*.

Appendix A: Vietnamese Characters under VISCII and VIQR by Collating Order

Char	VIQR	VISCII	Char	VIQR	VISCII	Char	VIQR	VISCII	Char	VIQR	VISCII
A	A	065	N	N	078	a	a	097	n	n	110
Á	A´	193	O	O	079	á	a´	225	o	o	111
À	A`	192	Ó	O´	211	à	a`	224	ó	o´	243
Ả	A?	196	Ò	O`	210	ả	a?	228	ò	o`	242
Ã	A~	195	Ỏ	O?	153	ã	a~	227	ỏ	o?	246
Ạ	A.	128	Õ	O~	160	ạ	a.	213	õ	o~	245
Ă	A(	197	Ọ	O.	154	ă	a(	229	ọ	o.	247
Ắ	A(´	129	Ô	O^	212	ắ	a(´	161	ô	o^	244
Ằ	A(`	130	Ố	O^´	143	ằ	a(`	162	ố	o^´	175
Ẳ	A(?	002	Ồ	O^`	144	ẳ	a(?	198	ồ	o^`	176
Ẵ	A(~	005	Ổ	O^?	145	ẵ	a(~	199	ổ	o^?	177
Ặ	A(.	131	Ỗ	O^~	146	ặ	a(.	163	ỗ	o^~	178
Â	A^	194	Ộ	O^.	147	â	a^	226	ộ	o^.	181
Ấ	A^´	132	Ơ	O+	180	ấ	a^´	164	ơ	o+	189
Ầ	A^`	133	Ớ	O+´	149	ầ	a^`	165	ớ	o+´	190
Ẩ	A^?	134	Ờ	O+`	150	ẩ	a^?	166	ờ	o+`	182
Ẫ	A^~	006	Ở	O+?	151	ẫ	a^~	231	ở	o+?	183
Ậ	A^.	135	Ỡ	O+~	179	ậ	a^.	167	ỡ	o+~	222
B	B	066	Ợ	O+.	148	b	b	098	ợ	o+.	254
C	C	067	P	P	080	c	c	099	p	p	112
D	D	068	Q	Q	081	d	d	100	q	q	113
Đ	DD†	208	R	R	082	đ	dd	240	r	r	114
E	E	069	S	S	083	e	e	101	s	s	115
É	E´	201	T	T	084	é	e´	233	t	t	116
È	E`	200	U	U	085	è	e`	232	u	u	117
Ẻ	E?	203	Ú	U´	218	ẻ	e?	235	ú	u´	250
Ẽ	E~	136	Ù	U`	217	ẽ	e~	168	ù	u`	249
Ẹ	E.	137	Ủ	U?	156	ẹ	e.	169	ủ	u?	252
Ê	E^	202	Ũ	U~	157	ê	e^	234	ũ	u~	251
Ế	E^´	138	Ụ	U.	158	ế	e^´	170	ụ	u.	248
Ề	E^`	139	Ư	U+	191	ề	e^`	171	ư	u+	223
Ể	E^?	140	Ứ	U+´	186	ể	e^?	172	ứ	u+´	209
Ễ	E^~	141	Ừ	U+`	187	ễ	e^~	173	ừ	u+`	215
Ệ	E^.	142	Ử	U+?	188	ệ	e^.	174	ử	u+?	216
F	F	070	Ữ	U+~	255	f	f	102	ữ	u+~	230
G	G	071	Ự	U+.	185	g	g	103	ự	u+.	241
H	H	072	V	V	086	h	h	104	v	v	118
I	I	073	W	W	087	i	i	105	w	w	119
Í	I´	205	X	X	088	í	i´	237	x	x	120
Ì	I`	204	Y	Y	089	ì	i`	236	y	y	121
Ỉ	I?	155	Ý	Y´	221	ỉ	i?	239	ý	y´	253
Ĩ	I~	206	Ỳ	Y`	159	ĩ	i~	238	ỳ	y`	207
Ị	I.	152	Ỷ	Y?	020	ị	i.	184	ỷ	y?	214
J	J	074	Ỹ	Y~	025	j	j	106	ỹ	y~	219
K	K	075	Ỵ	Y.	030	k	k	107	ỵ	y.	220
L	L	076	Z	Z	090	l	l	108	z	z	122
M	M	077				m	m	109			

† VIQR also allows "Đ" to be represented by "Dd" or "dD". See Sec. 4.2.1.

Appendix B: Vietnamese Characters under VISCII and VIQR by Encoding Order

VISCII	Char	VIQR	Descriptive Name
002	Ả	A(	A breve hook-above
005	Ã	A(	A breve tilde
006	Ẫ	A^~	A circumflex tilde
020	Ỷ	Y?	Y hook-above
025	Ỹ	Y~	Y tilde
030	Ỵ	Y.	Y dot-below
065	A	A	A
066	B	B	B
067	C	C	C
068	D	D	D
069	E	E	E
070	F	F	F
071	G	G	G
072	H	H	H
073	I	I	I
074	J	J	J
075	K	K	K
076	L	L	L
077	M	M	M
078	N	N	N
079	O	O	O
080	P	P	P
081	Q	Q	Q
082	R	R	R
083	S	S	S
084	T	T	T
085	U	U	U
086	V	V	V
087	W	W	W
088	X	X	X
089	Y	Y	Y
090	Z	Z	Z
097	a	a	a
098	b	b	b
099	c	c	c
100	d	d	d
101	e	e	e
102	f	f	f
103	g	g	g
104	h	h	h
105	i	i	i
106	j	j	j
107	k	k	k
108	l	l	l
109	m	m	m
110	n	n	n
111	o	o	o

VISCII	Char	VIQR	Descriptive Name
112	p	p	p
113	q	q	q
114	r	r	r
115	s	s	s
116	t	t	t
117	u	u	u
118	v	v	v
119	w	w	w
120	x	x	x
121	y	y	y
122	z	z	z
128	Ạ	A.	A dot-below
129	Ắ	A(	A breve acute
130	Ằ	A(	A breve grave
131	Ặ	A(	A breve dot-below
132	Ấ	A^	A circumflex acute
133	Ầ	A^	A circumflex grave
134	Ẩ	A^?	A circumflex hook-above
135	Ậ	A^.	A circumflex dot-below
136	Ẽ	E~	E tilde
137	Ẹ	E.	E dot-below
138	Ế	E^	E circumflex acute
139	Ề	E^	E circumflex grave
140	Ể	E^?	E circumflex hook-above
141	Ễ	E^~	E circumflex tilde
142	Ệ	E^.	E circumflex dot-below
143	Ố	O^	O circumflex acute
144	Ồ	O^	O circumflex grave
145	Ổ	O^?	O circumflex hook-above
146	Ỗ	O^~	O circumflex tilde
147	Ộ	O^.	O circumflex dot-below
148	Ợ	O+.	O horn dot-below
149	Ớ	O+	O horn acute
150	Ờ	O+	O horn grave
151	Ở	O+?	O horn hook-above
152	Ị	I.	I dot-below
153	Ỏ	O?	O hook-above
154	Ọ	O.	O dot-below
155	Ỉ	I?	I hook-above
156	Ủ	U?	U hook-above
157	Ũ	U~	U tilde
158	Ụ	U.	U dot-below
159	Ỳ	Y	Y grave
160	Õ	O~	O tilde
161	ắ	a(	a breve acute
162	ằ	a(	a breve grave
163	ặ	a(	a breve dot-below

Appendix B: Vietnamese Characters under VISCII and VIQR by Encoding Order (continued)

VISCII	Char	VIQR	Descriptive Name
164	ấ	a^´	a circumflex acute
165	ầ	a^`	a circumflex grave
166	ẩ	a^?	a circumflex hook-above
167	ậ	a^.	a circumflex dot-below
168	ẽ	e~	e tilde
169	ẹ	e.	e dot-below
170	ế	e^´	e circumflex acute
171	ề	e^`	e circumflex grave
172	ể	e^?	e circumflex hook-above
173	ễ	e^~	e circumflex tilde
174	ệ	e^.	e circumflex dot-below
175	ố	o^´	o circumflex acute
176	ồ	o^`	o circumflex grave
177	ổ	o^?	o circumflex hook-above
178	ỗ	o^~	o circumflex tilde
179	Ỡ	O+~	O horn tilde
180	Ơ	O+	O horn
181	ộ	o^.	o circumflex dot-below
182	ờ	o+`	o horn grave
183	ở	o+?	o horn hook-above
184	ị	i.	i dot-below
185	Ự	U+.	U horn dot-below
186	Ứ	U+´	U horn acute
187	Ừ	U+`	U horn grave
188	Ử	U+?	U horn hook-above
189	ơ	o+	o horn
190	ớ	o+´	o horn acute
191	Ư	U+	U horn
192	À	A`	A grave
193	Á	A´	A acute
194	Â	A^	A circumflex
195	Ã	A~	A tilde
196	Ả	A?	A hook-above
197	Ă	A(	A breve
198	ả	a(?	a breve hook-above
199	ã	a(~	a breve tilde
200	È	E`	E grave
201	É	E´	E acute
202	Ê	E^	E circumflex
203	Ẻ	E?	E hook-above
204	Ì	I`	I grave
205	Í	I´	I acute
206	Ĩ	I~	I tilde
207	ỳ	y`	y grave
208	Đ	DD†	D bar
209	ứ	u+´	u horn acute

VISCII	Char	VIQR	Descriptive Name
210	Ò	O`	O grave
211	Ó	O´	O acute
212	Ô	O^	O circumflex
213	ạ	a.	a dot-below
214	ỷ	y?	y hook-above
215	ừ	u+`	u horn grave
216	ử	u+?	u horn hook-above
217	Ù	U`	U grave
218	Ú	U´	U acute
219	ỹ	y~	y tilde
220	ỵ	y.	y dot-below
221	Ý	Y´	Y acute
222	ỡ	o+~	o horn tilde
223	ư	u+	u horn
224	à	a`	a grave
225	á	a´	a acute
226	â	a^	a circumflex
227	ã	a~	a tilde
228	ả	a?	a hook-above
229	ă	a(	a breve
230	ữ	u+~	u horn tilde
231	ẫ	a^~	a circumflex tilde
232	è	e`	e grave
233	é	e´	e acute
234	ê	e^	e circumflex
235	ẻ	e?	e hook-above
236	ì	i`	i grave
237	í	i´	i acute
238	ĩ	i~	i tilde
239	ỉ	i?	i hook-above
240	đ	dd	d bar
241	ự	u+.	u horn dot-below
242	ò	o`	o grave
243	ó	o´	o acute
244	ô	o^	o circumflex
245	õ	o~	o tilde
246	ỏ	o?	o hook-above
247	ọ	o.	o dot-below
248	ụ	u.	u dot-below
249	ù	u`	u grave
250	ú	u´	u acute
251	ũ	u~	u tilde
252	ủ	u?	u hook-above
253	ý	y´	y acute
254	ợ	o+.	o horn dot-below
255	Ữ	U+~	U horn tilde

† VIQR also allows "Đ" to be represented by "Dd" or "dD". See Sec. 4.2.1.

Một Khuôn Khổ Thống Nhất Cho Việc Xử Lý Dữ Kiện Việt Ngữ

Nhóm Nghiên Cứu Tiêu Chuẩn Tiếng Việt[1]
Tháng Chín, 1992[2]

TÓM LƯỢC

Nhiều loại nhu liệu ứng dụng có thể dùng Việt ngữ đã xuất hiện nhằm đáp ứng nhu cầu xử lý dữ kiện Việt ngữ bằng điện toán ngày càng gia tăng. Nhu cầu tất yếu của việc tích hợp tiếng Việt vào môi trường điện toán hiện thời, cũng như việc trao đổi dữ kiện giữa các môi trường này đều cho thấy sự cần thiết phải có một tiêu chuẩn chung. Văn kiện này trình bày những cân nhắc kỹ thuật có tính cách thực tiễn và quan trọng mà một tiêu chuẩn như trên cần phải có, đồng thời cũng duyệt lại một số quy ước/đề án hiện hữu trong những lãnh vực quan trọng này. Văn kiện cũng trình bày trọn vẹn đề án của nhóm Viet-Std, gồm những điểm sau: 1) Bảng mã số 8-bit cho tất cả mẫu tự Việt nguyên vẹn (tên Anh ngữ là Vietnamese Standard Code for Information Interchange, gọi tắt là VISCII), 2) Một tiêu chuẩn 7-bit đọc-được-trong-ngoặc (có tên Anh ngữ là Vietnamese Quoted-Readable, gọi tắt là VIQR), dùng để trao đổi dữ kiện qua các mạch 7-bit, có giao diện suông sẻ với hệ mã tự 8-bit nêu trên, 3) Một quy định giao diện đánh chữ cho người dùng có thể vận hành dễ dàng với cả 1 và 2. Tất cả những điểm trên tạo thành một khuôn khổ thống nhất cho môi trường xử lý Việt ngữ, vừa đơn giản, vừa có hiệu năng và tích hợp dễ dàng. Việc xây dựng khuôn khổ này trên thực tế đã thành công xuyên qua những ứng dụng hợp thức sản xuất bởi một số tập thể và cá nhân trên một số hệ thống máy khác nhau, gồm cả khiển hệ Unix và những biến thể tương tự, hệ thống khung X (X-window), MS-DOS, Windows, và xuyên qua các công trình đang được thực hiện ở các nơi khác.

1 LỜI GIỚI THIỆU

Với số lượng người Việt tại hải ngoại ngày càng gia tăng và việc sử dụng máy vi tính ngày càng lan rộng tại Việt Nam, việc sử dụng chữ Việt trong lãnh vực xử lý tin tức đã tăng trưởng nhanh chóng. Đồng thời nhu cầu về nhu liệu tiếng Việt cũng gia tăng khiến cho nhiều công ty đã được thành lập và thành công tại Hoa Kỳ và các nơi khác, phần lớn chuyên về nhu liệu xử lý chữ Việt (*Vietnamese word processing*). Ngoài ra, nhiều tổ chức cũng như cá nhân đã nỗ lực cung cấp nhiều ứng dụng công cộng miễn phí với phẩm chất cao cho cộng đồng Việt Nam. Tại Việt Nam, các trung tâm như Viện Tin Học chẳng hạn đã ghi nhận sự tiến bộ khả quan về nhiều mặt, trong đó có việc Việt Nam hóa những bộ nhu liệu phổ thông [1].

Tất cả những điều trên cho thấy rõ hai điểm quan trọng: 1) Nhu cầu về nhu liệu dùng được với chữ Việt càng ngày càng tăng, và 2) Không thiếu tài năng để phục vụ những nhu cầu trên. Tiếc thay, chúng ta vấp phải một trở ngại rất lớn: hầu hết các ứng dụng dùng chữ Việt hiện thời chỉ hoạt động được trong một khuôn khổ hay một môi trường duy nhất của người sản xuất và các ứng dụng do các nhà sản xuất khác nhau không tương hợp với nhau. Khối ứng dụng dùng chữ Việt sẽ không bao giờ theo kịp đà đòi hỏi của thị trường một khi khuynh hướng nêu trên vẫn còn tồn tại. Người dùng muốn sử dụng tiếng Việt trong nhiều lãnh vực khác nữa chứ không chỉ giới hạn trong lãnh vực xử lý chữ mà thôi, và việc mong đợi một công ty cung cấp mọi ứng dụng cho mọi lãnh vực cho mọi giàn máy khác nhau là một việc vô tưởng. Ngoài ra, những chuyên viên viết các ứng dụng này lại bị giới hạn vào các nhu liệu dụng cụ (*software tools*) tiếng Việt mà chính họ phải học và tự phát triển lấy. Do đó, việc định chuẩn là một điều bắt buộc. Bất cứ ai đã gặp phải tình trạng bất-tương-hợp giữa ASCII và EBCDIC đều có thể mường tượng ra một môi trường mà mỗi máy dùng một bộ mã tự khác nhau, lúc đó mới nhận ra rằng số lượng ứng dụng cho môi trường đó rất là giới hạn và việc trao đổi dữ kiện rất phiền toái. Một khuôn khổ thống nhất sẽ tạo rất nhiều thuận lợi cho cả người dùng lẫn người viết nhu liệu.

Bất cứ dự thảo tiêu chuẩn tiếng Việt nào cũng phải cứu xét một số điểm trọng yếu trong đúng phạm vi của nó. Điểm đầu tiên và quan trọng nhất là vấn đề tích hợp. Vì văn kiện này chú trọng về môi trường 7-bit và 8-bit hiện thời, mục đích chính yếu phải là sự tích hợp (hội nhập) tiếng Việt trực tiếp và dễ dàng vào các hệ thống máy hiện thời. Tiêu chuẩn phải sử dụng được ngay

[1]Địa chỉ: Viet-Std, 1212 Somerset Dr., San Jose, California 95132, USA. Địa chỉ điện thư: Viet-Std@Haydn.Stanford.EDU

[2]Ấn bản 1.1 này thay thế ấn bản 1.0 xuất bản vào tháng giêng 1992. Sự khác biệt chủ yếu giữa hai ấn bản là sự hoán đổi vị trí của hai mẫu tự "ạ" và "Õ" trong bảng mã 8-bit.

lập tức. Điều này ngụ ý việc sử dụng các mẫu tự Việt nguyên vẹn (*precomposed character*), thay vì dùng các dấu rời (*diacritic*) đi kèm với các nguyên âm, vì ngoài các nhu liệu đặc biệt không có ứng dụng tổng quát nào có thể dùng được các dấu rời. Tiêu chuẩn đề ra phải được thiết kế khéo léo để tận dụng tối đa các nhu liệu hiện thời. Quy luật quen thuộc "Đừng phát minh bánh xe lần nữa" không những chỉ là ưu thế mà còn là bắt buộc nếu chúng ta muốn xây dựng một số ứng dụng cơ bản cần thiết trong một thời gian vừa phải. Ngoài ra, nói chung về thời-gian xử-lý (*time*) cũng như chỗ chứa (*space*), mọi người đều rõ là việc xử lý các mã tự nguyên vẹn có hiệu năng cao hơn là xử lý các mã tự dùng dấu rời [2]. Do đó việc dùng dấu rời phải được giới hạn vào những trường hợp cần thiết bất khả kháng, như khi đánh bàn chữ hay khi truyền dữ kiện 7-bit. Ngoài ra không có lý do gì để bắt buộc mọi ứng dụng phải đương đầu với sự phức tạp và kém hiệu năng của dấu rời trong việc xử lý, lưu trữ, truyền tin, tạo hình trên màn ảnh và in dữ kiện 8-bit.

Điểm quan trọng thứ nhì là phải cứu xét những tiền lệ đã có sẵn trong khối nhu liệu tiếng Việt. Bất cứ việc tiêu chuẩn hóa nào cũng đòi hỏi thời gian để thích nghi, nếu tiêu chuẩn đòi hỏi quá nhiều thay đổi thì sẽ gặp nhiều phản kháng của người dùng, và chỉ làm chậm trễ việc áp dụng tiêu chuẩn mà thôi. Các tiêu chuẩn dữ kiện khổ 16-bit hoặc rộng hơn nữa đã dần dần xuất hiện như Unicode [3] và ISO 10646 [4]. Trong khi chờ đợi các tiêu chuẩn khổ rộng này trở thành phổ thông, chúng ta cần phải có một tiêu chuẩn tiếng Việt 8-bit và tiêu chuẩn này phải được chấp nhận mau chóng để khỏi trở thành lỗi thời. Một tiêu chuẩn 8-bit muốn được chấp nhận mau chóng không thể bỏ qua những tiền lệ trong khối nhu liệu hiện thời.

Điểm quan trọng thứ ba là phải giải quyết vấn đề giao diện với người dùng; nếu không đặt ra thì tối thiểu phải suy xét ảnh hưởng của nó đối với người dùng. Điểm này phần lớn liên quan tới bàn đánh chữ 7-bit và cách tượng trưng tiếng Việt bằng các ký tự 7-bit — trong cả hai trường hợp này, các dấu phụ phải là dấu rời và được tượng trưng bởi các ký tự 7-bit với hình thù gần giống như dấu thật. Đối với cách đánh chữ Việt, chúng ta phải duy trì, nếu được, những thói quen đánh chữ đã được quy định trên diễn đàn Viet-Net (một mạng lưới điện thư của người Việt) và diễn đàn Soc.Culture.Vietnamese (nhóm thông tin Việt Nam trên mạng lưới Usenet) với thành viên trên toàn thế giới. Đối với cách tượng trưng chữ Việt bằng 7-bit, điều quan trọng là phải "đọc được." Mục đích ở đây là rút ngắn thời gian học và cổ võ một giao diện thuần nhất để người dùng không phải tốn thời gian học cách dùng cho mỗi bộ nhu liệu khác nhau.

Cuối cùng, tiêu chuẩn phải cố gắng bằng mọi cách tuân theo khuôn khổ của các tiêu chuẩn quốc tế, như ISO-8859/x [5], để đảm bảo sự tương hợp với những môi trường hiện hữu. Chẳng hạn, điều này đòi hỏi tiêu chuẩn tiếng Việt phải duy trì bảng mã số ASCII của Hoa Kỳ, cũng như phải duy trì vị trí của tất cả những mẫu tự Việt nào đã có sẵn trong bảng 8859/Latin-1 để bảo đảm cho bàn đánh chữ 8859/Latin-1 có thể hoạt động bình thường cho các mẫu tự đó. Tuy nhiên, về mặt thực tế có một số yêu cầu đã lỗi thời. Thí dụ như gần đây, ủy ban Unicode/ISO-10646 đã quyết định bãi bỏ việc cấm dùng vùng kiểm tự (*control characters*) — mã số từ `xx00h` cho đến `xx1Fh`, ngoại trừ C0 — với lý do là điều này chỉ làm phí phạm mã số vô ích. Như ta sẽ thấy dưới đây, việc ấn định các mẫu tự Việt vào những khoảng trống này có những điểm lợi hại của nó. Việc chọn lựa lợi hại phải được biện minh bằng những lý do chính đáng, và phải nghiêng về thực tiễn hơn là lý thuyết.

Những yêu cầu chủ yếu trên đây của một tiêu chuẩn được tóm tắt như sau:

R1. Tích hợp dễ dàng và trực tiếp vào các hệ thống máy hiện thời.

R2. Làm cho các nhu liệu hiện thời thích nghi dễ dàng với tiêu chuẩn mới.

R3. Phương pháp mã hóa và giao diện phải dễ nhớ và dễ sử dụng.

R4. Tuân theo các tiêu chuẩn quốc tế.

R5. Việc chọn lợi hại phải được cân nhắc dựa trên thực tiễn và có lý do chính đáng.

Trong phần sau đây, chúng tôi sẽ duyệt lại ưu khuyết điểm của những cách mã hóa tiếng Việt. Phần 3 sẽ mô tả chi tiết bảng mã số Việt ngữ 8-bit của Viet-Std. Phần 4 sẽ trình bày một phương pháp mã hóa "đọc-được-trong-ngoặc" áp dụng cho dòng dữ kiện 7-bit trong đó có điện thư và cách đánh chữ. Cuối cùng, Phần 5 phác họa một số luật lệ và quy ước riêng biệt thích nghi cho một vài ứng dụng cụ thể.

2 DUYỆT LẠI NHỮNG QUY ƯỚC HIỆN THỜI

Khi duyệt qua những quy ước dùng bởi các nhóm phát triển nhu liệu hiện thời, ta thấy một đặc điểm nổi bật: hầu hết mọi người đều công nhận ưu điểm của phương pháp mã hóa mẫu tự nguyên vẹn và chọn đó làm điều kiện tiên quyết. Tuy nhiên, khi chọn phương pháp này ta gặp phải những khó khăn quen thuộc: ngoài những mẫu tự đã có sẵn trong bảng ASCII, tiếng Việt Nam còn cần thêm 134 mẫu tự nữa. Trong số này, 128 mẫu tự có thể được đặt trong vùng C1 và G1. Sáu mẫu tự Việt còn lại có thể được đặt trong vùng C0 và G0 bằng những phương pháp khác nhau:

A1. Xếp vào chỗ của 6 mẫu tự tương đối "ít được dùng nhất" trong vùng G0 khi xử lý tiếng Việt.

A2. Xếp vào chỗ của 6 ký tự trong tập ký tự khả hoán NRC[3] (*National Replacement Character set*).

A3. Bỏ hẳn 6 mẫu tự tương đối "ít dùng nhất"[4] trong tiếng Việt, như Ả, Ã, Ẫ, Ỷ, Ỹ, và Ỵ.

A4. Thay thế các mẫu tự với gốc "y" bằng "i," thí dụ như "kỹ sư" sẽ trở thành "kĩ sư."

A5. Đặt 6 mẫu tự này vào vùng kiểm tự C0 của ASCII.

Giải pháp A1 và A2 thỏa mãn các nhu cầu tiêu biểu của những môi-trường xử-lý-chữ mà trong đó ta có thể tránh sử dụng những ký tự ASCII ít dùng hoặc sử dụng chúng bằng cách chuyển phông (*font shifting*). Tuy nhiên cả hai giải pháp này đều phá tan viễn tượng tích hợp tiếng Việt vào các môi trường ASCII hiện thời mà trong đó tất cả các ký tự trong vùng G0 đều phải được duy trì. Mỗi ký tự G0 chỉ có một nhiệm vụ riêng và không thể được dùng lại vào một nhiệm vụ khác được. Lý do thứ nhất là việc tạo hình của ký tự G0 đó sẽ bị sai vì hình tạo ra sẽ là một mẫu tự Việt. Vì các ký tự G0 được sử dụng rất thường xuyên, việc xung đột giữa mẫu tự Việt và ký tự G0 trong một môi trường tích hợp là việc không thể chấp nhận được. Lý do thứ hai là trong khi phương pháp chuyển phông có thể cải thiện tình trạng xung đột này trong vài trường hợp, chúng ta sẽ gặp khó khăn trầm trọng hơn Môi trường nhu liệu thường thường ấn định cho mỗi ký tự trong G0 một ý nghĩa riêng, đặc biệt là các ký tự NRC. Hãy xét trường hợp chúng ta thay thế ký tự gạch-chéo-ngược "\" bằng một mẫu tự Việt (như mẫu tự "ổ" chẳng hạn) trong môi trường Unix. Ký tự \ được dùng trong nhiều cơ chế thoát (*escape mechanism*) của Unix thành thử ra mẫu tự "ổ" không thể được dùng một cách bình thường mà phải được "thoát" đặc biệt bằng cách này hay cách khác. Đây không phải chỉ là một sự bất tiện nhỏ; việc trao đổi dữ kiện sẽ gặp rắc rối nhiều vì các hệ thống máy khác sẽ không hiểu cơ chế thoát đặc biệt này, do đó dữ kiện sẽ không được bảo toàn. Bất cứ tiêu chuẩn nào áp dụng giải pháp này sẽ không làm tròn chức năng căn bản của nó là cung cấp sự thuần nhất trên các hệ thống máy khác nhau. Trên đây là nói về ký tự "\" nhưng các ký tự G0 khác cũng có những khó khăn tương tự.

Các đề nghị A3 và A4 hạn chế dữ kiện Việt ngữ bằng cách này hoặc cách khác. Hầu hết mọi người đồng ý rằng việc loại bỏ một số mẫu tự Việt là việc không thể chấp nhận được; thật vậy, trong phần thảo luận ở trên, chúng ta đã coi việc bảo toàn tất cả mẫu tự Việt là một yêu cầu đương nhiên. Tuy thế, cần phải nói thêm nơi đây là đề nghị A4 không phải là không có lý do chính đáng. Đã có một trường phái tư tưởng nghĩ rằng trong các chữ chỉ chứa "y" (và dấu giọng nếu có) là nguyên âm duy nhất, chữ "y" có thể được thay thế bằng mẫu tự "i" vì cách phát âm của cả hai chữ tương đương với nhau. Khái niệm này đã có từ năm 1948 [6, 7]. Tuy nhiên, nhiệm vụ của một tiêu chuẩn mã hóa không phải là giải quyết các vấn đề liên hệ tới ngôn ngữ. Do đó việc chọn đề nghị A4 là một điều không tốt.

Lý do đầu tiên để bác bỏ đề nghị A5 chủ yếu phát xuất từ lãnh vực truyền tin vì các mạch chuyển tin (*data communication channel*) dùng nhiều mã tự C0 trong việc kiểm soát dữ kiện. Ngoài ra, đề nghị này tạo thêm một số khó khăn khi tích hợp tiếng Việt vào những môi trường mà trong đó một số mã tự C0 được dùng trong giao diện bàn đánh chữ (*keyboard interface*) và trong việc điều khiển khuôn thức dữ kiện (*data format control*), tương tự như những khó khăn gặp phải trong đề nghị A1 và A2. Tuy thế, như sẽ được trình bày trong các phần kế tiếp, việc chọn lựa thận trọng 6 mã tự C0 trong thực tế đã cho thấy có kết quả tốt đẹp mà vẫn tránh được các mã tự quan trọng dùng trong việc thông tin dữ kiện. Hơn nữa, hầu hết các mạch chuyển tin cho phép chuyển đạt dữ kiện 8-bit một cách minh bạch, trung thực, và không có lý do gì để e ngại rằng chúng ta không thể chuyển bất cứ mã số nào qua các mạch chuyển tin này.

Trong những trường hợp cá biệt mà C0 được dùng trong giao diện bàn đánh chữ, việc chọn lựa khôn khéo cũng như việc ấn định lại nhiệm vụ của các phím chữ sẽ làm giảm thiểu những xung khắc. Các mã tự dùng trong việc điều khiển khuôn thức dữ kiện thường thay đổi theo từng nhu liệu ứng dụng nhưng thông thường chúng nằm rải rác trong vùng C0 và C1. Do đó đây là một khó khăn phổ thông cho việc tích hợp bởi vì chúng ta cần phải dùng tất cả các mã số trong C1 cho chữ Việt. Tuy nhiên, một lần nữa, việc xung khắc có thể được giảm thiểu bằng cách nghiên cứu các ứng dụng quan trọng. Cuối cùng, chúng ta có thể chọn 6 mẫu tự Việt ít dùng nhất để làm giảm tối đa xác suất xung khắc.

Xin chú ý là phần trình bày trên đây đã phân tích từng đề nghị một để xem chúng có thỏa mãn các điều kiện cho việc tích hợp chữ Việt Nam vào các ứng dụng và các hệ thống máy hiện thời, như đã nêu ra trong Phần 1. Chúng ta không thể không nhấn mạnh tầm mức quan trọng lớn lao của mục tiêu tích hợp này. Mục tiêu này đã nảy sinh nhiều khó khăn và khiến cho chúng ta phải chấp nhận Nguyên Tắc Thực Dụng sau đây: Không có cách nào tạo ra một tiêu chuẩn vận hành hoàn hảo với tất cả mọi ứng dụng hiện thời, do đó, phải cân nhắc thực tế để đề ra một tiêu chuẩn có thể sử dụng được trong

[3] Tập NRC gồm có 12 ký tự ASCII ở vùng G0 là #, $, @, [, \,], ^, `, {, |, }, ~. Những ký tự này có thể được thay thế bằng các chữ khác tùy theo nhu cầu của mỗi quốc gia.

[4] Ít dùng bởi vì những mẫu tự này (a) ít khi bắt đầu một chữ và vì thế ít khi cần mặt hoa, và (b) ít xuất hiện trong các chữ Việt.

càng nhiều ứng dụng quan trọng và càng nhiều hệ thống máy càng tốt. Chúng tôi xin nhấn mạnh ở điểm "có thể sử dụng được."

3 VISCII: QUY ĐỊNH MÃ 8-BIT CHO VIỆT NGỮ

3.1 ĐỘNG LỰC

Những bằng chứng cụ thể cho thấy hướng giải quyết A5 mô tả ở phần trên, đặt 6 mẫu tự Việt vào vùng C0, là hướng có nhiều khả năng nhất để thỏa mãn những điều kiện đặt ra trong Phần 1. Việc lựa chọn 6 mã tự C0 và 6 mẫu tự Việt ít dùng nhất, nếu được cân nhắc kỹ càng, trên thực tế sẽ làm giảm rất nhiều xác suất xung khắc. Những ưu tư trong lãnh vực thông tin dữ kiện được giải quyết bằng cách tránh các mã tự C0 thường dùng để điều khiển khuôn thức dữ kiện. Thật ra, mối quan tâm trong lãnh vực thông tin dữ kiện nên hướng vào các ký tự trong vùng C1 và G1; một thí dụ nổi bật là việc truyền điện thư qua các cổng 7-bit và các bộ chuyển điện thư. Những thất bại trong việc chuyển tin ở đây thường là do việc dùng bit thứ tám chứ không phải vì các mã tự C0. Dầu thế nào đi nữa, ta vẫn có những phương thức để truyền dữ kiện trong trạng thái "8-bit" nào đó, hoặc dùng dạng "tiếng Việt đọc-được-trong-ngoặc" như được mô tả trong Phần 4.

Ưu điểm chính của phương pháp này là sự sẵn sàng và dễ dàng tích hợp vào môi trường hiện thời mà không gặp phải những trắc trở như các hướng giải quyết khác, giả sử rằng các hướng giải quyết kia có thể tích hợp được. Sự kiện bản văn kiện này được soạn bằng hệ thống TEX chạy trên Unix là một bằng chứng hùng hồn cho thấy sự thành công của hướng giải quyết này. Bản văn được soạn trong một khung-X 8-bit, dùng một ấn bản Elvis được biến đổi chút ít[5] (Elvis là một ấn bản 8-bit công cộng dùng giống như bộ soạn văn Vi của Unix.) Cả TEX (một hệ thống soạn tài liệu) và Dvi2ps (một ứng dụng tạo ra dạng PostScript) đều nhận và xử lý dữ kiện tiếng Việt (8-bit) một cách dễ dàng và thông suốt. Các ứng dụng khác gồm có bảng tính (*spreadsheet*), ứng dụng nhìn chữ (*text viewer*), in PostScript và ma trận điểm (*dot-matrix*), WordPerfect, Word, PC Tools của DOS, v.v... đã được thử nghiệm qua và chạy tốt đẹp với văn kiện tiếng Việt. Tất cả các biến đổi, nếu có, chủ yếu là để các ứng dụng này chấp nhận dữ kiện 8-bit. Một ứng dụng giáo khoa tiếng Việt đã được viết bằng ngôn ngữ thảo chương C với các câu văn tiếng Việt 8-bit đính kèm trong bản thảo chương. Với đà gia tăng quốc tế hóa nhu liệu hiện nay, các ứng dụng và nhu liệu dụng cụ đang được sửa đổi để nhận dữ kiện 8-bit, và do đó việc tích hợp của bộ mã tự Việt này lại càng dễ dàng hơn.

[5] Sửa để có khả năng đánh chữ Việt như sẽ được mô tả trong các phần sau.

3.2 CÁC LÝ DO BIỆN MINH VIỆC MÃ HÓA

Một điều kiện căn bản là phải bảo toàn bảng ký tự hình (*graphic character*) ASCII 7-bit (G0) vì mục tiêu là tích hợp với các môi trường hiện thời. Do đó, vùng G0 được giữ nguyên vẹn. Đối với 6 mã tự C0, đầu tiên chúng tôi phác họa ra vùng C0 và xem xét các cách dùng tiêu biểu và liệt kê trong Bảng 1. Các mã tự được chọn, STX (2), ENQ (5), ACK (6), DC4 (20), EM (25), và RS (30) là những mã tự ít gây trở ngại nhất cho việc truyền tin cũng như cho các ứng dụng quan trọng đã được cứu xét. Chẳng hạn như cách dùng của mã tự ACK thật sự tùy thuộc vào ngữ cảnh của nó. Trong những biên bản (*protocol*) mà chúng tôi đã duyệt qua, ACK chỉ được coi là một mã tự "điều khiển" khi nằm bên ngoài khung dữ kiện; bên trong khung dữ kiện nó được coi như một mã tự bình thường. Để làm giảm xác suất xung khắc hơn nữa, 6 mẫu tự chữ hoa tiếng Việt tương đối ít dùng nhất, Ả, Ã, Ẽ, Ỷ, Ỹ, và Ỵ, được đặt vào chỗ của 6 mã tự C0 này.

Vấn đề tiếp theo là mã hóa 128 chữ Việt còn lại trong vùng ASCII-rộng (C1 và G1). Vì không có một tiêu chuẩn quốc tế duy nhất trong vùng này để dựa theo, phương châm tốt nhất là cẩn thận tối đa để khi gặp trường hợp xấu nhất thì người dùng vẫn còn có thể sử dụng tất cả mẫu tự con.

Việc mã hóa các ký tự trong vùng C1 gặp ít rắc rối hơn, mặc dầu một vài ký tự C1 được sử dụng với ý nghĩa đặc biệt trong một số ứng dụng. Duyệt qua các công trình hiện thời để tiêu chuẩn hóa việc chuyển điện thư 8-bit, chúng ta thấy các ký tự C1 được coi là các ký tự hình và không bị gán cho một ý nghĩa đặc biệt gì cả. Tuy nhiên, đường lối thận trọng là chỉ nên đặt các mẫu tự chữ hoa vào trong vùng C1.

Đối với vùng G1, chúng tôi nhắm vào việc đáp ứng nhu cầu của bộ mã tự dùng trên máy vi tính PC (code page 850) và tuân theo, nếu có thể được, tập ký tự 8859/Latin-1 mà trong đó một số mẫu tự Việt đã có sẵn.

Kinh nghiệm trong việc thiết kế bộ mã tự cho hệ thống MS-DOS đã đưa đến việc cứu xét các ký tự vẽ đường thẳng trong bộ mã tự PC. Nếu muốn cho phép một số ứng dụng vừa dùng chữ Việt vừa dùng ký tự vẽ đường thẳng mà không phải chuyển phông, chúng ta chỉ có thể bảo toàn tối đa là các mẫu tự con và các ký tự vẽ đường đơn và đôi (*single- and double-line drawing characters*) mà thôi. Điều này có nghĩa là phải đặt các mẫu tự chữ hoa vào vị trí của các ký tự vẽ đường đơn và đôi. Với cách này, người dùng MS-DOS có thể được cung cấp tập mã tự có tất cả các mẫu tự Việt hoặc tập mã tự trong đó một số chữ hoa bị thay thế bằng các ký tự vẽ đường đơn và đôi. Đối với các ứng dụng có sẵn, người dùng có thể

Bảng 1. Những xung đột có thể xảy ra khi dùng C0. Các mã số dùng trong tiêu chuẩn 8-bit VISCII được nêu ra với dấu † bên cạnh.

MÃ SỐ	TÊN	CTRL-	CHUNG	MÁY IN (PC)	PC	UNIX	VI (Unix)
0	NUL	@	C string			strings	
1	SOH	A					
2†	STX	B					back screen
3	ETX	C	INTR		INTR	INTR	INTR
4	EOT	D	EOF			EOF	back tab
5†	ENQ	E					
6†	ACK	F					forw.screen
7	BEL	G	BEL	BEL	BEL	BEL	
8	BS	H	BS	BS	BS	BS	BS
9	HT	I	HT	HT	HT	HT	HT
10	LF	J	LF	LF	LF	LF	LF
11	VT	K		VT			
12	FF	L	FF	FF		FF	redraw
13	CR	M	CR	CR	CR	CR	CR
14	SO	N		wide on(IBM)			
15	SI	O		comp.on(IBM)			
16	DLE	P			Prt.on/off		
17	DC1	Q	XOFF	XOFF	XOFF	XOFF	
18	DC2	R		comp.off(IBM)		retype	
19	DC3	S	XON	XON	XON	XON	
20†	DC4	T		wide off(IBM)			forw.tab
21	NAK	U		clr buf(IBM)		kill	kill
22	SYN	V				literal	literal
23	ETB	W				werase	werase
24	CAN	X				kill	
25†	EM	Y				suspend	
26	SUB	Z			EOF	suspend	
27	ESC	[	ESC	ESC sequence	ESC	ESC	ESC
28	FS	\				quit	
29	GS	]	Telnet ESC				
30†	RS	^					
31	US	_			Windows		

Bảng 2. Những mẫu tự Việt Nam đã có sẵn trong tiêu-chuẩn 8859/Latin-1.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Cx	À	Á	Â	Ã					È	É	Ê		Ì	Í		
Dx	Đ		Ò	Ó	Ô	Õ				Ù	Ú			Ý		
Ex	à	á	â	ã					è	é	ê		ì	í		
Fx	đ		ò	ó	ô	õ				ù	ú			ý		

Bảng 3. VISCII. Dự thảo tiêu chuẩn 8-bit cho chữ Việt.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x	NUL	SOH	Ả	ETX	EOT	Ã	Ẫ	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1x	DLE	DC1	DC2	DC3	Ỷ	NAK	SYN	ETB	CAN	Ỹ	SUB	ESC	FS	GS	Ỵ	US
2x	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL
8x	Ạ	Ắ	Ằ	Ặ	Ấ	Ầ	Ẩ	Ậ	Ẽ	Ẹ	Ế	Ề	Ể	Ễ	Ệ	Ố
9x	Ồ	Ổ	Ỗ	Ộ	Ợ	Ớ	Ờ	Ở	Ị	Ỏ	Ọ	Ỉ	Ủ	Ũ	Ụ	Ỳ
Ax	Õ	ắ	ằ	ặ	ấ	ầ	ẩ	ậ	ẽ	ẹ	ế	ề	ể	ễ	ệ	ố
Bx	ồ	ổ	ỗ	Ỡ	Ơ	ộ	ờ	ở	ị	Ự	Ứ	Ừ	Ử	ơ	ớ	Ư
Cx	À	Á	Â	Ã	Ả	Ă	ả	ã	È	É	Ê	Ẻ	Ì	Í	Ĩ	ỳ
Dx	Đ	ứ	Ò	Ó	Ô	ạ	ỷ	ừ	ử	Ù	Ú	ỹ	ỵ	Ý	ỡ	ư
Ex	à	á	â	ã	ả	ă	ữ	ẫ	è	é	ê	ẻ	ì	í	ĩ	ỉ
Fx	đ	ự	ò	ó	ô	õ	ỏ	ọ	ụ	ù	ú	ũ	ủ	ý	ợ	Ữ

chọn tập mã tự nào thích hợp nhất cho mục đích của họ. Khi bắt buộc phải sử dụng tập mã tự vẽ đường thẳng, những thiệt hại vì thiếu các mẫu tự Việt cũng giảm thiểu vì các mẫu tự thiếu là mẫu tự chữ hoa tương đối ít được dùng. Đối với các ứng dụng mới, phương thức "thay tập mã tự" có thể được sử dụng, nếu muốn.

Nhu cầu tương hợp với tiêu chuẩn 8859/Latin-1 chỉ là một yếu tố phụ nhằm thuận tiện cho người dùng chứ không phải là điều bắt buộc. Một thí dụ cụ thể là người dùng ở Pháp nghĩ rằng họ chỉ cần nhấn cùng những phím chữ như nhau để tạo ra những mẫu tự Việt và mẫu tự Pháp giống nhau, như chữ "é" chẳng hạn. Đó cũng là điều tự nhiên và hợp lý. Việc chọn tương hợp với 8859/Latin-1 xuất phát từ sự phổ thông và thịnh hành của bàn đánh chữ và phông chữ 8859/Latin-1, như loạt thiết bị đầu cuối VT (*VT-terminal series*) của hãng Digital, bảng phím chữ Xterm, và ứng dụng khung của Microsoft (*MS Windows*). Bảng 2 liệt kê ra các mẫu tự 8859/Latin-1 trong vùng G1 trùng hợp với mẫu tự Việt.[6] Có thể kết luận rằng tất cả văn bản chứa chữ 8859/Latin-1 mà phần lớn các chữ là ASCII và những mẫu tự thuộc Bảng 2, chẳng hạn như văn bản tiếng Pháp, đều có thể đọc được, với tỉ lệ cao, trong môi trường Việt ngữ.

Cuối cùng, một số ứng dụng không hiển thị (*render*) được một số mã tự trong vùng G1 như những chữ có mã số 160 (mã tự cách-dính (*non-breaking space character*) trong 8859/Latin-1), 202 (mã tự cách-dính dùng trên máy Macintosh), hoặc 255. Danh sách những mã tự có thể không hiển thị được có thể rất dài: gần 30 mã tự trong MS Windows 3.0 và khoảng 25 mã tự trong MS Windows 3.1. Tuân theo phương châm cẩn thận đã nêu trên, chúng tôi phải đặt những mẫu tự chữ hoa vào những vị trí này. Trong những ứng dụng cho phép chuyển phông, việc hiển thị các chữ hoa có thể được giải quyết bằng cách cung cấp một cặp phông cho mỗi kiểu chữ: phông bình thường và phông chữ hoa. Trong phông chữ hoa, tất cả những mẫu tự đáng lẽ là chữ con đều được biến đổi thành chữ hoa tương ứng. Trong thực tế khi gặp một chữ hoa (thí dụ chữ Õ) không thể hiển thị được, người dùng chỉ cần chuyển qua phông chữ hoa tương ứng và đánh vào chữ thường (õ).

Sau khi đã định ra những nguyên tắc chỉ đạo nêu trên, công việc chỉ còn là xếp đặt các mẫu tự Việt còn lại theo một lối nào đó, hay có thể tùy tiện cũng được. Việc này đã được thực hiện sao cho bảng mẫu tự có một dạng tương đối cân đối, thẩm mỹ. Kết quả là tất cả nguyên tắc chỉ đạo nêu trên đều được thỏa mãn, ngoại trừ việc tương hợp với chữ Õ trong 8859/Latin-1. Cần ghi nhận là không có cách nào có thể bảo toàn được thứ tự các mẫu tự Việt, nhưng đây không phải là một vấn đề lớn vì thứ tự các mẫu tự không phải ASCII có thể được giải quyết bằng cách "tra bảng."

Bản dự thảo tập mã tự tiếng Việt VISCII 8-bit (Bảng 3) đã được hình thành dựa trên các nguyên tắc nêu trên. Chúng tôi có ý định xem đây là một bảng mã tự duy nhất áp dụng vào mọi việc sử dụng dữ kiện tiếng Việt như lưu trữ, xử lý, truyền, và mã hóa phông chữ. Điều này sẽ đơn giản hóa rất nhiều quá trình tích hợp, thực hiện, và sử dụng, và thật sự là một trong những điểm son của bản dự thảo này.

4 VIQR: QUY ĐỊNH VIỆT NGỮ ĐỌC-ĐƯỢC-TRONG-NGOẶC

4.1 ĐỘNG LỰC

Trong khi quy định 8-bit đang cố gắng tiêu chuẩn hóa chữ Việt trong môi trường 8-bit thì vẫn còn rất nhiều vấn đề phải được giải quyết trong môi trường 7-bit, chẳng hạn như việc chuyển điện thư và các đường dây chuyển tin 7-bit khác, cũng như các giao diện để tạo chữ Việt cũng cần phải được tiêu chuẩn hóa.

Sự khó khăn khi phải truyền nhiều hơn 128 mã số khác nhau qua kênh (*channel*) 7-bit không phải là một vấn đề riêng của tiếng Việt. Ngay từ sau khi Quy Luật Chuyển Điện Thư Liên Lưới Đơn Giản ("SMTP", [8]) được đề nghị năm 1982, đã có nhiều nỗ lực khai triển quy luật này nhằm đáp ứng nhu cầu chuyển dữ kiện 8-bit hoặc nhiều hơn cho những chữ Latin ở Âu Châu và những chữ tượng hình ở Đông phương (xem [9] chẳng hạn). Mặc dù nhu cầu chuyển vận 8-bit thật cần thiết, các cổng chuyển điện thư không dễ gì thay đổi trong một sớm một chiều. Trong tương lai trước mắt, chúng ta vẫn có nhu cầu chuyển điện thư tiếng Việt minh bạch qua mạch 7-bit.

Quả thật đã có một tiêu chuẩn đặc biệt dùng trên Viet-Net và trong nhóm thông tin Soc.Culture.Vietnamese trên mạng lưới Usenet. Đó là cách dùng những ký tự thích hợp dễ nhớ đi theo sau một nguyên âm để tượng trưng cho dấu phụ (thí dụ như ^ tượng trưng dấu mũ); chẳng hạn, "Việt Nam" được viết thành "Vie^.t Nam." Tuy nhiên, quy tắc này không được rõ ràng bởi vì một số ký tự dùng làm dấu phụ có thể đóng vai trò dấu chấm câu; thí dụ "tha?" có thể hiểu là "tha?" hoặc "thả."

Quy ước của Viet-Net cũng tương tự như khái niệm "đọc được trong ngoặc" do K. Simonsen [10, 11] đề nghị, đã làm sáng tỏ những trường hợp mơ hồ như trên bằng cách quy định thêm trạng thái cho bản văn ở cả cấp bậc mẫu tự lẫn cấp bậc hệ mẫu tự. Không may, trong nỗ lực cung cấp một giải pháp cho toàn thế giới, đề nghị này không giải quyết thoả đáng tiếng Việt. Đầu tiên, quy tắc này giới hạn những ký tự dễ nhớ trong tập hợp 83

[6] Lưu ý rằng chữ "đ" trong Bảng 2 thật ra là chữ "edh" của tiếng Băng Đảo theo 8859/Latin-1; dạng "đ" của tiếng Việt đúng theo bảng 8859/Latin-2 hơn.

ký tự hình cố định của ISO-646 [12]. Điều này tỏ ra tốt đẹp trên nguyên tắc, nhưng lại làm cho chữ Việt khó đọc. Thí dụ dấu "hỏi" và "ngã" được quy định lần lượt là "2" và "?", để tránh dùng dấu "~", vì dấu này không phải là một ký tự cố định. Sự phổ biến rộng rãi của các bàn đánh chữ ASCII trong đa số các người dùng chữ Việt cho thấy việc giới hạn này không hợp lý. Cũng xin nhấn mạnh là chúng tôi đang biện hộ quan điểm "dễ đọc cho đa số người dùng" hơn là "khó đọc cho tất cả người dùng." Hơn nữa, với đà tiến của việc quốc tế hóa các bàn đánh chữ và màn ảnh, thí dụ trong môi trường biểu họa khung (*graphical window environment*), việc tái định nghĩa các phím chữ và thay đổi phông có thể được thực hiện dễ dàng khiến cho giới hạn trên càng ngày càng lỗi thời.

Điểm khó khăn lớn hơn của quy tắc trên là phương pháp mã hóa bằng hai ký tự (chiều dài cố định)[7] làm cho chữ Việt khó đọc, nhất là những mẫu tự Việt có hai dấu phụ (thí dụ "ấ"). Phương pháp mã hóa dùng nhiều ký tự (chiều dài thay đổi)[8] cũng khó đọc và không có hiệu năng vì chứa đầy nghẹt những ký tự dùng để mở và đóng trong khi Việt ngữ lại dùng nhiều dấu. Mặc dù máy vi tính có thể đọc dễ dàng bất cứ phương pháp "dễ nhớ" nào, phương pháp áp dụng cho người dùng phải là phương pháp khiến họ có thể đọc và viết một cách dễ dàng khi dùng những nhu liệu viết bài (*editor*) 7-bit. Người dùng Việt ngữ không muốn phải học hoặc nhớ những chuỗi ký tự như "a5" tượng trưng cho "ấ", hoặc phải đánh những chuỗi dài như "&_a(´_" để tượng trưng cho một mẫu tự Việt nào đó trong cả một bài dài.

Để thỏa mãn nhu cầu dễ đọc và uyển chuyển, chúng ta cần ấn định một quy tắc khác. Cách tốt hơn là dùng phương pháp chuyển mã-tự-hệ (*code-page switching*) như quy định ISO-2022 [13] để chuyển văn bản vào trạng thái Việt ngữ và tối ưu hóa việc mã hóa tuỳ theo trạng thái ngôn ngữ. Gần đây, van der Poel đề xướng một phương-pháp dễ nhớ [14], nhấn mạnh về những quy ước riêng của từng ngôn ngữ. Đề nghị này cung cấp một phương tiện để quy định trạng thái ngôn ngữ, với mỗi ngôn ngữ tự quy định lấy cách mã hóa sao cho hữu hiệu nhất. Lợi điểm chính của nó là những ứng dụng dựa theo phương pháp này không cần phải tạo hình cho tất cả các tập mã tự được chỉ định trong dòng tin, mà chỉ cần tùy nghi báo tin về những ngôn ngữ không được hỗ trợ như "ở đây có chữ Hy Lạp không thể tạo hình được" (xin xem [14] để biết thêm chi tiết chính xác của quy định). Phương pháp này cho phép mỗi cộng đồng dùng cách thức riêng tốt nhất để mã hóa ngôn ngữ của mình. Quy ước VIQR phù hợp với đường lối này, và có thể được sát nhập dễ dàng vào khuôn khổ này.

Những quy định đặt ra ở đây sẽ áp dụng vào mọi dòng dữ kiện, gồm có chuyển chữ (*text transfer*), hồ sơ xuất nhập (*file I/O*), và cách đánh chữ. Nguyên tắc này là nguyên nhân chính đưa đến sự thành công của khiển hệ Unix, trong đó người viết nhu liệu không phải quan tâm đến các chi tiết đặc thù của các bộ phận phụ, mà chỉ quan tâm đến một giao diện đồng nhất (*uniform interface*) để từ đó có thể chia xẻ các nhu liệu thư viện của khiển hệ. Do đó điều cần thiết là cung cấp một căn bản chung để từ đó xây dựng những phần dịch (*data interpreter*) cho mọi dòng dữ kiện bất kể xuất xứ. Trên thực tế, điều này đã giúp cho việc phát triển những nhu liệu dùng chữ Việt được dễ dàng rất nhiều.

Ngoài ra, người dùng được hưởng lợi ích lớn từ việc tiêu chuẩn hóa cách đánh chữ. Họ không cần phải học nhiều cách đánh chữ khác nhau khi xử dụng những nhu liệu khác nhau. Nếu tất cả các nhu liệu cùng hỗ trợ một tiêu chuẩn chung, người dùng đã quen với tiêu chuẩn này có thể đánh tiếng Việt ngay mà không cần phải học lại. Tiêu chuẩn trong bài này quy định những đặc tính tối thiểu mà các nhu liệu hỗ trợ nó cần phải có; dĩ nhiên các kỹ thuật đánh chữ khác có thể được sát nhập cùng với tiêu chuẩn này thành một quy định tổng quát hơn. Điều này sẽ được bàn luận thêm nữa trong Phần 5.2 nói về cách đánh chữ Việt.

4.2 QUY ĐỊNH "ĐỌC-ĐƯỢC-TRONG-NGOẶC" (VIQR)

Quy định này hoàn toàn sử dụng kiểu mẫu "dễ đọc" của Viet-Net. Quy định Việt ngữ "đọc được trong ngoặc," VIQR, gồm có ba trạng thái: nguyên dạng, Anh Ngữ và Việt Ngữ. Trạng thái nguyên dạng chủ ý để chuyển tin y nguyên, không thay đổi (ngoại trừ những chuỗi thoát (*escape sequence*) để mở đầu và kết thúc trạng thái nguyên dạng). Trạng thái Anh Ngữ và Việt Ngữ được dùng chủ yếu cho những dòng dữ kiện có pha trộn Anh ngữ và Việt ngữ, với mỗi trạng thái được tối ưu hóa về hình thức cũng như khối lượng tùy theo văn bản chứa đa số là Anh ngữ hoặc Việt ngữ. Mỗi trạng thái đều có cơ chế riêng để viết mẫu tự Việt, bằng cách dùng một hoặc hai dấu phụ theo sau nguyên âm căn bản.

Trước hết xin giới thiệu khái niệm tạo chữ ngầm (*implicit composition*) và tạo chữ chỉ định (*explicit composition*).

4.2.1 Phép Tạo Chữ Ngầm

Phép tạo chữ ngầm thường được dùng một cách hữu hiệu cho những dữ kiện phần lớn là chữ Việt. Trong phép tạo chữ ngầm, mỗi khi một hay hai dấu phụ đi theo sau một nguyên âm cơ bản thì chúng sẽ kết hợp với nguyên âm đó thành một mẫu tự Việt duy nhất sao cho phù hợp với

[7] Cách dùng là "&xy", mà x là chữ chính và y là chữ phụ để ghép với x.

[8] Cách dùng là "&_xxxx_", mà xxxx có thể là bất cứ chuỗi ký tự nào.

quy tắc văn phạm. Thí dụ:

```
a^      →  â
o+?     →  ở
ơ?      →  ở
Vie^.t  →  Việt
Viê.t   →  Việt
la´^n   →  lá^n (không phải lấn)
lá^n    →  lá^n (không phải lấn)
```

Trong hai thí dụ chót, chuỗi `"a^´"` không tương đương với `"a´^"` hay `"á^"` về mặt văn phạm. Thông thường, ba ký tự phụ `"("`, `"^"`, và `"+"` phải theo sát sau các nguyên âm thích hợp thì mới kết hợp được.

Chuỗi đặc biệt `"dd"` kết hợp thành `"đ"`; `"DD"`, `"dD"`, và `"Dd"` đều tượng trưng cho `"Đ"`.

Những nguyên âm cơ bản gồm có `a, ă, â, e, ê, i, o, ô, ơ, u, ư, y,` và những mẫu tự hoa tương ứng. Mã số của những mẫu tự này được liệt kê trong Bảng 3, Dự Thảo Tiêu Chuẩn 8-bit VISCII.

Những dấu tiếng Việt được tượng trưng bằng những ký tự ASCII có hình dạng tương tự. Bảng 4 liệt kê 7 ký tự ASCII dễ nhớ được dùng để thay thế những dấu tiếng Việt. Phụ lục A và B liệt kê, theo thứ tự sắp chữ và thứ tự mã số, tất cả mẫu tự Việt và chuỗi VIQR tương ứng.

Bảng 4. Ký tự ASCII dùng thay dấu tiếng Việt

Dấu	Ký tự	Mã số ASCII
trăng (˘)	(	0x28, mở ngoặc
mũ (ˆ)	^	0x5E, mũ
móc (ʾ)	+	0x2B, dấu cộng
sắc	´	0x27, ngoặc đơn
huyền	`	0x60, ngoặc đơn ngược
hỏi	?	0x3F, dấu hỏi
ngã	~	0x7E, dấu ngã
nặng	.	0x2E, dấu chấm

4.2.2 Phép Tạo Chữ Chỉ Định

Phép tạo chữ chỉ định dựa trên khái niệm dùng một ký tự đi trước để báo tin việc tạo chữ một cách rõ ràng. Ký tự báo tin là gạch-chéo-ngược (`"\"`, ASCII 0x5C), từ nay sẽ gọi là ký tự `<COM>`. Những ký tự đi theo sau nó sẽ được kết hợp theo cùng quy tắc văn phạm như phép tạo chữ gián tiếp. Do đó, những thí dụ ở phần trên sẽ hiện ra như sau khi dùng phép tạo chữ chỉ định:

```
\a^      →  â
\o+?     →  ở
Vi\e^.t  →  Việt
```

Phép tạo chữ chỉ định tỏ ra tiện lợi trong dòng dữ kiện mà phần lớn là Anh ngữ, đồng thời cũng thích hợp với cách đánh chữ thực thời được đề cập ở Phần 5.2.

Sau đây, chúng ta sẽ phân tích cách xử dụng hai phép tạo chữ trên trong ba trạng thái. Trạng thái của dòng dữ kiện được chỉ định bằng chuỗi gồm hai ký tự `<COM>x`, với `x` được quy định như sau đây.

4.2.3 Trạng Thái Nguyên Dạng

Sự xuất hiện của `<COM>L or <COM>l` trong dòng dữ kiện mở đầu trạng thái nguyên dạng (hay nguyên trạng). Mục đích là để chuyển vận dữ kiện trong trạng thái hầu như nguyên vẹn không biến đổi. Cả hai phép tạo chữ ngầm và chỉ định đều không được áp dụng ở đây, kể cả ký tự `<COM>` cũng không có ý nghĩa đặc biệt trừ khi ký tự này được theo sau bởi một trong sáu chữ `l, L, v, V, m,` hoặc `M,` vì lúc đó nó sẽ mở đầu một trong ba trạng thái.[9]

4.2.4 Trạng Thái Anh Ngữ

Trạng thái Anh ngữ được bắt đầu bằng chuỗi `<COM>M` hay `<COM>m`. Trong trạng thái Anh ngữ, chỉ có phép tạo chữ chỉ định được hỗ trợ. Điều này có nghĩa là để tạo ra một chữ Việt, ta cần phải dùng ký tự báo tin `<COM>`. Nếu chuỗi ký tự không được mở đầu bằng `<COM>`, chuỗi này sẽ không được phép kết hợp. Thí dụ:

```
\mD\u~ng, how are you?  →  Dũng, how are you?
\mKho\e?  kh\o^ng?      →  Khoẻ không?
```

Như đã nói, chuỗi `"you?"` không được phép đổi thành `"youủ"` vì không có ký tự báo tin `<COM>` đi trước mẫu tự `u`.

4.2.5 Trạng Thái Việt Ngữ

Chuỗi `<COM>V` hay `<COM>v` chuyển trạng thái của dòng dữ kiện sang trạng thái Việt ngữ. Ở trạng thái này, ta có thể dùng cả hai phép tạo chữ ngầm và chỉ định. Những thí dụ sau đây dựa trên giả thiết trạng thái ban đầu của dòng dữ kiện là trạng thái Anh ngữ:

```
\vCh\u+~ Vi\e^.t  →  Chữ Việt
\vChu+~ Vie^.t    →  Chữ Việt
Chu+~ \vVie^.t    →  Chu+~ Việt
```

[9] Để có được chính chữ `<COM>L, <COM>M, hoặc <COM>V` ta cần phải chuyển sang trạng thái Việt hoặc Anh và dùng đặc điểm "nguyên dạng" quy định trong những trạng thái đó. Xem 4.2.6

Trạng thái Việt ngữ dùng phép tạo chữ ngầm hầu giúp cho văn bản gãy gọn hơn vì không phải chứa nhiều ký tự báo tin <COM> một cách rườm rà vô ích như trong trường hợp tạo chữ chỉ định. Ngoài ra, trạng thái Việt ngữ cũng cho phép tạo chữ chỉ định để duy trì sự tương hợp (*compatibility*) với trạng thái Anh ngữ nhằm tránh việc quy định thêm ý nghĩa của những chuỗi bắt đầu bằng ký tự <COM>. Ngoài ra, phương pháp tạo chữ chỉ định cũng duy trì sự tương hợp với cách đánh chữ thực thời (*real-time keyboarding*).

4.2.6 Nguyên Tự trong Trạng Thái Anh Ngữ và Việt Ngữ

Hãy xét thí dụ sau:

```
\vDu~ng, how are you?   →   Dũng, how are youỏ
```

Trong thí dụ này, chuỗi "you?" trở thành "youỏ" vì dòng dữ kiện đang ở trạng thái Việt ngữ. Vì thế ta thấy đôi khi cần tạm ngưng việc kết hợp chữ mà không phải chuyển trạng thái. Tính chất "nguyên dạng" của mẫu tự <COM> trở thành tiện dụng ở đây . Trong cả hai trạng thái Việt ngữ và Anh ngữ, mỗi khi ký tự <COM> được theo sau bởi một ký tự không thể kết hợp *c*, kết quả sẽ là chính ký tự *c* còn ký tự giới thiệu <COM> sẽ bị loại bỏ khỏi dòng dữ kiện. Để có ký tự <COM>, dùng <COM><COM>. Hãy xem những thí dụ sau:

```
\vddi dda^u?     →   đi đâủ
\vddi dda^u\?    →   đi đâu?
\vddi v\o^?      →   đi vổ
\vddi v\o^\?     →   đi vô?
\\               →   \
\\V              →   \V
\\M              →   \M
\\L              →   \L
```

4.2.7 Ký Tự Hoàn Cấu

Dòng dữ kiện có thể chứa một ký tự đặc biệt gọi là ký tự hoàn cấu (ký tự hoàn tất việc cấu tạo chữ, *closure character*) dùng để kết thúc việc kết hợp chữ đang diễn tiến. Ký tự này là CTRL-A (ASCII 0x1), từ nay gọi là <CLS>. Khi gặp <CLS> trong dòng dữ kiện, tất cả những việc kết hợp chữ đang tiến hành đều được kết thúc. <CLS> luôn luôn bị loại bỏ, trừ khi nó xuất hiện trong chuỗi nguyên dạng <COM><CLS>.

Ký tự hoàn cấu có ích trong những ứng dụng thực thời như đánh chữ, khi cần phải cho biết là chuỗi kết hợp đã kết thúc, và cơ phận nhận tin không cần phải chờ thêm dữ kiện nữa.

5 CÁC ỨNG DỤNG ĐẶC BIỆT

Phần này phác họa những nguyên tắc chỉ đạo và quy ước đặc thù cho những ứng dụng đã được dùng trong giới phát triển nhu liệu. Mục đích của nó là cung cấp một tài liệu sống bao gồm những kinh nghiệm tích lũy trong thời gian qua cũng như những kinh nghiệm sắp tới. Chúng tôi hoan nghênh độc giả tham gia vào những cuộc thảo luận này và cống hiến vào việc phát triển những nguyên tắc chỉ đạo nói riêng, và việc tiêu chuẩn hóa nói chung.

5.1 ĐIỆN THƯ CHUYỂN QUA MẠCH 7-BIT

Đa số những mạch hiện hữu dùng để chuyển điện thư vẫn còn ở trong giới hạn 7-bit. Tập mã tự 8-bit quy định ở Phần 3 không thể được truyền nguyên dạng qua những mạch này. Do đó VIQR đóng một vai quan trọng vì nó có thể dùng để chuyển văn bản tiếng Việt 7-bit một cách minh bạch, không bị mơ hồ vì tính cách lưỡng dụng của những ký tự vừa tượng trưng dấu phụ vừa tượng trưng dấu chấm câu như dấu "?". Do tính chất 7-bit của các mạch này, bộ chuyển thư sẽ không gặp phải những mẫu tự Việt nằm trong G1 như ă, Ă, â, Â, ê, Ê, ô, Ô, ơ, Ơ, ư và Ư. Tuy nhiên, bộ chuyển thư chế tạo cho mạch 8-bit sẽ phải giải quyết những mẫu tự này đúng theo quy tắc kết hợp VIQR, nghĩa là phải kết hợp nguyên âm cơ bản và dấu phụ nếu được. Ví dụ:

```
ă´   →   ắ
```

Để được hiểu đúng, điện thư phải chỉ định rõ ràng trạng thái ngôn ngữ, hoặc ở trong phần dẫn đầu (*header*), hoặc ở trong phần nội dung (*text body*) của thư. Chúng ta không thể phỏng đoán trạng thái của bộ phận nhận tin ở đầu mỗi lá thư, vì thư có thể được đọc từ một hồ sơ (*file*) gồm nhiều lá (*message*) chứ không phải chỉ có một lá, do đó khó biết đâu là chỗ bắt đầu của lá thư khác.

Hơn nữa, nếu trong thư có chứa một chuỗi chỉ định trạng thái ngôn ngữ (\L, \V hoặc \M), thì lá thư nên được kết thúc trong trạng thái nguyên dạng, nghĩa là kết thúc bằng <COM>L. Việc này giúp cho ứng dụng đọc thư đọc được những lá thư sau nằm cùng trong một hồ sơ, chẳng hạn ứng dụng đọc thư trên màn ảnh. Điều này tỏ ra ích lợi vì phần dẫn đầu của điện thư nói chung không tuân theo quy tắc VIQR, và do đó có thể bị hiểu sai khi không ở trong trạng thái nguyên dạng.

5.2 ĐÁNH CHỮ VIỆT

Bàn đánh chữ càng ngày càng được quốc tế hóa thêm. Như đã nói ở phần quy định 8-bit, đây là một lý do chính để dùng cùng một mã số cho những mẫu tự Việt đã có sẵn trong bảng ISO 8859/Latin-1. Bộ điều khiển bàn chữ Việt, được thiết kế để dùng trong môi trường 7-bit mà

thôi, có thể giả sử là sẽ không bao giờ gặp những nguyên âm cơ bản của tiếng Việt nằm trong vùng G1. Nhưng bộ điều khiển bàn chữ dùng trong môi trường 8-bit, cũng như bộ nhận tin 8-bit (Phần 5.1), phải sẵn sàng tiếp nhận bất cứ nguyên âm cơ bản nào, kể cả những nguyên âm nằm trong G1.

Việc tạo hình mẫu tự trên màn ảnh (*echoing behavior*) khi đang kết hợp chữ từ bàn đánh chữ cần được quy định thêm. Chúng ta có thể tạo hình cho mẫu tự chỉ sau khi việc kết hợp đã hoàn tất. Chúng ta cũng có thể tạo hình cho tất cả những dạng trung gian, hình của dạng sau được tạo ra bằng cách trở lui (*backspace*) để xóa rồi in chồng lên dạng trước. Mỗi cách đều có hữu dụng riêng như sẽ mô tả sau đây.

5.2.1 Cách Tạo Hình Lập Tức Trong Phép Tạo Chữ Ngầm

Phép tạo chữ ngầm được đặt ra để tiện cho việc xử lý những dữ kiện mà phần lớn là Việt ngữ. Với mục tiêu đó, người đánh chữ cần thấy ngay những chữ vừa đánh. Trong phép tạo chữ ngầm, bàn đánh chữ hoạt động trong trạng thái tạo hình lập tức. Mỗi phím chữ được nhấn (*keypress*) sẽ lập tức tạo ra một biến cố phím chữ (*key event*). Nếu một mẫu tự (`a`) kết hợp với một dấu phụ (`^`) theo sau nó, một thoái tự (*backspace*) (thường là BS, ASCII `0x8`) sẽ được gửi đi kèm theo sau là mẫu tự mới vừa thành lập (`â`). Chu kỳ này tái diễn cho đến khi việc kết hợp chữ hoàn tất. Trong cách tạo hình lập tức, những biến cố tạo ra do việc nhấn những phím chữ `"a^´n"` là:

1. Người dùng đánh `a`, `a` được gửi cho ứng dụng
2. Người dùng đánh `^`, `BS` và `â` được gửi đi
3. Người dùng đánh `´`, `BS` và `ấ` được gửi đi
4. Người dùng đánh `n`, `n` được gửi đi

Thoái tự có thể được thay đổi tùy theo ứng dụng, khiển hệ, và môi trường của người dùng. Bộ điều khiển của bàn đánh chữ nên dùng đúng thoái tự, và/hay cho phép người dùng chỉ định thoái tự theo ý thích.

5.2.2 Cách Tạo Hình Chậm Trong Phép Tạo Chữ Chỉ Định

Khi việc kết hợp chữ mới bắt đầu, bộ điều khiển của bàn đánh chữ không gửi cho ứng dụng từng biến cố phím chữ mà phải chờ cho đến khi việc kết hợp chấm dứt. Việc kết hợp có thể chấm dứt một cách tự nhiên khi chuỗi ký tự kết hợp đã đầy đủ hoặc khi bộ điều khiển nhận được ký tự không thể kết hợp được, hoặc chấm dứt khi nhận được mã tự hoàn cấu `<CLS>`. Sau đó, chỉ có một mẫu tự duy nhất được tạo thành và được gửi cho ứng dụng đang chờ. Việc xử lý sau đó sẽ tiến hành tự nhiên trở lại. Hãy xem diễn tiến khi người dùng nhấn chuỗi phím `"\a^´n"`:

1. Người dùng đánh `\`, không có chữ nào được gửi đi
2. Người dùng đánh `a`, không có chữ nào được gửi đi
3. Người dùng đánh `^`, không có chữ nào được gửi đi
4. Người dùng đánh `´`, chữ `ấ` được gửi đi
5. Người dùng đánh `n`, chữ `n` được gửi đi

Ví dụ sau đây dùng mã tự hoàn cấu `<CLS>`, `"t\o+<CLS>"`:

1. Người dùng đánh `t`, chữ `t` được gửi đi
2. Người dùng đánh `\`, không có chữ nào được gửi đi
3. Người dùng đánh `o`, không có chữ nào được gửi đi
4. Người dùng đánh `+`, không có chữ nào được gửi đi
5. Người dùng đánh CTRL-A, một chữ `ơ` được gửi đi

Để ý là nếu không có phím hoàn cấu `<CLS>`, bộ điều khiển bàn đánh chữ sẽ vẫn tiếp tục chờ sau khi phím `"+"` được bấm, vì người dùng vẫn còn có thể đánh một dấu giọng như là một phần tử của chuỗi kết hợp.

Phương pháp tạo hình chậm của phép tạo chữ chỉ định được đề ra để bảo đảm việc tích hợp với những ứng dụng đòi hỏi mỗi mã tự phải liên kết với một biến cố phím chữ, nhất là trong trạng thái Anh ngữ vì trạng thái này chỉ cho phép tạo chữ chỉ định mà thôi.

Mặc dù có thể tạo hình lập tức trong phép tạo chữ chỉ định hay tạo hình chậm trong phép tạo chữ ngầm, nhưng những cách thức này không hữu ích và chỉ làm cho người dùng lầm lẫn. Do đó, việc đơn giản nhất là chỉ liên kết cách tạo hình lập tức với phép tạo chữ ngầm, và cách tạo hình chậm với phép tạo chữ chỉ định. Những cách thức này có vẻ tự nhiên hơn.

Tiêu chuẩn trong văn kiện này quy định những đặc tính tối thiểu về mặt "hình thức và cảm giác" mà người dùng có thể kỳ vọng ở một nhu liệu Việt Ngữ hợp thức. Một giao diện được tiêu chuẩn hóa sẽ giảm thiểu thời gian phải học cho mỗi ứng dụng mới. Tiêu chuẩn trong bài này không loại bỏ những hệ thống đánh chữ khác mà mục đích của chúng là giúp người dùng dễ sử dụng hơn, chẳng hạn, đánh dấu bằng bảng-điều-khiển khôn ngoan, hay giúp đánh chữ nhanh bằng cách dùng các phím CONTROL hay FUNCTION chẳng hạn. Bất cứ sự gia công (*enhancement*) nào trong những ứng dụng hợp thức (*compliant application*) đều là một điều tốt cho người dùng, miễn là những sự gia công này không xung đột với những đặc tính tối thiểu mô tả trong bài này.

5.3 HỢP THỨC HÓA ỨNG DỤNG VIỆT NGỮ HIỆN HÀNH

Bất cứ phương pháp thực tế nào để định chuẩn cũng cần phải dự trù những sự chống đổi thay của các ứng dụng hiện hành. Trong khi mong muốn rằng tiêu chuẩn 8-bit trong bài này được ủng hộ hoàn toàn, chúng tôi cũng đề nghị một giải pháp khác dễ dàng được chấp nhận nhanh chóng hơn. Tất cả những ứng dụng cần phải cung cấp phương tiện để nhận vào và xuất ra những dữ kiện mã hóa theo tiêu chuẩn VISCII 8-bit. Đồng thời, những ứng dụng đó phải thực hiện một giao diện đánh chữ tuân theo VIQR, nếu không phải là phương pháp đánh chữ chủ yếu thì ít nhất cũng là một phương pháp phụ thêm cho người dùng. Những việc này rất cần thiết cho cả người dùng lẫn người bán. Người dùng có thể dùng nhu liệu ngay vì giao diện đánh chữ đồng nhất, cũng như có thể xử lý dữ kiện từ những ứng dụng khác nhau và trên những hệ thống máy khác nhau. Điều đó sẽ làm gia tăng năng suất và sự trao đổi giữa các người dùng. Việc dễ sử dụng sẽ khiến cho ứng dụng được chấp nhận rộng rãi hơn, và do đó người bán sẽ có nhiều khách hàng hơn.

6 TÓM TẮT & KẾT LUẬN

Văn kiện này vừa trình bày một dự thảo tiêu chuẩn hóa việc xử lý dữ kiện Việt Ngữ. Nhu cầu tiêu chuẩn hóa cũng đã được làm sáng tỏ. Chúng tôi mong rằng đã khuyến khích giới chế tạo nhu liệu và người dùng nhu liệu Việt ngữ cộng tác với nhau để đạt mục đích này hầu đem lợi ích đến cho tất cả mọi người liên hệ. Việc bàn luận những phương pháp mã hóa khác nhau đã đưa đến sự chọn lựa dự thảo VISCII 8-bit. Chúng tôi đã đề nghị một bảng mã tự duy nhất, và quá trình thử nghiệm thực tiễn cho thấy bảng này vận hành tốt đẹp cho Việt Ngữ qua các công việc như viết bài, xử lý, lưu trữ, chuyển tin, mã hóa phông chữ, và ấn loát. Trong những lãnh vực mà việc dùng 8-bit chưa cho phép hoặc không đáng tin cậy, chẳng hạn như việc chuyển điện thư, chúng tôi đã đề ra quy định Việt ngữ đọc-được-trong-ngoặc (VIQR) để cung cấp một cổng lọc suông sẻ. VIQR đã được quy định độc lập với nguồn xuất phát dữ kiện, do đó đã được thiết kế để có thể áp dụng được cho cả bàn đánh chữ tiếng Việt lẫn các máy lọc dữ kiện. Tất cả những điều này đã được chứng tỏ là có thể tích hợp vào những môi trường hiện hữu, giúp cho việc sử dụng những hư cụ và ứng dụng hiện hữu được trở thành dễ dàng hơn — một ưu điểm lớn của phương pháp mã hóa này. Cuối cùng, những quy định này đã được liên kết với nhau một cách suông sẻ trong mọi giai đoạn của chu kỳ xử lý dữ kiện (gồm có nhận dữ kiện, xử lý/truyền dữ kiện, và xuất ra dữ kiện, gọi tắt là chu kỳ nhập-biến-xuất). Những quy định này đã cung cấp một khuôn khổ thống nhất thực sự cho việc xử lý dữ kiện Việt Ngữ.

Tài Liệu Tham Khảo

[1] Bạch Hưng Khang. "Institute of Informatics,". Hà Nội, Việt Nam, tháng hai 1991.

[2] B. Jerman-Blažič, "Will the Multi-octet Standard Character Set Code Solve the World Coding Problems for Information Interchange?," *Computer Standards & Interfaces*, vol. 8, trang 127–136, 1988.

[3] The Unicode Consortium. *The Unicode Standard: Worldwide Character Encoding Version 1.0.* Addison-Wesley, Reading, MA, bản thứ nhất, tháng mười 1991.

[4] ISO Technical Committee, "Universal Multiple-Octet Coded Character Set (UCS), ISO/IEC DIS 10646-1.2," Draft standard, International Organization for Standardization, 1992.

[5] International Organization for Standardization. *ISO 8859/x: 8-bit International Code Sets.* ISO, 1977.

[6] Famjxuæn Thais. *Việt Ngữ Cải Cách.* Tứ Hải, Hà Nội, Việt Nam, tháng ba 1948.

[7] Phạm Xuân Thái. *Chữ Việt Hợp Lí.* Tín-Đức Thư-Xã, Sài Gòn, Việt Nam, tháng tư 1958.

[8] J. Postel, "Simple Mail Transfer Protocol," RFC 822, USC Information Sciences Institute, tháng tám 1982.

[9] J. C. Klensin et al., "SMTP Extensions for Transport of Text-Based Messages Containing 8-bit Characters," Internet draft, Massachusetts Institute of Technology, tháng bảy 1991.

[10] K. Simonsen, "Character Mnemonics & Character Sets," Internet draft, Danish Unix Users Group, tháng giêng 1992.

[11] K. Simonsen, "Mnemonic Text Format," Internet draft, Danish Unix Users Group, tháng tám 1991.

[12] International Organization for Standardization. *ISO 646: 7-bit Coded Character Set for Information Interchange.* ISO, bản thứ ba, 1991.

[13] International Organization for Standardization. *ISO 2022: 7-bit and 8-bit Coded Character Sets—Code Extension Techniques.* ISO, bản thứ ba, 1986.

[14] E. M. van der Poel, "Multilingual Character Encoding for Internet Messages," Internet draft, Software Research Associates, Japan, tháng giêng 1992.

[15] D.E. Knuth. *The TEXbook.* Addison-Wesley, Reading, MA, 1984.

THUẬT-NGỮ ANH VIỆT

Announcer: mã tự (hay chuỗi mã tự) báo tin. Khi mã tự này xuất hiện trong dòng dữ kiện thì nó báo cho biết những mã tự đi sau có một ý nghĩa đặc biệt. Trong văn kiện này, nó cho biết sự mở đầu của việc kết hợp chữ Việt.

ASCII: American Standard Code for Information Interchange, bộ mã tự tiêu-chuẩn Hoa-kỳ dành cho việc trao

dổi tin-tức. Bộ mã này có 128 mã số được hầu hết các máy vi-tính dùng để đặc trưng và truyền đi các dữ kiện chữ. Mỗi chữ trong bộ mã này có mã số trong khoảng từ 0 đến 127. Những bộ mã 8-bit hoặc 9-bit trong đó 128 mã tự đầu tiên tương ứng với ASCII được gọi là bộ mã ASCII-rộng (*extended ASCII*). Những mã tự thêm vào là mẫu tự La-tinh có dấu rời, mẫu tự phi-La-tinh, kiểm tự điều khiển màn ảnh, vân vân.

Backslash: gạch-chéo-ngược (\).

Base Vowel: nguyên âm cơ bản. Đứng trên phương diện mã hóa chữ Việt, văn kiện này coi những nguyên âm sau đây là cơ bản: `a ă â e ê i o ô ơ u ư y` và những chữ hoa tương ứng.

Binary Data: dữ kiện nhị phân; tùy theo ngữ cảnh còn mang nghĩa *dữ kiện 8-bit*, nhất là trong lãnh vực chuyển tin.

C0 Space: Vùng (miền) C0. Đây là tập hợp gồm những mã tự có số thập-lục phân từ `00` đến `1F` (vùng "kiểm tự" của bộ mã ASCII).

C1 Space: Vùng (miền) C1. Đây là tập hợp gồm những mã tự có số thập-lục phân từ `80` đến `8F` (vùng "kiểm tự" của bộ mã ASCII-rộng).

Character: mẫu tự, ký tự, mã tự. Trong tin-học, *character* thường được dùng để chỉ bất cứ cái gì được liên kết với một mã số (*code*) nên nghĩa đúng nhất là *mã tự*. Mã tự có thể là mẫu tự (như `a, b, c, ...`), hoặc dấu hiệu, ký hiệu (như `+, -, =, ...`), hoặc một tín hiệu điều khiển. Ký tự chỉ ký hiệu theo nghĩa rộng, bao gồm các dấu hiệu, mẫu tự, hoặc chữ tượng hình như chữ Hán.

Character Set: tập mã tự, bộ mã tự. Có thể dịch thoát là *hệ mã tự* vì mỗi tập mã tự là một *hệ* thống ký tự cho một hoặc nhiều ngôn ngữ. Cũng có thể dịch thoát là *bảng mã tự* vì thường thường tập mã tự được trình bày dưới dạng bảng. Số lượng mã tự trong mỗi tập là 2^n với n là số bit dùng để mã hóa một mã tự. Các hệ mã tự quen thuộc là bộ ASCII 7-bit của Hoa-kỳ, hệ 8-bit như các tập mã tự ISO-8859/X, hệ 16-bit như Unicode, hệ 32-bit như dự thảo ISO DIS 10646.

Code: mã số (trong thông tin dữ kiện), con số tượng trưng cho một mẫu tự, ký tự, hoặc tín hiệu điều khiển. Thí dụ số thập phân 65 trong bộ mã tự ASCII Hoa kỳ tượng trưng cho chữ `A`.

Code Page: thuật ngữ thường dùng để chỉ những tập mã tự dùng trên máy IBM PC, viết tắt là CP. CP850 là tập mã tự đa ngữ, CP860 là tập mã tự Bồ-đào-nha, CP863 cho tiếng Pháp ở Gia-nã-đại, CP865 cho Na-uy.

Code Page Switching: đổi bảng (tập, bộ, hệ) mã tự.

Compatible: tương hợp, tương dung

Compliant: hợp thức, tuân theo đúng cách.

Composed Character: chữ ghép, chữ rời (theo quan điểm mã hóa). Xem chữ `floating diacritic`.

Context-Dependent: tùy thuộc vào ngữ cảnh (ý nghĩa chung quanh).

Control Character: mã tự điều khiển, kiểm tự. Đó là mã tự ASCII nằm trong khoảng từ 0 đến 31, và mã tự 127, tương phản với những mã tự có thể in ra được (gọi là *ký tự hình*) nằm trong khoảng từ 32 đến 126. Trên các bàn chữ ASCII, kiểm tự (thí dụ `CTRL-A`, mã số 1) được tạo ra bằng cách chận phím CTRL xuống rồi đánh chữ liên hệ (`A`).

Cross-Platform: xuyên-giàn, xuyên qua nhiều hệ thống máy khác nhau.

Data: dữ kiện, dữ liệu.

Data Channel: mạch dữ kiện

Data Communication: thông tin dữ kiện, lãnh vực chuyển tin.

Data Frame: khung dữ kiện.

Data Integrity: sự toàn vẹn dữ kiện, sự bảo toàn dữ kiện.

Data Stream: luồng (dòng) dữ kiện, dòng tin.

Diacritic: dấu phụ. Dấu phụ là những nét thêm vào một mẫu tự "gốc" để tạo ra mẫu tự khác. Chẳng hạn mẫu tự `Â` được cấu tạo từ mẫu tự gốc `A` và dấu phụ `^`.

Display: Hiển thị, tạo hình, in hình (trên màn ảnh). Xem chữ *rendering*.

EBCDIC: Extended Binary Coded Decimal Interchange Code, bộ mã 8-bit gồm 256 mã tự dùng trên các máy IBM mainframes.

Editor: ứng dụng viết bài, viết-cụ (dụng cụ viết bài).

Electronic Mail: điện thư.

to Encode: mã hóa.

Escape Mechanism: cơ chế thoát.

Fax: điện hình thư. Khác với điện thư vì chỉ chuyển đi những chấm hình. Người nhận không thể dùng những ứng dụng viết bài để sửa đổi được.

File: hồ sơ. Có trường phái dịch là tệp.

Floating Diacritic: dấu rời (theo quan điểm mã hóa). Một mẫu tự có dấu phụ có thể được mã hóa bằng một mã số duy nhất hoặc nhiều mã số. Thí dụ `Ư` có thể được mã hóa bằng một mã số duy nhất và được gọi là chữ nguyên vẹn hoặc chữ dựng sẵn (*precomposed character*), hoặc mã hóa bằng hai mã số, một cho nguyên âm gốc `U` và một cho dấu móc (`ʾ`). Trong trường hợp sau, `Ư` được gọi là chữ ghép (*composed character*) và dấu móc được gọi là dấu rời.

Font: "phông," bộ kiểu chữ, một tập hợp các hình chữ có chung một số đặc tính nào đó và có thể in ra được trên màn ảnh hoặc trên giấy. *Italic font:* bộ chữ nghiêng. *Bold face font:* bộ chữ in đậm. Mỗi hình chữ (*glyph*) trong phông được ấn định một mã số. Phông 8-bit có thể có tất cả 256 hình chữ. Mã số của hình chữ (*glyph code*) không nhất thiết phải giống với mã số của chữ (*character code*) tương ứng trong tập mã tự. Thí dụ, dữ kiện chữ `A` có mã số 65 trong bảng ASCII nhưng hình chữ `A` có thể được quy định ở vị trí thứ 35 trong một bảng phông nào đó, nếu muốn. Nhưng một hình chữ trong phông lại có thể tương ứng với nhiều chữ trong tập mã tự. Điều này thường xảy ra trong phương pháp mã hóa dùng dấu rời (xem *floating diacritic*). Trong phương pháp này, chữ `À` thực ra là sự kết hợp của dữ kiện chữ `A` (mã số 65) và dữ kiện dấu huyền ` (thí dụ mã số 196), nhưng khi tạo hình thì dùng hình dạng của chữ `À`, thí dụ nằm ở vị trí 135 của bảng phông chữ. Điều này làm cho việc xử lý phức tạp và kém hiệu năng nên hầu hết các nhu liệu và cương liệu Âu Mỹ tránh không dùng. Do đó, để có thể tích hợp vào các môi trường nhu liệu hiện hữu, mã số hình chữ và mã số chữ phải giống nhau.

Font Shifting: chuyển phông, chuyển mẫu chữ.

Format: khuôn thức (khuôn khổ và hình thức).

Format Data Control Character: mã tự điều khiển khuôn thức.

Frame: khung, sườn.

Framework: khuôn khổ.

G0 Space: vùng ký tự hình (*graphic character*) có mã số thập-lục phân từ `20` đến `7F`.

G1 Space: vùng ký tự hình có mã số thập-lục phân từ `A0` đến `FF`.

Gateway: cổng, đầu cầu.

Glyph: hình chữ, một phần tử của phông chữ (*font*).

Graphic Character: ký tự hình (ký tự có thể in ra được).

Interface: giao-diện, mạch nối.

Interface Between 2 Computers: giao diện (mạch nối) giữa 2 máy.

ISO: International Organization for Standardization. Một ủy ban quốc tế tự nguyện gồm các ủy ban định chuẩn các nước hợp tác với nhau để định ra các tiêu chuẩn trong tất cả mọi lãnh vực, trong đó bao gồm máy vi tính, xử lý tin tức, tập mã tự.

ISO 646: Tiêu chuẩn mã tự 7-bit, tương tự với ASCII.

ISO Standard 8859: Tiêu chuẩn ISO 8859. Tiêu chuẩn này quy định một loạt các tập mã tự 8-bit bao gồm chữ của nhiều ngôn ngữ. Loạt này bao gồm những tập mẫu tự La-tinh 1–9, áp dụng cho tất cả những ngôn ngữ có chữ viết dựa trên mẫu tự La-mã, cộng thêm một số những tập mẫu tự đặc biệt như Cyrillic, Hy-lạp, và Do-thái.

ISO 8859/1: Tiêu chuẩn ISO 8859 Mẫu tự La-tinh số 1. Tối thiểu những ngôn ngữ sau đây sử dụng được: La-tinh, Đan-mạch, Đức, Dutch, Anh-ngữ, Faeroese, Phần-lan, Pháp, Băng-đảo, Ái-nhĩ-lan, Ý, Na-uy, Bồ-đào-nha, Tây-ban-nha, và Thụy-điển.

ISO 2022 and ISO 4873: Tiêu chuẩn ISO dùng để chuyển hệ mã tự.

ISO DIS 10646: ISO Draft International Standard, dự thảo tiêu chuẩn quốc tế ISO 16-bit và 32-bit cho tất cả các ngôn ngữ trên thế giới.

Latin: chỉ bộ mẫu tự La-tinh hoặc La-mã, gồm có các chữ từ A đến Z, hay tất cả các bộ mẫu tự dựa vào La-tinh.

Keyboard Interface: giao diện bàn đánh chữ.

Integrated Environment: môi trường tích hợp.

Integration: sự tích hợp, sát nhập.

Keyboard: bàn đánh chữ.

Line-Drawing: vẽ đường thẳng.

Literal: nguyên dạng, theo sát nghĩa đen.

Literal character: nguyên tự, chữ viết sao hiểu vậy.

Lower-Case Character: chữ in thường.

Look-and-Feel: hình thức và cảm giác.

Mail Agent: bộ chuyển thư, đại lý thư từ.

Mnemonic: dễ nhớ.

Modifier: Dấu phụ để thay đổi âm. Trong tiếng Việt, đó là các dấu trăng (˘), dấu mũ (ˆ), và dấu móc (˒).

On the Fly: theo lúc đó, ngay lúc đó.

PC: Personal Computer, máy vi tính cá nhân. Trong văn kiện này, chữ PC chỉ toàn bộ các máy IBM PC và PS/2 cùng những máy tương hợp, kể cả máy AT, 286, 386 và 486.

PostScript: Một ngôn ngữ mô-tả từng trang (giấy, sách, báo, v.v.) một, có khả năng xử lý đồ-hình (*graphic capabilities*), dùng trong việc ấn loát bằng máy vi-tính. Đây là một ngôn ngữ thảo chương cấp cao và độc lập với mọi thiết bị. PostScript là nhãn hiệu cầu chứng của công ty Adobe Systems Incorporated.

Precomposed Character: chữ nguyên vẹn, chữ dựng sẵn (theo quan điểm mã hóa). Xem chữ `floating diacritic`.

Processing: xử lý.

Protocol: biên bản, nghi thức

Public-Domain: thuộc lãnh vực công cộng.

Quoted-Readable: đọc được trong ngoặc. Từ xưa, ý nghĩa của những mã tự đặc biệt có thể thay đổi bằng cách bỏ chúng trong ngoặc đơn hoặc ngoặc kép, hoặc dùng một ký tự báo tin đi trước như dấu gạch chéo ngược (\) dùng trong khiển hệ Unix hay ngôn ngữ thảo chương `C`. Do đó xuất hiện từ "trong-ngoặc" (*quoted*). Cơ chế ngoặc được gọi là *quoting mechanism*. Thí dụ trong khiển hệ Unix, ký tự * được khai triển thành những ký tự khác, nên mệnh lệnh

```
rm *
```

sẽ xóa hết mọi hồ sơ, trong khi đó mệnh lệnh

```
rm "*"
```

chỉ xóa một hồ sơ có tên là ký tự *. Tương tự, nếu cơ chế ngoặc được quy ước là `<cd>`, trong đó `c` là nguyên âm, `d` là dấu phụ, thì ta có thể diễn tả chữ `cả` bằng `c<a?>`, vì chuỗi `a?` có nghĩa là `ả` khi nằm trong ngoặc. Quy định VIQR trong văn kiện này dùng những ký tự gợi hình như ´`?˜. để tượng trưng cho dấu giọng nên dễ đọc và dễ nhớ.

Real Time: thực thời. Trong lãnh vực điện toán, chỉ việc xử lý xảy ra đồng bộ với biến cố thực sự.

Rendering: sự hiển thị, sự tạo hình, in hình (lên màn ảnh hoặc giấy). Xem chữ *display*.

Sequence: chuỗi, loạt.

Software: nhu liệu. Có trường phái dịch thoát là *hư liệu* (hư là hư thể, không có thực).

Software Application: nhu liệu ứng dụng, hoặc dịch tắt là ứng dụng.

Software Library: thư viện nhu liệu. Có thể dịch thoát là hư-viện.

Software Tool: nhu liệu dụng cụ, hư cụ.

Source Code: chương trình gốc (bản gốc của chương trình để cho bộ dịch (*compiler*) đọc và dịch ra ngôn ngữ máy).

Specification: quy định.

Table-Lookup: tra bảng.

TEX: Một hệ thống ấn loát được điện toán hóa do Donald Knuth [15] phát triển, có khả năng đáp ứng mọi nhu cầu ấn loát các ký hiệu toán học và các văn bản với phẩm chất cao. TEX là nhãn hiệu cầu chứng của Hội Toán-học Hoa-kỳ.

Text: văn bản.

Text Viewer: ứng dụng nhìn chữ. Thường dùng để duyệt qua trên màn ảnh máy vi tính xem văn bản hiện ra như thế nào trước khi in ra giấy.

Trade-off: việc chọn lợi hại.

Transmission: chuyển, truyền.

Transparent: thông suốt, vô hình, không dấu vết. Ở đây dùng để chỉ việc xử lý ở những cấp bậc dưới có thể khác nhau nhưng vẫn làm cho giao diện ở cấp bậc cao hơn không thấy có gì thay đổi hoặc phân biệt được.

Unicode: Một bộ mã tự 16-bit do liên-đoàn công ty Unicode Consortium thiết lập. Ấn bản *(versio*n) 1.0 [3] được phát hành năm 1991 bao gồm hầu hết các ngôn ngữ trên thế giới, trong đó có các chữ La-tinh, Tây-Âu, Đông Âu, Việt-nam, Hy-lạp, Nga, Do-Thái, Ả-rập, chữ tượng-hình Hán Nhật Đại-hàn, v.v. Trong ấn-bản 1.0, chữ Việt-nam được mã hóa theo phương pháp dùng dấu rời (*floating diacritic*). Tin bán chính thức cho biết khi tiêu chuẩn Unicode sát nhập với ISO DIS 10646, chữ Việt-nam sẽ được mã hóa theo phương pháp chữ nguyên.

Unix: Một hệ thống điều khiển (*operating system*) máy vi tính rất phổ thông, được phát triển ở trung tâm nghiên cứu AT&T Bell Laboratories. Rất nổi tiếng vì có thể vận hành trên nhiều hệ thống máy khác nhau.

Upper-Case Character: chữ hoa.

Usenet: một mạng lưới thông tin điện toán quốc tế cho phép người dùng máy vi-tính gửi tin tới những người khác và những người này có thể phúc đáp. Việc tham gia vào mạng lưới Usenet cũng giống như việc đăng ký xem những tạp chí điện toán (*electronic magazine*, tạp chí hiện diện trong mạng lưới điện toán, không phải là tạp chí in trên giấy). Những tạp chí này, gọi là nhóm-tin (*news-group*), chuyên về rất nhiều chủ đề khác nhau. Nhóm-tin "Soc.Culture.Vietnamese" rất phổ thông trong giới người Việt và ngoại quốc ở khắp nơi trên thế giới.

User: người dùng, người sử dụng.

User-Interface: giao-diện cho người dùng.

Utility Software: nhu liệu dụng cụ, dụng liệu (nhu liệu hữu dụng). Những ứng dụng để trợ giúp việc phát triển nhu liệu như nhu liệu viết bài (*editor*), nhu liệu tìm chữ (grep, awk trong khiển hệ Unix), nhu liệu để điều khiển in nhiều bài.

Vietnamese Character Code: bộ mã chữ Việt, bộ Việt-tự-mã.

Vietnamese Character Encoding Standard: bộ Việt-tự-mã tiêu chuẩn.

Viet-Std: Một nhóm vô vụ lợi gồm các chuyên viên Việt-nam ở hải ngoại hợp tác để định ra các tiêu chuẩn về cương-liệu và nhu liệu cho chữ Việt. Các thành viên trao đổi ý kiến qua điện thư và các cuộc họp.

Variant: dị-bản

Word Processing: xử lý chữ.

Word Processor: máy xử lý chữ.

Word Processing Software: nhu liệu xử lý chữ.

Phụ Lục A: Mẫu Tự Việt Liệt Kê theo Thứ Tự Sắp Chữ

Chữ	VIQR	VISCII
A	A	065
Á	A´	193
À	A`	192
Ả	A?	196
Ã	A~	195
Ạ	A.	128
Ă	A(	197
Ắ	A(´	129
Ằ	A(`	130
Ẳ	A(?	002
Ẵ	A(~	005
Ặ	A(.	131
Â	A^	194
Ấ	A^´	132
Ầ	A^`	133
Ẩ	A^?	134
Ẫ	A^~	006
Ậ	A^.	135
B	B	066
C	C	067
D	D	068
Đ	DD†	208
E	E	069
É	E´	201
È	E`	200
Ẻ	E?	203
Ẽ	E~	136
Ẹ	E.	137
Ê	E^	202
Ế	E^´	138
Ề	E^`	139
Ể	E^?	140
Ễ	E^~	141
Ệ	E^.	142
F	F	070
G	G	071
H	H	072
I	I	073
Í	I´	205
Ì	I`	204
Ỉ	I?	155
Ĩ	I~	206
Ị	I.	152
J	J	074
K	K	075
L	L	076
M	M	077

Chữ	VIQR	VISCII
N	N	078
O	O	079
Ó	O´	211
Ò	O`	210
Ỏ	O?	153
Õ	O~	160
Ọ	O.	154
Ô	O^	212
Ố	O^´	143
Ồ	O^`	144
Ổ	O^?	145
Ỗ	O^~	146
Ộ	O^.	147
Ơ	O+	180
Ớ	O+´	149
Ờ	O+`	150
Ở	O+?	151
Ỡ	O+~	179
Ợ	O+.	148
P	P	080
Q	Q	081
R	R	082
S	S	083
T	T	084
U	U	085
Ú	U´	218
Ù	U`	217
Ủ	U?	156
Ũ	U~	157
Ụ	U.	158
Ư	U+	191
Ứ	U+´	186
Ừ	U+`	187
Ử	U+?	188
Ữ	U+~	255
Ự	U+.	185
V	V	086
W	W	087
X	X	088
Y	Y	089
Ý	Y´	221
Ỳ	Y`	159
Ỷ	Y?	020
Ỹ	Y~	025
Ỵ	Y.	030
Z	Z	090

Chữ	VIQR	VISCII
a	a	097
á	a´	225
à	a`	224
ả	a?	228
ã	a~	227
ạ	a.	213
ă	a(	229
ắ	a(´	161
ằ	a(`	162
ẳ	a(?	198
ẵ	a(~	199
ặ	a(.	163
â	a^	226
ấ	a^´	164
ầ	a^`	165
ẩ	a^?	166
ẫ	a^~	231
ậ	a^.	167
b	b	098
c	c	099
d	d	100
đ	dd	240
e	e	101
é	e´	233
è	e`	232
ẻ	e?	235
ẽ	e~	168
ẹ	e.	169
ê	e^	234
ế	e^´	170
ề	e^`	171
ể	e^?	172
ễ	e^~	173
ệ	e^.	174
f	f	102
g	g	103
h	h	104
i	i	105
í	i´	237
ì	i`	236
ỉ	i?	239
ĩ	i~	238
ị	i.	184
j	j	106
k	k	107
l	l	108
m	m	109

Chữ	VIQR	VISCII
n	n	110
o	o	111
ó	o´	243
ò	o`	242
ỏ	o?	246
õ	o~	245
ọ	o.	247
ô	o^	244
ố	o^´	175
ồ	o^`	176
ổ	o^?	177
ỗ	o^~	178
ộ	o^.	181
ơ	o+	189
ớ	o+´	190
ờ	o+`	182
ở	o+?	183
ỡ	o+~	222
ợ	o+.	254
p	p	112
q	q	113
r	r	114
s	s	115
t	t	116
u	u	117
ú	u´	250
ù	u`	249
ủ	u?	252
ũ	u~	251
ụ	u.	248
ư	u+	223
ứ	u+´	209
ừ	u+`	215
ử	u+?	216
ữ	u+~	230
ự	u+.	241
v	v	118
w	w	119
x	x	120
y	y	121
ý	y´	253
ỳ	y`	207
ỷ	y?	214
ỹ	y~	219
ỵ	y.	220
z	z	122

† Quy-định VIQR còn cho phép tượng trưng "Đ" bằng "Dd" hoặc "dD". Xin xem Phần 4.2.1.

Phụ Lục B: Mẫu Tự Việt Liệt Kê theo Thứ Tự Mã Số Thập Phân.

VISCII	Chữ	VIQR	Tên Anh-ngữ
002	Ả	A(?	A breve hook-above
005	Ã	A(~	A breve tilde
006	Ẫ	A^ ~	A circumflex tilde
020	Ỷ	Y ?	Y hook-above
025	Ỹ	Y ~	Y tilde
030	Ỵ	Y .	Y dot-below
065	A	A	A
066	B	B	B
067	C	C	C
068	D	D	D
069	E	E	E
070	F	F	F
071	G	G	G
072	H	H	H
073	I	I	I
074	J	J	J
075	K	K	K
076	L	L	L
077	M	M	M
078	N	N	N
079	O	O	O
080	P	P	P
081	Q	Q	Q
082	R	R	R
083	S	S	S
084	T	T	T
085	U	U	U
086	V	V	V
087	W	W	W
088	X	X	X
089	Y	Y	Y
090	Z	Z	Z
097	a	a	a
098	b	b	b
099	c	c	c
100	d	d	d
101	e	e	e
102	f	f	f
103	g	g	g
104	h	h	h
105	i	i	i
106	j	j	j
107	k	k	k
108	l	l	l
109	m	m	m
110	n	n	n
111	o	o	o

VISCII	Chữ	VIQR	Tên Anh-ngữ
112	p	p	p
113	q	q	q
114	r	r	r
115	s	s	s
116	t	t	t
117	u	u	u
118	v	v	v
119	w	w	w
120	x	x	x
121	y	y	y
122	z	z	z
128	Ạ	A .	A dot-below
129	Ắ	A(´	A breve acute
130	Ằ	A(`	A breve grave
131	Ặ	A(.	A breve dot-below
132	Ấ	A^ ´	A circumflex acute
133	Ầ	A^ `	A circumflex grave
134	Ẩ	A^ ?	A circumflex hook-above
135	Ậ	A^ .	A circumflex dot-below
136	Ẽ	E ~	E tilde
137	Ẹ	E .	E dot-below
138	Ế	E^ ´	E circumflex acute
139	Ề	E^ `	E circumflex grave
140	Ể	E^ ?	E circumflex hook-above
141	Ễ	E^ ~	E circumflex tilde
142	Ệ	E^ .	E circumflex dot-below
143	Ố	O^ ´	O circumflex acute
144	Ồ	O^ `	O circumflex grave
145	Ổ	O^ ?	O circumflex hook-above
146	Ỗ	O^ ~	O circumflex tilde
147	Ộ	O^ .	O circumflex dot-below
148	Ợ	O+ .	O horn dot-below
149	Ớ	O+ ´	O horn acute
150	Ờ	O+ `	O horn grave
151	Ở	O+ ?	O horn hook-above
152	Ị	I .	I dot-below
153	Ỏ	O ?	O hook-above
154	Ọ	O .	O dot-below
155	Ỉ	I ?	I hook-above
156	Ủ	U ?	U hook-above
157	Ũ	U ~	U tilde
158	Ụ	U .	U dot-below
159	Ỳ	Y `	Y grave
160	Õ	O ~	O tilde
161	ắ	a(´	a breve acute
162	ằ	a(`	a breve grave
163	ặ	a(.	a breve dot-below

Phụ Lục B: Mẫu Tự Việt Liệt Kê theo Thứ Tự Mã Số Thập Phân (tiếp theo).

VISCII	Chữ	VIQR	Tên Anh-ngữ
164	ấ	a^´	a circumflex acute
165	ầ	a^`	a circumflex grave
166	ẩ	a^?	a circumflex hook-above
167	ậ	a^.	a circumflex dot-below
168	ẽ	e~	e tilde
169	ẹ	e.	e dot-below
170	ế	e^´	e circumflex acute
171	ề	e^`	e circumflex grave
172	ể	e^?	e circumflex hook-above
173	ễ	e^~	e circumflex tilde
174	ệ	e^.	e circumflex dot-below
175	ố	o^´	o circumflex acute
176	ồ	o^`	o circumflex grave
177	ổ	o^?	o circumflex hook-above
178	ỗ	o^~	o circumflex tilde
179	Ỡ	O+~	O horn tilde
180	Ơ	O+	O horn
181	ộ	o^.	o circumflex dot-below
182	ờ	o+`	o horn grave
183	ở	o+?	o horn hook-above
184	ị	i.	i dot-below
185	Ự	U+.	U horn dot-below
186	Ứ	U+´	U horn acute
187	Ừ	U+`	U horn grave
188	Ử	U+?	U horn hook-above
189	ơ	o+	o horn
190	ớ	o+´	o horn acute
191	Ư	U+	U horn
192	À	A`	A grave
193	Á	A´	A acute
194	Â	A^	A circumflex
195	Ã	A~	A tilde
196	Ả	A?	A hook-above
197	Ă	A(	A breve
198	ả	a(?	a breve hook-above
199	ã	a(~	a breve tilde
200	È	E`	E grave
201	É	E´	E acute
202	Ê	E^	E circumflex
203	Ẻ	E?	E hook-above
204	Ì	I`	I grave
205	Í	I´	I acute
206	Ĩ	I~	I tilde
207	ỳ	y`	y grave
208	Đ	DD†	D bar
209	ứ	u+´	u horn acute

VISCII	Chữ	VIQR	Tên Anh-ngữ
210	Ò	O`	O grave
211	Ó	O´	O acute
212	Ô	O^	O circumflex
213	ạ	a.	a dot-below
214	ỷ	y?	y hook-above
215	ừ	u+`	u horn grave
216	ử	u+?	u horn hook-above
217	Ù	U`	U grave
218	Ú	U´	U acute
219	ỹ	y~	y tilde
220	ự	y.	y dot-below
221	Ý	Y´	Y acute
222	ỡ	o+~	o horn tilde
223	ư	u+	u horn
224	à	a`	a grave
225	á	a´	a acute
226	â	a^	a circumflex
227	ã	a~	a tilde
228	ả	a?	a hook-above
229	ă	a(	a breve
230	ữ	u+~	u horn tilde
231	ẫ	a^~	a circumflex tilde
232	è	e`	e grave
233	é	e´	e acute
234	ê	e^	e circumflex
235	ẻ	e?	e hook-above
236	ì	i`	i grave
237	í	i´	i acute
238	ĩ	i~	i tilde
239	ỉ	i?	i hook-above
240	đ	dd	d bar
241	ự	u+.	u horn dot-below
242	ò	o`	o grave
243	ó	o´	o acute
244	ô	o^	o circumflex
245	õ	o~	o tilde
246	ỏ	o?	o hook-above
247	ọ	o.	o dot-below
248	ụ	u.	u dot-below
249	ù	u`	u grave
250	ú	u´	u acute
251	ũ	u~	u tilde
252	ủ	u?	u hook-above
253	ý	y´	y acute
254	ợ	o+.	o horn dot-below
255	Ữ	U+~	U horn tilde

† Quy-định VIQR còn cho phép tượng trưng "Đ" bằng "Dd" hoặc "dD". Xin xem Phần 4.2.1.

Announcement

of

VISCII-Compliant Software Applications

Release 3 – September 1992

by

The TriChlor Group

Announcement of VISCII-Compliant Software Applications - Release 3

San Jose, California, September 1992
The TriChlor Group
Email: trichlor@haydn.stanford.edu

This release includes major support for Microsoft Windows 3.1,
and a move to VISCII 1.1. It has been an unusually productive
period for TriChlor and TriChlor-Talk members. Many thanks to
everyone for their time and efforts. New software added in
this release is:
   winvnkey, vnfont1, vn-nhac, butviet, more bdf fonts for X,
   vietxrn, vprint, thitap1.

With the newly added Windows and TrueType supports, one can
generate "beautiful" and "professional" publications easily,
such as DDa(.c San, Ba'o Xua^n, Bi'ch Ba'o, Sa'ch ....
And data files are identical across Unix, DOS and Windows
platforms, with the same keyboard entry style.
If you use TriChlor software to generate these publications,
we would be glad to receive a compliment/souvenir copy.

Most of TriChlor software provides a foundation upon which other
Vietnamese dedicated applications can be built.
Following is a short description of all programs.
All are Viet-Std VISCII 1.1 & VIQR compliant.
Unless otherwise specified, most software is written/ported
by Cuong T. Nguyen (cuong@haydn.stanford.edu)
and Cuong M. Bui (bui@berlioz.nsc.com)

For UNIX & X-Window:

include: Include files for software developers.

lib: Library routines for software developers.

fonts: Contains X-Window bitmap fonts, METAFONT sources for
TeX, pre-made TeX bitmap .pk fonts, and scalable outline
Type-1 fonts.

METAFONT can be used to generate different point sizes
and typefaces for TeX bitmap .pk fonts.

```
        X-Window .bdf fonts include:
          vn10x20.bdf, vnlucida18      [NEW]
                   designed by Thu V. Vu (tvu@sg102a.ess.harris.com)
          vn9x15.bdf, vn-r14.bdf.

        Type-1 fonts: include all the TrueType fonts in vnfont1
        archived under VN/WINDOWS

vietterm:
        A modified version of X-Window xterm. It allows
        entry and display of Vietnamese characters, at the same time
        stays compatible with US-ASCII environment, and uses existing
        8-bit clean Unix tools. Changes include 8-bit Viet-Std
        compliance, support for both Athena and MOTIF widgets, radio
        buttons to switch/display keyboard and screen modes.

vnterm:
        A modified version of X-Window xterm. It allows entry
        and display of Vietnamese characters, at the same time stays
        compatible with US-ASCII environment and uses existing 8-bit
        clean Unix tools. Changes include 8-bit Viet-Std support.

vn7to8:
        Filters for conversion between 8-bit Viet-Std (VISCII)
        and 7-bit Viet-Std (VIQR), for printing, posting or mailing.
        SCV articles and Viet-Net mails (7-bit) can be printed by
        converting them to 8-bit Viet-Std.
           Algorithm by Hoc D. Ngo (ngo@nas.nasa.gov)

vnpstext:
        Prints 8-bit Viet-Std compliant text files to PostScript
        printers. Uses PostScript Type 3 Courier font.
        Both regular and bold typefaces are supported.
        Default to font size 10. Prolog files can be modified
        for different font size.
        Also available for DOS.

vietxmail:
        A modified version of X Window xmail. Used together
        with "vnmailtool" to allow one to read/write 8-bit
        Vietnamese chars in 7-bit channel mail transparently.
        Updates include 8-bit Viet-Std VISCII codes.
        Allows choice of different mail-agents (sendmail) or filters.
        Needs utilities from "vnmailtool" to work.

vnmailtool:
        A set of conversion filters, when set up properly,
        will process Viet-Std mail messages from 8-bit VISCII
```

to 7-bit VIQR and vice-versa transparently.
Allows choice of different mail-agents (sendmail)
or filters. Can be used with vietxmail or vietterm/vnterm.
Also included are viet7to8 and viet8to7 for conversion
between 8-bit VISCII and 7-bit VIQR for posting or mailing.

vn_editmsg:
Vietnamese filter for Unix mail, rn, Pnews under X.
Requires vnterm or vietterm, vnelvis, and vn7to8. Allows
you to edit your mail messages in Vietnamese and send
them off after automatic conversion to 7-bit VIQR.
Similar capability for rn and Pnews. Implemented as a
Unix shell script.

vnelvis:
A modified "elvis1.4" ("vi" editor clone), tuned for 8-bit
Viet-Std & C0 space characters. Needs vietterm or vnterm.
Works with most Unix systems. Also available on DOS.
Needs vietdos on DOS.

vpp8:
A modified version of vpp, a troff preprocessor, used to
print 8-bit VISCII files. Original program written by Hu+~u,
ported to C by Nha^n Tra^‘n (tran@peora.sdc.ccur.com), ported
to 8-bit VISCII by TriChlor. Support for Italic added.
troff is standard on most Unix systems. Scalable fonts.

vnfortune:
Same as Unix "fortune" program. Displays Vietnamese poems,
tu.c ngu+~, ca dao ... randomly. Users can add or modify
the database. Database consists of around 100 phrases,
contributed by SCV and Vietnetters, most notably,
Nguye^~n Thi. Ca-Dao (ca-dao@sjsuvm1.bitnet)
Author: Qua^n Tra^‘n (qdt00@duts.ccc.amdahl.com)

locale.sun:
VISCII locale for SUN running OS 4.1. When installed will
allow direct use of Sun’s "vi", "ls" and other tools
that support locale to work also on 8-bit VISCII
Vietnamese chars.

vnless:
A modified version of "less" to work with 8-bit VISCII.
Very powerful pager.

vietsc:
A modified "sc", a popular public domain Unix spreadsheet.
Includes mortgage calculation examples. May need vncurses.

vncurses:
Vietnamese capable curses library.
Unix's curses from BSD 4.3 or SVR3 is not-8 bit clean.

lambai:
A sample application for teaching Vietnamese. Teachers
can generate coded quiz in Vietnamese. Students answer
in either English or Vietnamese.
Also available on IBM PC.

vietxrn:
Vietnamese capable version of xrn 6.17. For posting, reading
and replying using Vietnamese 8-bit Viet-Std. Automatic,
transparent conversion to Viet-Std 7-bit VIQR for posting and
mailing. To compose and print in 8-bit, needs vietterm or
vnterm, vnelvis or locale, and vpp8 or vnpstext. Supports
both MOTIF and Athena widgets.

For IBM-PC WINDOWS
==========
winvnkey: [NEW]
Allows entry of Vietnamese chars in Microsoft Windows 3.1
applications. Works with most existing Windows
applications such as Word, Notepad, Paintbrush, ...
Viet-Std compliant.

vnfont1: [NEW]
Set of 4 Vietnamese TrueType fonts, VISCII encoding.
HeoMay (vn-helvetica)
HoangYen (vn-present-script)
MinhQuan (vn-courier)
UHoai (vn-utopia)
From these fonts, Windows can generate many font sizes and
variations such as Italic, Bold, and Bold-Italic.

monmoi: [NEW]
Adds Vietnamese description to icon. Window's File Manager
has trouble doing this.

thitap1: [NEW]
Electronic poetry collection of 11 poems by Quang Du~ng.
Displays and prints poems using Windows' Help facility.

For IBM-PC DOS
==============

vietdos:
Allows entry and display of Vietnamese chars. Works with
most existing softwares such as vi, vnelvis, pctools, xtree,
MS Edit, WordPerfect, brief, Paradox, Word, C, Pascal, ASM...
Requires EGA or VGA. Uses vprint or vlaser to print.

vnelvis:
"elvis1.4", a "vi" editor clone, tuned for Vietnamese.
This is the same version of vnelvis on Unix, ported to DOS.

viet78:
filters for 7-bit (used by SCV, vietnet) to 8-bit
VISCII. Also from 8-bit VISCII to 7-bit.

less:
public domain "less" pager, better than "more", with
environment variable for charset tuned for 8-bit Viet-Std

lambai:
same version of lambai on Unix, ported to DOS.
Dedicated for teaching/testing using Vietnamese language.
Requires vietdos.

vlaser2:
printing 8-bit Viet-Std Vietnamese on laser printers.
Supports 4 typefaces.
Author: Ho‘a Nguye^~n (nguyenh@cod.nosc.mil)
Also author of viet1p6, a Vietnamese word processor.

testdata.std:
A file of VISCII-1.1 characters for testing software.
Author: Ho‘a Nguye^~n (nguyenh@cod.nosc.mil)

vn-nhac: [NEW]
A colorful program that display collection of Vietnamese
songs, menu driven. Currently, there are four volumes
consisting of about 200 songs
contributed by SCVers and Viet-Netters.
Author: Ho^‘ Phu+o+’c Hu‘ng (hho@aludra.usc.edu)

butviet: [NEW]
A Vietnamese editor, Viet-Std VISCII and VIQR
compliant, can be used with CGA, Hercules, EGA, VGA.
Has multi-windows and mouse support.
Author: Nguye^~n Doa~n Vu+o+.ng (vnguyen@adobe.com)

```
vprint:          [NEW]
         VISCII-1.1 printer driver for Epson and IBM dot-matrix
         printers.  Can be invoked from a pipe.
            Author: Nguye^~n Doa~n Vu+o+.ng (vnguyen@adobe.com)

convert 1.3:     [NEW]
         Converts files in one Vietnamese coding format to a dozen
         other Vietnamese coding formats, including VISCII and VIQR.
            Author: Vietnamese Professional Society
                    (Ho^.i Chuye^n Gia Vie^.t Nam)
                    Email: hcgvn@netcom.com

Work in progress
================
vietvu: similar to winvnkey, can also use clipboard besides
        message hook.
         Author: Vu~ V. Cha^u (chauv@microsoft.com)

thitap: a bookshelf of poetry volumes, currently has
        thitap1 (Quang Du~ng). Will have thitap2 (Nguye^~n Bi'nh),
        thitap3 (Nhie^'u ta'c gia?) ...
        Many poems were posted by Viet-Netters and SCV-ers.
        Needs volunteer editors! Contact bui@berlioz.nsc.com if
        interested.


Proposed projects (Need volunteers!)
====================================

TrueType fonts: Additional fonts are needed. Should be public domain,
         re-distributable.

text2speech: to encode Vietnamese speech and implement text to
         speech conversion for Vietnamese text files.

multimedia application:

vietspell: a dictionary-less speller based on lex/yacc.


Getting the software
====================
There is an archive site at sonygate for anonymous ftp, or
if you don't have ftp, there is a mailserver at saigon.com.
Pre-compiled binaries for IBM-PC and  Sun 4/SPARC  are also
available via floppy disks. Binaries are compiled under Sun OS 4.1,
X11R4, OpenWindow 3.0, Motif 1.1.
```

```
Using anonymous FTP
====================

Here are the instructions for accessing the anonymous FTP archive.

Site name: sonygate.Sony.Com [192.65.138.2]
                  (try 192.65.138.240 if the first one doesn't work)
Login as anonymous, password is anything
                  (please use your email name).

  % mkdir VN          # where you'll be compiling everything
  % cd VN
  % ftp sonygate.sony.com
  Connected to sonygate.sony.com
  220 sonygate.sony.com FTP server ready.
  Name (sonygate.sony.com:<your_login>): anonymous
  Password (sonygate.sony.com:anonymous): <type_your_name>
  331 Guest login ok, send ident as password.
  230 Guest login ok, access restrictions apply.
  ftp> cd tin/VN
  250 CWD command successful.
  ftp> type image  # (or "binary")
  200 Type set to I.
  ftp> get <file>.tar.Z
  ftp> quit

  If you have question/problem accessing the archive, please contact
        Tin Le (tin@smsc.sony.com)

Using modem or mailserver at Saigon.COM
==============================
From: tin@smsc.sony.com (Le TinTin)
Subject: [TECH] Instruction for accessing Viet files
Date: Tue, 4 Aug 92 00:19:56 GMT

The mailserver at Saigon.COM has been updated to contain the latest
files from the anon FTP archive at sonygate.  This is for those of
you without FTP access.  You can retrieve the files through email
requests.

You may also dial up Saigon.COM directly and download the files.
The front end to it is Station Zebra, a BBS.  It is located in the
San Francisco Bay Area.  The phone number is 408-739-1520.  Login
with user id of bbs.
```

```
[ MailServer Instructions ]

Vietnet and Viet software MailServer archive

This is the MailServer archive for Vietnet and Vietnamese software.
It is a duplicate of the anonymous FTP archive.  The purpose of this
mailserver is to allow those without ftp access to retrieve these
files.

The MailServer looks for its instructions in the body of the message.
Subject header is ignored.  Some of the instructions are:

  LIST directory_name
  DIR directory_name
  CD directory_name
  GET filename
  HELP

Please make sure that your email address in the header of the
message is good.  The MailServer uses that to send the files to you.
We've gotten a lot of bounced messages from those of you with
unreachable email address.

Here is an example session:

  $ mail mailserver@saigon.com
  Subject: <anything here>
  DIR viet.net
  GET viet.net/vnterm.tar.Z
  EOF (type a Control-D)

The email address of the server is mailserver@saigon.com.
Some of you who are in the UUCP domain might have problem sending
mail to an Internet style address.
You could try the following bang paths:

  uunet!sonyusa!szebra!mailserver
  claris!szebra!mailserver
  zorch!szebra!mailserver

Using modem to phsys.com
========================
 VIETNET BBS - phsys.com (PH. Systems)
      Phone No: 1-213-734-8528
                (Modem: Hayes V32-V42, 8N1, 300-38400bps)
      Login: NEW  (then fill out questionaire for account)
      Owner: Hu'ng P. Ho^', hung@phsys.com
             (Los Angeles, California, USA)
```

Mail Server will be in operation 2nd week of Oct 92,
please mail to mail-server@phsys.com for more info

The BBS is a 24-hour public system with Vietnamese newsgroups, Job listings, limited Usenet News, mail connections to many other nets including Internet. Archive site of all Vietnamese PD software and technical documents from Viet-Std, Vietnamese Professional Society, TriChlor, MAT/CAT etc.. Graphics section has been updated with *.gif from the Viet Nam War era.
You are welcome to login and download software/gif. Please leave me a message/mail so I'll give you full access.

Using Post Office
==========
For DOS/Windows send $5.00 to cover the set of 2 floppy disks, and specify your disk type (5-inch disk or 3.5-inch disk). Setup and installation are automated, executables and files will be put in proper place and proper Windows group.

For Sun Sparc/Sun4/X-Windows binaries, send $10.00 to cover the set of 2 floppy disks which contain the following programs: vietterm, vnpstext, vpp8, vietxrn, vnelvis, locale, ... and bdf fonts.

Make check payable to: Cuong T. Nguyen (acting accountant)
Please send to:

TriChlor Software
3388 Burgundy Drive
San Jose, CA 95132

We definitely are not going to profit from this. The $5 covering mailing expenses doesn't come anywhere close to representing the huge effort everyone, named and unnamed, has put into this, both in terms of time, keyboard stress, head strain, and out-of-pocket expenses to buy compilers, font converters, hardware, etc. to bring VISCII/VIQR compliant software to the Vietnamese community.

If you like what we are doing and would like to to help us meet expenses---which are quite real and considerable---your contribution in any amount is always welcome, but is never required.

[For an experimental period, precompiled binaries for a select set of TriChlor software will be available via anonymous FTP to Haydn.Stanford.EDU (36.22.0.47). Although limits are not enforced, please try to schedule your access between 7PM - 6AM Pacific time. Depending on the resulting load on the system, this service may become permanent, off-hours, or altogether removed.

```
  Binaries are currently available for DecStation 3100, Sun-3,
  and Sun-4.  Knowledgeable users with binaries compiled for
  these and other platforms are encouraged to make them available
  to other netters.  Trichlor-binaries@haydn.Stanford.EDU
  is a mailing list for that circle of volunteers who participate
  in this distribution system.  If you have binaries and are
  interested in helping, please send a "SIGN-ON" message to
  trichlor-binaries-request@haydn.Stanford.EDU. ]


Copyright
=========
All software products are copyrighted by respective authors or
the TriChlor group. All rights reserved.
You may copy, distribute, and use these software for any purpose,
as long as you do not charge a fee for doing so, and
include this copyright notice.  All software is provided "as is",
without express or implied warranty.

All software is provided free of charge as copyrighted freeware
(that means you can have it for free and make as many copies for
as many friends as you like, but we reserve the copyright in order
to prevent unauthorized modifications and to prevent people from
selling it to unsuspecting users---we want to keep the freeware
free).

About the TriChlor group
========================
TriChlor is a group of volunteers whose common interests are to
provide quality and free utilities, codes, libraries for use with the
Vietnamese language, and to integrate Vietnamese language to current
computing environments.

If you would like to pitch in and help, we can be contacted at
    trichlor@haydn.stanford.edu
If you wish, you may get on the mailing list and participate in
discussions by sending a message to
    trichlor-talk-request@haydn.stanford.edu
 with the subject
    "SIGN-ON".
But please be prepared to donate your time :-).
All works, unless specified, will be placed on public domain.



The TriChlor Group
```