DESIGN OF INFORMATION SYSTEMS FOR ARABIC

Dr. Patrick A.V. Hall
Ismael A. Hussein
SCICON International
P.O. Box 3883
RIYADH
KINGDOM OF SAUDI ARABIA

ABSTRACT

The design of information systems for the handling of Arabic text is considered. The final objective is assumed to be the storage of information in Arabic, with the requirement later to retrieve the information using Arabic.

Initially, those aspects of Arabic which are of concern in the design of computer based information systems are surveyed. Arabic script is examined for its division into words and its letters and their various forms. Particular attention is paid to the use of spaces and decorative bars. Names of particular importance in many information systems, and Arabic names are examined in depth.

Then systems design in all its aspects is considered. Peripheral design and the design of internal codes are first considered, weighing the alternative choices against the impact these would have on the retrieval algorithms. Then the various algorithms for searching using indexes, comparing text for identity, and comparing for sorting are examined. Also algorithms of secondary concern, searches for appropriate matches, are discussed.

Finally, it is argued that codes and character forms should be standardised. Recommendations can be made concerning the handling of Arabic text and Arabic names, but it is not proposed that these should be the basis for standardisation.

1. INTRODUCTION

This paper began with an anecdote. We were told a story of a group of European computing experts who built a demonstration system for Arabic users - it would store, in Arabic, a person's name, and some associated information, and enable the person to retrieve the associated information later using the person's name as a key. The first Arab they tried, was called Ali (علي): he typed his name and other information, then details of his friends. He then switched across to retrieval to recover the information about himself - and failed ! What had gone wrong ? Had the computer broken down, or the software developed on bug ? Nothing of the sort: an Arab was using a computer system which had been designed on non-Arabic principles. Ali had typed his name with a flourish the first time, as علي , but left out the flourish the second time, typing على . In Arabic this variation is natural, in European languages, it is unthinkable.

We took the anecdote to heart. Maybe we as European technologists were in danger of building into our computer system too many assumptions about European languages, and not recognising the differences that exist in Arabic.

So we started to look at Arabic without making assumptions to discover what Arabic was really like, and to design afresh a computer system for Arabic. One of the authors is a specialist in computing, but being European necessarily has all sorts of in-bred assumptions about the nature of language and computer information systems which may be invalid for Arabic. The other author is an Arab and a specialist in linguistics. Together we hope we have been able to sow the seeds of a fully Arabic Computer System.

In section 2, we survey the aspects of Arabic that seem particularly relevant to computer systems. In section 3, we then examine the design of computer systems for Arabic, discussing the options available from the internal codes, to the peripherals, to the data capture and retrieval mechanisms. Finally in section 4 we examine systems design from the point of view of standardisation.

2. SURVEY OF ARABIC

2.1 Arabic characters and Text

European languages have developed a script in which text can either be cursive as in normal handwriting, or it can be non-cursive as in most printing and typing. In cursive handwriting all letters within a word join together with breaks between words. In non-cursive printing or typing, letters within a word are packed close together, while between words pronounced spaces occur.

By contrast, Arabic is always cursive in both handwriting and printing. Some letters join together, other do not. Breaks in the cursive flow can occur within words, and must occur between words.

Our primary concern in investigating Arabic is to discern how Arabic would "naturally" be typed into the computer by untrained personnel, and how it should be printed for easy reading by untrained personnel. Then, in designing our computer system we must arrive at a balance between making the computer handle Arabic naturally, and training the users to work effectively within the limitations imposed upon Arabic by the computer system.

2.1.1 Character Form

Arabic letters vary in shape in accordance with their position within a word. While in general letter forms vary a lot (eg. 6) it is useful to adopt the approach of recognising four basic potential forms of letter - initial, medial, final or isolated, depending upon the position of the letter in a cursive sequence of letters. Cursive sequences correspond to words or fragments of words.

Arabic letters are of two types. Some of them can join the following letters and are

called "inseparable letters" (Mitchell's terminology). Others never join the following letter and are called "separate letters". Each of the inseparable letters may appear in one of four forms. Separate letters can appear only in two forms. Figure 1 shows the different forms of Arabic letters together with their pronunciation in English.

There is the tendency to use only two forms of the inseparable letters on typewriters and printers, the initial and the isolated forms, and only one form of the separate characters - the isolated.

Let us consider one particular typewriter in detail. Figure 2 shows the different forms of the Arabic letters, as they appear on this typewriter. In addition, this typewriter has the diacritic symbols, shadda, madda and hamza.

Recently there have been several attempts to simplify the Arabic characters, using only one form, the initial form, for each letter (see the paper by Almajid, 1978). These developments are clearly of great interest to the computing profession and should be followed closely.

Written Arabic basically records the consonants of the language, together with the long vowels. The short vowels are added by the reader, but can be written using diacritic marks. There are several other diacritics in use in Arabic. In writing and printing, these diacritics are usually omitted unless they are necessary to avoid ambiguity. On typewriters, and some printing presses, it is only possible to insert some of the diacritics, and the practice is to insert a parenthetical remark to indicate a missing diacritic if this is necessary.

2.1.2 The Extension Bar

In Arabic handwriting (see Figure 3 (a)) it has become accepted practice to be able to extend the line joining adjacent letters at the discretion of the calligrapher. This idea has been carried through into typed and printed Arabic, a special extension bar character being used for this purpose. Figure 3(b) illustrates.

This extension bar is used to enhance readability, to justify the text being written on both the left and the right, or for artistic purposes. This extension bar has no linguistic significance at all.

2.1.3 Widths of Characters

The Arabic characters do not have equal widths. Some characters could take a space five times as big as others. On most typewriters two character widths are adopted.

For example, on one particular typewriter about forty percent of the forms of the letters are given single space widths (1/16 inch) while the other sixty percent of the forms of the letters are given double space widths (1/8 inch).

However, some characters run together, so that on occasions the characters must be separated by the appropriate use of the extension bar or the space character.

FIGURE 1. The different forms of the Arabic Letters, as used in printing (from Mitchell and Barber, 1972)

	isolated	final	medial	initial	English symbol aa/aa (')	name 'alif (hamza) (see below)
1.	ا	ا			b	baa'
2.	ب	ب	ب	ب	t	taa'
3.	ت	ت	ت	ت	th	thaa'
4.	ث	ث	ث	ث	j	jiim
5.	ج	ج	ج	ج	H	Haa'
6.	ح	ح	ح	ح	kh	khoo'
7.	خ	خ	خ	خ	d	daal
8.	د	د	د	د	dh	dhaal
9.	ذ	ذ	ذ	ذ	r	roo'
10.	ر	ر	ر	ر	z	zaay
11.	ز	ز	ز	ز	s	siin
12.	س	س	س	س	sh	shiin
13.	ش	ش	ش	ش	S	Saad
14.	ص	ص	ص	ص	D	Daad
15.	ض	ض	ض	ض	T	Taa'
16.	ط	ط	ط	ط	DH	DHaa'
17.	ظ	ظ	ظ	ظ	9	9ayn
18.	ع	ع	ع	ع	gh	ghayn
19.	غ	غ	غ	غ	f	faa'
20.	ق	ق	ق	ق	q	qaaf
21.	ك	ك	ك	ك	k	kaaf
22.	ل	ل	ل	ل	l	laam
23.	م	م	م	م	m	miim
24.	ن	ن	ن	ن	n	nuun
25.	ه	ه	ه	ه	h	haa'
26.	و	و	و	و	w; uu	waaw
27.	ي	ي	ي	ي	y; ii	yaa'
28.	ى	ى	ى	ى		

FIGURE 2. The forms of the Arabic Letters appearing on an OLIVETTI LINEA 98 Arabic Typewriter.

	isolated	final	medial	initial	English Name
1.	ا				alif
2.	ب	ب	ب	ب	baa'
3.	ت	ت	ت	ت	taa'
4.	ث	ث	ث	ث	thaa'
5.	ج	ج	ج	ج	jiim
6.	ح	ح	ح	ح	Haa'
7.	خ	خ	خ	خ	khaa'
8.	د	د			daal
9.	ذ	ذ			dhaal
10.	ر	ر			raa'
11.	ز	ز			zaay
12.	س	س			siin
13.	ش	ش			shiin
14.	ص	ص			Saad
15.	ض	ض			Daad
16.	ط	ط			Taa'
17.	ظ	ظ			DHaa'
18.	ع	ع	ع	ع	9ayn
19.	غ	غ	غ	غ	ghayn
20.	ق	ق	ق	ق	faa'
21.	ك	ك	ك	ك	qaaf
22.	ل	ل	ل	ل	kaaf
23.	م	م	م	م	laam
24.	ن	ن	ن	ن	miim
25.	ه	ه	ه	ه	nuun
26.	و	و	و	و	haa'
27.	ي	ي	ي	ي	waaw
28.	ى	ى	ى	ى	yaa'

2.1.4 The Space Character

Keyboard devices, such as typewriters, are equipped with a space character. Under what circumstances is this used in Arabic? We have already seen in the last section that the space character may be used to enhance legibility, to prevent letters running into each other. But we would also expect the space character to be used for the separation of words, a grammatical use of the space. For the moment we take the concept of word as understood - the concept is explored further in section 2.2.1.

In handwriting, Arabic is written without leaving significant gaps between words with only breaks in the flow being breaks in the joining of successive letters, either between words or within words. Figure 3(a) illustrates. In printing, however, it is normal to leave significant gaps between words, as seen in Figure 3(b). In typing the practice seems to be somewhere in between; the space character is, or should be, used between words, but the gap created may not appear significant visually.

From the linguistic point of view, however, gaps between words are not necessary, these being purely graphically devices inserted to improve the appearance and readability of the text.

A problem related to that of the insertion of gaps into text, is the continuation of text onto the following line. Where can the text be broken in order to continue onto the following line? In English the rule is either between words, or within a word between syllables with the use of a hyphen. In Arabic, the rule is only to break a line of text at the division between words.

2.1.5 Arabic Numbers

Arabic numbers are used the same way as the English numbers, with units on the right. Normally we write Arabic numbers in the same order that we read them, as we do in English, that is we start from the most significant digit. However, when transcribing a number, one would proceed from right to left.

In Arabic, the decimal point is written with a comma (as in German), using the letter 'raa' for this purpose and not the Arabic comma. However, when numbers exceed a thousand, the numerals can be grouped with the groups separated again by a 'comma'. Thus numbers can be ambiguous, and the ambiguity must be resolved by the context in which the number appears.

2.1.6 Dates

Two systems of dates are used in the Arab world. The Gregorian system used in Europe and America is in general use, but additionally the traditional Islamic Hajjerian calendar is used, especially in Saudi Arabia.

The Hajjerian system of dating is based upon the phases of the moon, there being 12 lunar months per Hajjerian year. This means that the Hajjerian year is slightly shorter than the Gregorian year which is based upon the phases of the sun. The Hajjerian new year falls approximately ten days earlier each Gregorian year.

FIGURE 3. Samples of Arabic Text: (a) handwritten (b) printed . In both cases note the alignment left and right by extending the line joining adjacent letters, and note the absence of clear gaps between adjacent words. Both passages have been repeated in (c) and (d) to mark the word breaks with a solidus.

(a) handwritten, from a newspaper advertisement

علمًا بأنه لن يتم تمديد هذه الفترة
مرة أخرى، وسوف يجري بيع أسهم
المتخلفين بعد خمسة عشر يومًا من

(b) printed,

بهذه المناسبة ينبغي ان نذكر بشعور الامتنان العميق وعظيم التقدير لما يقدمه لنا صاحب
السمو الملكي الرئيس العام لرعاية الشباب وسمو نائبه الكريم من توجيهات وارشادات واهتمام بالغ
بتشجيع الشباب السعودي من اجل احياء وتطوير الفنون التشكيلية وايجاد المناخ الفني الملائم
لمسيرة النهضة الكبرى التي تشمل مملكتنا الحبيبة .

(c) handwriting showing the word breaks

علمًا/بأنه/لن/يتم/تمديد/هذه/الفترة/
مرة/أخرى/،/وسوف/يجري/بيع/أسهم/
المتخلفين/بعد/خمسة/عشر/يومًا/من/

(d) printing showing the word breaks.

بهذه/المناسبة/ينبغي/ان/نذكر/بشعور/الامتنان/العميق/وعظيم/التقدير/لما/يقدمه/لنا/صاحب/
السمو/الملكى/الرئيس/العام/لرعاية/الشباب/وسمو/نائبه/الكريم/من/توجيهات/وارشادات/واهتمام/بالغ/
بتشجيع/الشباب/السعودى/من/اجل/احياء/وتطوير/الفنون/التشكيلية/وايجاد/المناخ/الفنى/الملائم/
لمسيرة/النهضة/الكبرى/التي/تشمل/مملكتنا/الحبيبة/.

FIGURE 4. Examples of Arabic Words. The word is a compound object, and nouns have attached to them the article, prepositions, possessive pronouns, and the 'word' "and".

people	ناس	book	كتاب
the people	الناس	his book	كتابه
with the people	بالناس	like his book	كتابه
like the people	كالناس	and his book	وكتابه

Since both systems are defined in terms of astronomical phenomena, we would expect a very exact correspondence between the dates of one system and the dates of the other system. But this is not so. Each calendar system is concerned for the correspondence between two astronomical phenomena - days (i.e. rotations of the earth on its axis) and lunar or solar periods. Both systems must regulate themselves so that a whole number of days fit into the larger astronomical period on which they are based. The Gregorian system regulates itself by the prediction of astronomical events, with the number of days per year varying on a way which is intended to maintain the calendar in step with the phases of the sun. By contrast the Hajjerian system regulates itself by astronomical observation, with the onset of successive months being determined by the observation of the new moon. Occasionally the expected number of days in a month has to be adjusted: the adjustment may only be announced after a few days, then becoming retroactive to the beginning of that month.

The nett effect of these different methods of regulating the calendar is that an exact correspondence between predicted Hajjerian dates and predicted Gregorian dates is not possible, though correspondence to within a day is predictable. For historical dates a precise correspondence between the Gregorian calendar and the Hajjerian calendar is possible, but this correspondence will not be possible by calculations based solely upon astronomical laws. Rather a system of tabulated dates with interpolation between using astronomical laws is necessary. However, for many purposes an exact correspondence is not necessary, and in this case a correspondence based solely upon astronomical laws can be used, giving an error of one or two days at most. Note, however, that for the period of any retroactive adjustment there will be some uncertainty concerning the precise date recorded on documents - was the date recorded, adjusted, or not ?

2.2 Some Aspects of Grammar

2.2.1 What is a word ?

The notion of a word in Arabic is relatively complex, since an addition to an Arabic word can become a part of that word. Look at the examples of Figure 4: the most surprising example is probably the inclusion of "and" into the word that follows.

2.2.2 Consonantal Roots

Most Arabic words are made by adding different prefixes or suffixes to what is called the root of a word. This root is often made of three consonants. (eg. Tritton).

Figure 5 shows some examples of words derived from a single root. Note that in some variations, consonants actually occur between the consonants of the root.

2.2.3 Alphabetical Order for Arabic Words

The letters of the Arabic alphabet are traditionally ordered as in Figure 1, starting with Alif and ending in yaa. To this must be added the rule that hamza precedes alif, and that the compound character laam-alif ل ا is treated as a laam follo-

FIGURE 5. Examples of words derived from the consonantal root ع ق د

To assemble	اعتقد
To make a contract	تعاقد
To believe	اعتقد
Knot	عقدة
Braid weaver	عقّاد
Lieutenant Colonel	عقيد
Contractor	عقيد
	معاقد
	متعاقد
Faith, belief	عقيدة
Bunch	عقود
Belief	اعتقاد
Dogma, principle	معتقد
Complicated, complex	معقد
Pacreas	معقد
Tied, bound	معتود

FIGURE 6. Examples of Arabic names, to illustrate the variety of problems that might be encountered when analysing names.

(a) Ismael Abdelfatah Hussein Salih	اسماعيل عبد الفتح حسين صالح
(b) Ahmad Hassan Asseeri	أحمد حسن عسيري
(c) Abu Ja'far Mohammed ibn Musa Al Khowarizmi	أبو جعفر محمد بن موسى الخوارزمي
(d) Suraj Aldin Abd Rub Alnaby Saud Alkabeer	سراج الدين عبد رب النبي سعود الكبير
(e) Mohammed Hafez Mansour	محمد حافظ منصور

wed by an alif.

Traditionally all Arabic words in an Arabic dictionary are ordered in lists in accordance with the alphabetical order of their consonantal root. So when looking up the word

MAKTUUB مکتوب

we cannot find it under meem because we have to extract its consonantal root which is
K T B ك ت ب

so we will find it under K - T - B.

For telephone directories and similar works it is already established practice to arrange these alphabetically in the European manner, and there have been some attempts to do this for Arabic dictionaries.

2.3 Arabic Names

It is worth considering how names are used in an information system. Names are given to people as a label with which to refer to the person; but names are more than just a label, they tell you something about the person. They tell you the nationality or ethnic and linguistic group to which the person belongs, and by analysing the name, the names will tell you the sub-group within the ethnic group from where the person comes - his family title or clan, the area in which he was born, his father's and grandfather's names, perhaps even his mother's name. Thus a person's name carries with it a lot of information, an important aspect of the manipulation of names in an information system is the analysis of the name and the division of it into its information components. You want to be able to answer the question "who is this person?"

It is worth distinguishing between an individual's legal name and his informal names. With modern administration society people acquire a legal name at birth, and this name can only be changed by properly constituted legal processes. The most common of these is marriage, but there are processes in most administrations whereby a person can voluntarily change his or her name, and whereby titles can become incorporated in the person's legal name.

2.3.1 Examples of Arabic Names

Consider the examples shown in Figure 6, where Arabic original and English transliterations are shown together.

The traditional structure of Arabic names is a declaration of lineage; each child is given a personal name, and the person's full name consists of his personal name followed by that of his father then his grandfather, and so on. The first example of Figure 6 is the full name of one of the authors recording that Ismael's father was Abdelfatah, grandfather was Hussein, and so on. Names such as these can go back for as many generations as desired. The connective "bin" (often also transliterated

as "ibn" or "ben") meaning 'son of' is frequently used, as in the second example.

It is becoming increasingly common today for Arabs to use a family-name, presumably as a result of contact with the west. This family name can either be a statement of the place or origin; example (b) is Ahmad son of Hassan from the Asseer province of Saudi Arabia, while example (c) is Mohammed son of Musa from the village of Khowarizm. The family name is usually written last, and often prefixed by an "Al" (or "El") as in example (c). Most commonly the family name appears to have been acquired by choosing one of the names in the sequence of lineage (typically the last out in the sequence written down - thus the author Ismael of this paper has on occasions been known as Mr. Hussein, and Mr. Salih). In formal administration systems one is typically asked for four names, the last of which is taken as the family name. The names in the Riyadh telephone directory are sequenced by family name: about half these have the prefix "Al".

There are some bemusing anomalies to the use of "Al". The name "Sheikh" also occurs in the form "Al sheikh" - and hence the family name "Al Alsheikh". However, when faced with the family name "Alsheikh" is the "Al" a prefix for the family name or not? Sometimes a famous ancestor is singled out as the giver of the family name: this could be the father, with further names written after it. Furthermore, sometimes the name can be prefixed by 'Al', but not be a family name. There is also some variability in the way the prefix 'Al' is used. It is usually treated as a simple prefix, being joined to the family name, but in some cases can be treated as a separate word.

People are often simply known as the son of their father, as in "Ibn Saud". This can also happen with the mother's name, for example, "bin Kabina" (Thessiger). In passports in the Arab world it is usual to include the mother's name (only the given name), as an extra aid to the identification in cases where otherwise the names would be the same.

It is quite common for people to become known as "father of" as in example (c). Traditionally this was more than just a nickname, for the person becomes known by this name and the names "Abu Ja'far" etc. will be quoted in the lineage and in modern usage can become part of the family name. In current usage however the Abu name is a nickname in situations where courtesy is called for. In various areas of the Arab world there are alternatives to "Abu" - for example, "Aba", "Ba", or "Bo". For women an equivalent "mother of" nickname can be used.

The prefix "Abu" in example (c) is an example of a comparatively frequent occurrence - a single name composed of several words. The most common version of this is those names starting with "Abd" meaning "servant of". Example (d) shows a name of three components but seven words.

As a final example, an added complexity occurs occasionally where a person is given a double name as in Example (e): the father Mansour gave his son the double name "Mohammed Hafez" and the son is known variously as "Mohammed", as "Hafez", as Moh-

ammed Haffez".

Most of the phenomena noted above do also occur in European names. Thus in European names it is not always obvious which is the family name especially since frequently this is written first. It is also common for names to consist of several words, especially for familynames (e.g. "van der Merwe"), while double names are also frequent (e.g. Jean-Louise). In European names one also has the extra complication of abbreviations. Currently abbreviations in Arabic are infrequent, and do not happen with names, excepting that prefixes such as "Abd" are sometimes dropped in European circles.

Note that in Islamic tradition at marriage women do not adopt the name of their husband, but retain their own name with their own paternal lineage. However in European influenced Islamic societies, at marriage women do change their name.

2.3.2 How Titles are used in Arabic

It is not customary for Arab people to address each other using their pure names. We have already seen the use of "Abu", and in formal situations one would insert a title into the name of the person addressed.

Normally titles precede the name they qualify. Such are titles like AMIR (prince), SHEIKH, SHERIF (nobleman), DOCTOR, USTAZ (teacher, professor) and HAJ (HAJJJI). Only AMIR, DOCTOR AND SHERIF always accompany the name of the person.

The confusing titles are SHEIKH and USTAZ. Although USTAZ is basically used for teacher at an educational centre, it can be used for any educated man. The title SHEIKH is still more confusing. Originally it was used for the head of a bedouin tribe, but nowadays the word SHEIKH is used for middle-aged (or older) people who are usually very rich and enjoy a very high rank whether official, commercial or social.

A military title is given by a very high authority, and one cannot change it. Even when pensioned off one will still have the right to use it as one's title (usually it is denoted that the involved person is retired). These military titles become an integral part of the person's legal name.

Sometimes a man would get a title after his profession. In such a case this title is added to the end of the name. Here we cannot guarantee that the title will appear with the person's name on all occasions. Examples of such titles are: najjar (carpenter), Haddad (blacksmith), LahHam (butcher) and kattan (cotton dealer).

There are a few titles which have become rather archaic nowadays. These are titles like Basha, Beik and Afandi. These titles used to be given to government employees, Basha being the highest then Beik and the last is Afandi. Such titles are usually placed directly after the first name of a person, and the first name plus title were treated as the full name of the person.

3. ARABIC TEXT IN THE COMPUTER

In the design of a computer system to handle Arabic, we have several related decisions to make. We must design the peripherals, displays and printers, and decide in the external representation of the Arabic characters on each of the devices. We must decide on the internal representation of the Arabic script. And we must also design the algorithms for use in the software for matching, searching and sorting.

Some discussions of these problems have been made in the literature (e.g. the papers of Tawakol (15) and of Seflan (14) focussing primarily on the problems associated with the character sets offered by one particular manufacturer).

3.1 The Coding of Arabic Text

We wish to store inside the computer a representation of Arabic text. But what is Arabic text ? Is it a sequence of letters divided into words, or is it a sequence of characters, letters in their various forms, with no distinct words, or what is it ? The answer to this will determine how we construct our code for Arabic. Do we code letters only, leaving the conversion to the various forms to a transformation just prior to printing, or do we code the individual forms of the letter ?

We could choose one distinct code per letter, and generate the various external forms of characters automatically - the particular form depends at most upon the letters on either side, assuming that a special symbol is always used to separate words. The generation of correct printed form would necessarily be expensive. Note that this transformation could actually happen in software if some other code was used for interchange between peripheral and computer, but would most sensibly be done in a micro-computer in the peripheral. With a single code per letter, matching for identity is trivial. Ehlers (6) has reported a system for computer typesetting of Arabic with an input keyboard for letters rather than forms and an output character set of many hundreds of forms.

As the alternative to a code per letter, we could choose one code for each printed form of each character. With this strategy, there is the cost of added complexity of matching and comparing in the computer, though this cost is small. With this course there could be an interest (a disadvantage) in reducing the number of forms, since the number of codes available might be limited. The character set may have to accommodate other sets such as Latin capitals. We would then have to decide how many forms of each letter are necessary and what the widths of all the forms are. Having too few forms leads to loss of legibility, but having too many permits alternatives choices for a given position (e.g. medial and initial) and thus extra complexity in matching.

Independently of the decision to code letters or forms is the decision concerning fixed or variable length codes.

If a variable length code was used, it could be convenient to make the lengths of the

code (measured in bits or bytes) proportional to the length of the printed character. This would then have the advantage that space requirement for printing a string of text would be proportional to the internal storage requirement. The price for this is a small increase in complexity of matching and comparison algorithms; a single byte by byte comparison from the right hand end still works, but different forms must be recognised as equivalent and only if the different forms were different lengths would any significant extra complexity arise. However, this variable length strategy would be paid for in extra internal storage requirements, possibly leading to the use of too few multiple length forms and degraded legibility.

Another reason for moving to variable length codes could be that the number of codes of fixed length available is less than the number of characters or forms to be represented. In this case it would be sensible to choose the longer codes on the basis of minimising storage for less frequent codes. This would most likely be for non-Arabic characters and an idea worth considering is automatic shift and unshift characters for sequence of non-Arabic characters. One can readily envisage an Arabic - ASCII code to complement the current Latin-ASCII, where in the Latin - ASCII there is a shift-to-Arabic code, while in Arabic - ASCII there is a shift to the Latin code. This strategy is (almost) open-ended, and codes for other scripts could easily be added.

Using a fixed length code with a variable length external representation should, in fact, give remarkably little difficulty, though it would be thought novel for those used to the usual Latin peripherals. In a simple storage and retrieval application, one would capture the data in some predetermined space - and would be able to retrieve to that same size space. What goes in comes out. Now however there is uncertainty about how much internal storage will be required for a given amount of printing space, but an upper bound is easily estimated. There should be little or no storage cost and if an estimate of printing space is required for a given internal string, it is readily calculated by scanning the text. To assist with alignment of columns in printout of tables, a TAB facility on all output peripherals would be very useful.

In all cases, the code sort orders as binary numbers should be the same as the normal Arabic character sort order, with some convention concerning different forms, for example isolated before final before medial before initial.

A problem of some significance is concerned with the diacritics. Which of these are coded? Of those that are coded, would the code for the diacritic precede or follow that of the letter which it qualifies? The answer to this second question may well depend upon consideration of peripheral design, addressed in the next section. If diacritics are included, we may have complications in the matching algorithms since in some cases the use of diacritics is optional.

3.2 Peripheral Design

Arabic text stored inside the computer needs to be displayed or printed for the user.

It is also necessary to be able to obtain information in Arabic from the user. The peripheral devices for this input and output need to be designed: the part of their design that is of interest here is the form the Arabic text takes on paper or visual display screen, and the keyboard used for input.

We assume here that the discourse between peripheral and computer will use the code discussed in section 3.1. We saw there how the design of this code depended upon considerations of what would be printed - how many different forms of characters are really necessary and how their width would vary.

Arabic characters vary so tremendously in width in normal writing and printing (3 to 5 times) that at least two widths must be considered. Having recognised this inevitability, it is necessary to decide on the division of the characters into single and double lengths. The considerations here are legibility and printing density. If too many characters are forced into single lengths, the basic lengths of these must increase or legibility will suffer. This leads to reduced packing, with the text requiring more space than necessary upon the screen or paper. Note that there is an alternative course of action: it might be possible to accommodate all characters in a single width if suitably redesigned, though this may lead to too large a departure from tradition to be acceptable.

Other considerations of significance are diacritics and the TAB facility discussed in the last section. Keyboard layout is also important, and clearly a layout similar to that of typewriters is desirable.

3.3 Coding Forms

When Arabic is typed from a hand-written original, the exact form of the outcome is unpredictable, since the typist has a lot of discretion about layout, and extension bars and spaces can be introduced, and spaces may even be omitted. This same thing could happen during the entry of data into a computer system, unless we take steps to avoid it.

Given a background of European computing it seems reasonable to use coding forms, with each character written in a separate box, so that a precise character by character specification will be given. But, coding forms for Arabic may not be a reasonable facility to use.

As has been described above, Arabic uses a cursive script in which characters vary up to five times in width. There is no hand-written block printing, and even in learning to write Arabic, one progresses almost immediately to the writing of cursive words. Thus a coding form for Arabic would have to be filled in using cursive writing: to see that this has problems, imagine filling in a coding form in cursive English!

We have conducted a small trial using a standard English coding form for Arabic text, and found that the tradition of extending the separation between characters by using

a horizontal bar is now an asset, and placing letters firmly into boxes is easy apart from four very broad letters (Saad, Daad, Seen and Sheen) which seem to demand two boxes ! For these letters, we would need to adopt special conventions of writing (as is done for some Latin characters, such as zed, to avoid confusions), so that a one box one keystroke identity can be preserved.

In the context of the design of a single complete computer system, the introduction of coding forms with rigid blocks for letters could involve a considerable cost in training and in error - correction post data-entry. The alternative is to accept the traditions of written and printed Arabic and arrange for the computer to handle the traditional form of Arabic. It will be seen in the following sections that the cost in software is by comparison very small.

Of course, the use of coding boxes for numerical data in Arabic is perfectly reasonable.

3.4 Matching and Sorting

Let us start by considering the comparison of two strings of Arabic characters. Let us take as a basic algorithm, a character by character comparison starting from the right. Then a simple-minded method for comparison would be to say the strings are identical if each pair of characters were identical, while they would be different if one or more pairs were different (or if one had more characters than the other) with the sort order being determined by the rightmost non-matching character. This is precisely the method we would use based on experience with European languages, but does it work for Arabic ? We must be clear what we are trying to achieve when comparing two text strings. We are not simply looking for superficial resemblances, we are asking whether they are the same strings as sequences of symbols in the Arabic language.

If the coding of Arabic text is at the abstract level, with letters coded and grouped together as words, then already the nature of Arabic has been taken into account, so the simple-minded comparison methods described above will suffice. However, the coding methods that we meet will almost invariably be an encoding of letter forms, as discussed in section 3.1.

The most obvious variability in Arabic words occur with the optional use of the extension bar and space characters. We could avoid this problem by imposing upon the user a discipline in the use of these symbols, so that he always uses them in the same way. At its simplest, this discipline could take the form of disabling the keys for these characters so that they can never be input. This may be acceptable for the extension bar, but for spaces we would most likely have to train him in the 'correct' use of the space character. The alternative is to allow free use of these characters, so that text can be formatted as desired by the user, but take compensatory action in software. What we want to do is to compare the strings of text without any extension bars (or spaces, if we allow freedom with those). The comparison algorithm could

either work directly upon the original strings, skipping any bars or spaces, or one or both strings could be reduced to canonical form by stripping all bars and spaces before making the comparison. If indexes are used, then the index must be built from the canonical form with all extraneous characters removed.

It is possible in typed or printed Arabic for different forms of the same letter to be used in the same position in the same word: this can only happen with particular letters, usually where the initial form can be used in a medial position. There are two ways out: either the peripherals and code could not distinguish between the forms, or the matching algorithms could take this into account. Since there is very little loss in legibility by removing the medial form, and a lot to gain in matching simplicity, it is clearly worth making this simplification in the character set.

If variable length codes are used (for example, some letter forms involving two or more bytes of code, or shift characters present), no further complications arise for exact matches, but for sorting comparisons extra complications can arise because of the variable length, especially if different forms of the same letter have different lengths.

It could happen that in performing a match, in searching for a particular word, one wants to find all grammatical variants of a particular word. To do this in English, you would use the stem or root of the word: for example to match the words "computer" and "computation" you would use the stem "comput-". In English it is adequate to provide the facility to match a string embedded in another string, since the grammatical variants almost always occur with prefixes and suffixes. Unfortunately, in Arabic the grammatical variants often insert extra letters into the middle of the root of the word, as well as adding suffixes and prefixes. (See section 2.2.2). Note that within the words derived from a root, the letters of the root can appear in any of their possible character forms and long vowels can actually change into other long vowels. Note also that it is not possible to automatically construct the consonantal root of an Arabic word from the word itself. Thus the user would need to supply the root, or the system would extract all possible candidates, and the user would have to tolerate the retrieval of many nonsense matches.

This discussion of consonantal root matching is one aspect of a more general system's requirement, that of retrieval of information given partial information, or imprecise information. The simplest form is the use of a key, or several keys, which are used to retrieve information. But often the exact form of the keys is not known - hence the consonantal root method. But partial knowledge manifests itself in other ways, and in section 3.6 on names, we will see partial knowledge in the form of parts of names, and variations in transliteration.

3.5 Text Analysis

Often it happens that it is necessary to analyse a piece of text and break it up into

its constituent parts. This takes us into the realms of linguistic analysis, and while it is not our intention to discuss this subject in depth, nevertheless it is necessary to say something if only as background to the examination of names made in the next section.

One of the simplest forms of analysis is the subdivision of the text into words. In English this can readily be accomplished by finding spaces, since it is an invariable rule that successive words are separated by a space, and possibly also some punctuation symbol or symbols. In some programming languages, spaces are also used to separate words ("language keywords") from each other. But in Arabic rules for the insertion of spaces are not invariably applied, as we have seen in section 2. And when they are, the rules are certainly very different from the conventions of English (for example, the inclusion of ' و ' meaning "and" into the word that follows). Thus, the segmentation of Arabic text into words can only be undertaken using spaces as separators if it is known that a consistent discipline concerning the use of spaces has been employed. In general, the segmentation of Arabic text into words must use some linguistic structure. However, this does not mean the full linguistic structure of the language, but simply enough structure for an analysis like the lexical analysis of compilers (e.g. Gries, 7).

Spaces are often used as separators in programming languages: their use in Arabic programming languages seems quite defensible, so one could imagine Arabic versions of languages like PL/I, as well as Arabic versions of ALGOL and FORTRAN where spaces are (generally) non-significant.

Another simple form of analysis is the searching of a string for a particular substring. Here all the matching considerations of section 3.4 have to be taken into account, together with the fact that if the search is for a word, bounding spaces cannot be assumed. In particular, this means that sophisticated methods for substring searching (e.g. Boyer and Moor (4)) may be inappropriate or at best in need of considerable modification.

3.6 Storage and Retrieval of Arabic Names

When Arabic names are to be stored in an information system, there are three levels of use to which these can be put. Firstly, they can be used passively, being retrieved only when something else is retrieved. They are not used as a basis for retrieval. Alternatively, the names can be used actively, as a basis for retrieval, in which case, secondly, only the complete name is used, or thirdly components of the name can be used.

For passive storage and retrieval, the name can simply be stored as text, and there are no problems. Any analysis of the name would be done by the users of the system following retrieval of the name.

If the name in full is to be used actively for retrieval, again it can be treated as

as a single text item. For fast retrieval an index could be built: there are many techniques available, some of which may involve truncation of the name (see for example Knuth or Lefkowitz): the matching techniques discussed in section 3.4. are appropriate.

It is only when the name is to be used actively with components of the name being used as the basis for retrieval that we become really concerned with the internal structure of names as explored in section 2. The basic requirement is for one or more components to be specified and for all names having these components to be retrieved together with any associated information stored in the system (such as telephone number, address, date of birth). The retrieval request will most likely be able to specify for each name component the role that it plays - first name, father's name, grandfather's name, family name. We may also wish to use other components of a person's name, such as title either formal or informal, or the name of the eldest son as used in nick-names of the Abu variety. How many name components are used, and what their functions are, are important issues which must be decided and once decided used consistently. However, what these decisions are does not affect the considerations of computer software that are made below.

The primary systems design question is how we store the names so that the required retrieval is efficient. The secondary question is how these names are input at initial capture and at retrieval. Two distinct approaches to storage present themselves. The name components can be separately indexed for fast retrieval on separate name components; or the words forming the name can be separately indexed, with retrieval using several words in combination. In both cases the full name may also be stored as text.

If name components are separately indexed, we have two choices, either to have a single index of all names, or to have one index for each distinct name-role (first, family, etc.). In the first case, the partial name for retrieval would have to be divided into components, each name being used to retrieve a set of record identifiers; these sets would be intersected to find those records that contain all the specified names; finally the records would be retrieved and the full names checked that the sequencing of the names and their positions is correct. In the second case, the partial names for retrieval would have to have their roles identified; the appropriate indexes would be selected and searched; the sets of records obtained from each index could be intersected to give the required record identifiers. Note that in the first method it is easier to answer a question like "Two of whose names are Mohammed and Ahmed", than by the second method.

With the name components separately indexed, the real problem lies in the capturing of the name. This cannot be input as a single string of normal text and segmented within the computer, because the rules of formation of names are inherently ambiguous, as was seen in section 2 (for example; Said Alkabeer, Alsheik, Mohammed-Haffez).

The computer has to be helped in the segmentation, and either the components have to be separately obtained from the user, or they have to be clearly identified in the text.

Two methods are possible for singling out the name components within text: either single names consisting of several words can be made into a single unit, either running the words together without spaces or using a hyphen convention as in European languages, with the division between name-components marked by a space, or the division between name-components should be explicitly marked with a special symbol.

Where the name components have to be given identified roles, there is the further requirement on data entry that these roles are made explicit. If the components were separately solicited, the role could be declared at that point. However, if the names are obtained as running text with component separators, it is necessary to have a convention for indicating missing names (e.g. a blank between separators) and possibly a convention for declaring the family name (e.g. the last in the sequence, or always the fourth, though the problem of family names which are not the last, seen in section 2, needs to be recognised). A convention regarding connectives like "bin" and "Al" needs to be firmly established.

Instead of indexing name components, one could index individual words within a single index, looking up each word, intersecting the sets obtained, and finally inspecting the full text of the name to eliminate those that do not match the sequence and position requested at retrieval. The advantage of this approach is that names could then be captured as free text, with all superfluous words like "bin" and "Al". The text would be segmented into words using a grammar for the formation of names, assuming that a dictionary of names could be produced. Alternatively the use of space characters could become disciplined and these be used for word segmentation with special routines being used for removing prefixes like "Al". This second approach would not require a comprehensive dictionary, and is very attractive.

A variation on this approach would be the indexing of word fragments rather than words, using some simple minded segmentation rule like all word fragments between initial and final forms of letters. Indeed, because of its simplicity in software, and potential retrieval power for the user, this method is extremely attractive. However, again some method needs to be found to handle the optional "Al" prefixes.

In both the above two approaches of indexing of words or word fragments what has happened is that the retrieval requirement of retrieval given part or all of the name and the retrieval requirement of fast retrieval using indexes, have been separated. The complete method is as follows: on storage, store the complete name as a single string, but index it by extracting suitable words or word fragments; in retrieval using a partial name, extract the words or word fragments by the same method used for storage, look up all the indexes indicated, and retrieve all the names appearing in all indexing results, finally checking the actual names against the retrieval names for matching

of words (word fragments) in correct order.

Finally let us consider the system where transliterations of names are used. Transliterations provide problems of variability which are comparable to writing down names from dictation in a noisy environment. An exact match may not be found because a different transliteration convention was used, and what is really required is a best match. Best matches in the context of name retrieval in European systems have been well studied and a matching technique based on dynamic programming (Wagner and Fisher 18) based on clusters (e.g. 13) would be appropriate: this method is a simple generalisation of indexed sequential methods of indexing. However it must be pointed out that these solutions are essentially novel, and though the algorithms involved are not particularly complex, they have been developed in a research environment and are not yet part of the software engineers standard collection of methods.

4. CONCLUSIONS

In section 3 we examined aspects of computer system design over a wide front. We saw many alternative solutions to the problems posed by Arabic discussed in section 2, and should now be in a position to choose the best solutions. But all applications are different so what is best in one application is not necessarily best in another. Nevertheless it is of general advantage to adopt a common approach in some aspects of system design, regardless of application. Some standardisation is necessary.

If standards are available, this simplifies the task of the implementor of the system. Instead of having to investigate factors like code design, or peripherals font design, he knows the methods embodied in the standard are good, and can be used with the confidence that he will not have problems later from this source. The designer may also find he has a choice of plug compatible modules to use, and thus save development costs in these areas.

All these advantages for the implementor should also mean a cheaper product for the customer. In addition the customer has the security of knowing that the product he is purchasing attains that level of quality which the standards ensure.

How should we approach standardisation? The most appropriate strategy seems to be to standardise those aspects of computing where standardisation is already accepted for European computing, and to provide guidelines for other areas.

The prime area for standardisation is undoubtedly character codes, with all the ramifications these imply for standardising character fonts (at least their widths). This alone would ensure a plug-compatible peripheral market, and there is clearly a need for free choice on a plug-compatible basis for Arabic peripherals.

A second area for standardisation is programming languages, though there appears to be little advantage in moving to Arabic programming languages other than a desire to become culturally self-sufficient. A programmer, if he is to perform effectively, needs to be competent in a European language in order to be able to use the documen-

tation of the computer manufacturers and to tap the large store of knowledge about computing available in European languages, principally English.

The area of information storage and retrieval has not yet developed to the point where standardisation is possible. Nevertheless guidelines, particularly in the area of name-handling would be very useful; if these guidelines could be given an authority bordering on that of a standard, a lot would be gained for all concerned.

Surely there is a need for all these matters to be widely discussed throughout the Arab world. But in the area of codes, delay should be avoided at all costs, and the sooner that a standard could be agreed at the International Standards Organisation, the better will be the future of computing in the Arab world.

ACKNOWLEDGEMENTS

In writing this paper we have drawn upon a considerable body of experience within SCICON. Part of this experience is embodied in SCICON's special Arabic peripherals and computing systems that we have studied, and part of this experience is embodied in the colleagues with whom we have discussed these issues. We are very grateful for having been able to tap this experience.

REFERENCES

1. ALBERGA, C.N. "String similarity and Misspellings". CACM, vol. 10, no. 5, pp 302 - 313. May, 1967.
2. AMAJID, Abdullah "The Arab Efforts to Simplify Arabic Writing". Addarah Journal, volume IV (3) 1978. King Abdul Aziz Research Centre, Riyadh, Saudi Arabia (in Arabic).
3. BEESTON A.F.L. "Written Arabic" Cambridge University Press 1968.
4. BOYER R.F. and MOORE J.S. "A Fast String Searching Algorithm" Comm. ACM October 1977 Volume 20 Number 10. pp 762 - 772.
5. DEHLAWI F. "Computer-assisted Instruction in Arabic" 4th Saudi Arabian Computer Conference, University of Riyadh, Saudi Arabia. March 1978.
6. EHLERS G. "Production of High Quality Arabic Texts on a CRT Filmsetting Machine" 4th Saudi Arabian Computer Conference, University of Riyadh, Saudi Arabia. March, 1978.
7. GRIES D. "Compiler Construction for Digital Computers". Wiley 1971.
8. KNUTH D. "The Art of Computer Programming. Vol. 3. Searching and Sorting". Addison-Wesley 1973.
9. LEFKOVITZ D. "File Structures for On-Line Systems". Sparton MacMillan 1969.
10. LIESER H.J. "The Use of Computers to Improve Readability of Print". 4th Saudi Arabian Computer Conference, University of Riyadh, Saudi Arabia. March, 1978.
11. MITCHELL T.F. "Writing Arabic" Oxford University Press 1953.
12. MITCHELL T.F. and BARBER D. "Introduction to Arabic" BBC 1972.
13. SALTON G. "Automatic Information Organisation and Retrieval" McGraw-Hill, 1968.
14. SEFLAN Ali Mohammed "The Need for a Standard Arabic Character Set", Second Saudi Arabian Computer Conference, Riyadh, Saudi Arabia. November 1975.
15. TAWAKOL Samy M.Y. "Arabic Character Graphic Representation" Second Saudi Arabian Computer Conference, Riyadh, Saudi Arabia, November 1975.
16. THESSIGER W. "Arabian Sands".
17. TRITON A.S. "Arabic" Teach Yourself Books 1943 Introduction pvii.
18. WAGNER R.A. and FISCHER M.J. "The String-to-String Correction Problem" J. ACM, Vol 21, no. 1, pp 168-173. January 1974.