INTERNATIONALIZATION OF
TELEMETRY SYSTEMS

William Anderson
Loral Data Systems
PO Box 3041
Sarasota, Florida 34230

ABSTRACT
The international market for telemetry systems is growing, and U.S. companies offer
technology that cannot be matched. Foreign customers increasingly require local
language user interfaces on delivered systems. Emerging software standards allow
these requirements to be addressed.

INTRODUCTION
Imagine you are responsible for setting up equipment to monitor the safety of a
prototype aircraft during flight testing. Development of the aircraft cost billions of
dollars, and you are using a state-of-the-art, foreign-manufactured, computer
controlled, data acquisition system to set up equipment and monitor the incoming
data. Your first menu to set up equipment looks like Figure 1-1.

Now imagine your operators are technicians with no training in any foreign language.
Successful equipment setup, data collection and accurate flight safety monitoring can
be compromised by the foreign language interface.

Foreign customers must consider these problems when evaluating U.S. manufactured
telemetry data processing systems. To address these problems, Loral Data Systems is
developing telemetry systems with user interfaces in the user’s native language. This
is a very challenging task.

Information that must generally be supported in the user’s language include menu
prompts, choice lists, help facilities, error messages, and annotation of displayed data.
User input in the native language can also be supported, but concurrent support of
multiple languages results in compromises; if one user defines measurement names in
one language and a second user refers to the measurements in a second language, then
operator-defined measurement names must be entered in both languages. This usually is not desirable.

INTERNATIONALIZATION CONSIDERATIONS

There are two basic approaches to converting systems to a foreign language environment. The first approach is localization. Localization involves rewriting software to support foreign text and conventions instead of English text and conventions. This approach is straightforward, but if a third language support is required then the software effort is repeated. Software development and maintenance costs increase linearly with each language added.

The second approach is internationalization: the process of generalizing system design to make it capable of handling a variety of languages and national conventions. Internationalization of software is more difficult than rewriting, but is far more cost effective for addressing a broad-based foreign telemetry market. This paper addresses
considerations for internationalizing telemetry systems, primarily in the software arena.

I18N is a commonly used abbreviation for “Internationalization” (it begins with “I, followed by 18 letters, and ends with “N”). I18N and internationalization are used interchangeably.

Character Sets

The interface between the user and the computer must be in the user’s language. Fundamental to the communication is the character set used.

ASCII code defines 96 characters that include the standard 26 upper and lower case English language characters. Standard ASCII also supports two other languages: Hawaiian and Swahili, neither of which gives a telemetry company a significant competitive advantage. Other character sets are needed.

International standard ISO 8859/1-1987, commonly referred to as “Latin-1,” extends the ASCII code to include other North American, South American, and Western European languages such as French, Spanish, German, and Italian. Additional ISO standards define character sets for the following geographic regions:

- ISO 8859/2-1987 Eastern European
- ISO 8859/3-1988 Southeastern European
- ISO 8859/4-1988 Northern European
- ISO 8859/5-1988 English & Cyrillic-Based
- ISO 8859/6-1987 English & Arabic
- ISO 8859/7-1987 English & Greek
- ISO 8859/8-1988 English & Hebrew
- ISO 8859/9-1988 Western European & Turkish

Compatibility with the above standards places new constraints on telemetry software. ASCII defines 7 bit codes, ISO defines 8 bit codes. Any software using the MSB in ASCII characters must be redesigned. Also, some languages, such as Arabic and Hebrew, require a right to left writing direction. Typically, support for ISO character sets is provided by the computer manufacturer through special peripherals, system software, and language tables.

Software design for Asian language character sets is somewhat more complex, and typically U.S. companies support only Japanese, Chinese, and Korean, if they support any Asian languages at all. Asian languages are based upon complex characters
formed by root radicals that are indivisible. When different root radicals are combined, characters with specific meanings are created. Two or more characters are often strung together to represent compound words or phrases. In Japanese, for example, a special character means woman; a complex character composed of two characters for woman means women. And a complex character composed of three characters for woman means noisy.

Each Asian language has one or more character sets, defined by its own national standards organization. Complex characters number in the thousands, each of which must be uniquely identified in an Asian language equivalent of the ASCII character set. Each character set contains identifiers that may require one, two, or four bytes per entry. For expediency, only Japanese is discussed to gain a general understanding of special considerations for Asian languages.

Japan has three commonly used character sets. Kanji is a set of ideographic characters originally derived from Chinese. The full Kanji set numbers about 50,000 characters; a subset of over 6,000 characters has been selected for computer implementation. The other two character sets represent phonetic characters. The Hiragana character set contains 46 phonetic characters and is used for writing Japanese words not expressed in Kanji and for verb endings. The Katakana character set contains 46 phonetic characters and is used for all foreign words.

The Japanese Industrial Standard Committee (JISC) has defined three character set standards for computer use. JIS X 0208-1983 consists of 6877 characters made up of Hiragana, Katakana, Kanji, the Roman alphabet, and special graphic characters. This is the character set commonly used by computers in Japan. JIS 0212, containing 6067 characters, was announced in October 1990. Both standards require a two-byte code for each character.

The third character set is JIS X 0201. It consists of Katakana and Roman alphabet characters only. Each character is a one-byte code.

The best advice for developing Asian language telemetry systems is to work with a computer manufacturer who delivers general purpose hardware and software supporting the Asian language required. Typical companies are DEC, Hewlett Packard, IBM, and Sun.
User-Defined Characters

In Asian languages, proper names and special terms may not be included in the standard character set, and there must be provisions for a system administrator to add them as required. User-defined characters require a unique character code, a character bit map, and rules for collation.

Input Method

User input for one-byte ISO character sets is similar to ASCII. Special caps on the keyboard and EPROMs for decoding the keystroke are generally adequate to support the language.

For multi-byte Asian language character sets, various special methods have been developed for user input. The most commonly used include: code-based input, where the assigned numeric code is entered; sound-based input, where phonetic characters are used to define the sound of the desired character; shape-based input, where keystrokes define the component shapes, radicals, or strokes; and direct key in, where a large digitizing tablet with character overlay charts is used. Each has major disadvantages, and no standard for input of Asian language characters exists.

Collating Sequence

In ASCII character sets the collating sequence is usually determined by the ASCII code for the character, with the major sort performed on the first character of the words, a secondary sort on the second character, and so on throughout all characters in the words. In ISO 8859 the character code cannot be used for collation, and independent collation sequences must be defined. Several special considerations are added: new characters must be incorporated into the collating sequence; multiple characters (such as CH in Spanish) must be collated as a single character; and several different characters may be collated into the same position.

In Asian languages collation rules become more exciting. Three different sorting sequences are commonly supported for computer processing. First, characters may be sorted by the root form of the character that gives the character its basic meaning; if more that one character has the same root form, then similar characters are sorted by the number of strokes that make up the character. Second characters may be sorted by the number of strokes in the character if more than one character has the same number of strokes, then similar characters are sorted by the root forms of the characters. And finally, characters may be sorted according to the sequence in which they appear in a phonetic alphabet, organized by romanized spelling.
Date And Time

International users expect the date and time in a variety of different conventions. Names of months and days should be in the user’s language, using local formats.

The telemetry industry in the U.S. has defined dozens of standard formats for time. Converting time to a foreign national standard format probably violates all sense of reason.

Numeric Formats

Different cultures interchange commas and decimal points in representing numbers. Local conventions should be supported.

Monetary Symbols

Most countries have different symbols for monetary formats, and some do not represent monetary amounts with two decimal places. Normally, these are incidental concerns for telemetry systems.

Answering Yes And No

It’s common for a program to ask questions requiring yes/no responses. I18N must support responses in characters appropriate to the user’s local language.

Terminals And Printers

Support of one-byte ISO character sets is commonly supported with custom EPROMs containing a bit map for each character, or with custom print wheels for character printers.

Support of multi-byte Asian language characters is supported on dot matrix printers and bit mapped terminals. The minimum size for a character cell is 16x16 bits; 24x24 bit cells are generally preferred.

STANDARDS

Until recently, each computer manufacturer developed I18N support independently. Non-English application software designed and developed for one computer was not compatible with other computers. Moreover, software often could not be ported to
another country on the same computer. None of this should be surprising, considering that even U. S. application software often could not be ported between computers.

The emerging POSIX standards can improve the situation. POSIX.1 (reference 1) defines the international standard interface between application software and operating systems. By reference, POSIX.1 also defines the international standard for the C programming language (reference 2). POSIX.2 (reference 3) defines the international standard for the command language and system utilities. These standards are the basis for “Open Systems,” by which conforming application software will be portable from one manufacturer’s computer to another with minimal change.

Each of the above standards addresses a uniform method to specify processing of local language character sets and conventions. Each computer manufacturer is free to choose the languages to be supported, and the method of implementation, but interfaces to application software must be identical for all systems.

The C language standard defines a `<locale.h>` header which defines local conventions for numeric data. It includes:

- a data structure, “lconv,” defining formats for numeric values,
- macros for use in the `setlocale()` function. Each macro defines the scope, or “category,” of the locale being defined:
  
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LC_ALL</td>
<td>locale for entire program</td>
</tr>
<tr>
<td>LC_COLLATE</td>
<td>locale for collation functions</td>
</tr>
<tr>
<td>LC_CTYPE</td>
<td>locale for character handling functions</td>
</tr>
<tr>
<td>LC_MONETARY</td>
<td>locale for monetary functions</td>
</tr>
<tr>
<td>LC_NUMERIC</td>
<td>locale for decimal point information</td>
</tr>
<tr>
<td>LC_TIME</td>
<td>locale for date/time functions</td>
</tr>
<tr>
<td>LC_XXXX</td>
<td>implementation-defined categories</td>
</tr>
</tbody>
</table>

the “setlocale ()” function sets, changes or queries the entire locale of a process, or portions thereof, according to the values of a “category” defined above and a “locale” argument. Choices for the “locale” argument are limited to implementation-defined strings. There are no firm standards on the way locales should be named. OSF recommends (reference 4) that the locale name specify language name, territory name and code set name, and offers a specific format for use. The rationale is that each of the items may be specified independently.
the “localeconv” function sets components in structure “lconv” with values appropriate for formatting numeric quantities according to the rules of the current locale.

The C language “locale” capability is incorporated into POSIX.1 and POSIX.2 by reference, and is extended to include shell language functions.

The C language standard includes provisions for multi-byte character sets, wide character type definition, and wide character processing in a limited number of string handling functions. However, the LC_CTYPE macro only supports one-byte, alphabetic languages. Independently, some companies are adding all-new functions to support multibyte character sets.

A fundamental requirement for software I18N is to separate messages from code. Then, as a user’s language changes, the same program selects text from a different file. This is an essential requirement for any I18N design, but is not addressed by current standards. The application designer must either implement a custom text management and retrieval system, or use the facilities supported on a particular manufacturer’s computer.

**IMPLEMENTATION**

Theory and emerging standards are fine for philosophical debates and technical papers, but customers need systems that work, and are delivered on schedule. How is a real system built?

Loral Data Systems has delivered a number of telemetry systems to foreign customers. The implementation approach used includes developing software to international standards to the extent feasible, and to work with a computer manufacturer who has a strong presence in foreign markets. LDS is designing its software to the standards discussed above, as well as a number of standards sponsored by industry groups. Software for the user’s interface complies with the OSF/Motif standard (reference 5). Motif provides support for non-English languages, including the multi-byte characters of the Asian languages.

Loral works closely with Digital Equipment Corporation (DECTM), a company with an excellent world-wide reputation, and a strong presence in most of our foreign markets. DEC also aggressively supports system integrators selling their products in the international market. They have published a general reference for developing international software which covers design considerations, VMS tools, Unix tools, and summaries of specific requirements in various geographic area (reference 6). DEC
has also published documents to aid the integrator with Asian language software (references 7, 8). Recently, DEC delivered to Loral Data Systems a system for Japanese language functions, including Japanese language hardware and software. The hardware included printers and terminals, while the software included the POSIX operating system, Windows user interface, 3-D graphics, and a relational data base management system. Both hardware and software have Japanese and English modes for system and application functions. DEC assigned a product manager to help with the start up, and a consultant from Japan was hired to help with language specific issues. Progress to date has been on schedule and budget, with remarkably few surprises. This was the first Japanese system delivered by DEC to a U.S. customer. Now DEC offers it as a standard product.

Other computer manufacturers also support foreign language requirements. Sun is addressing the Japanese market (reference 9), and general articles addressing overseas applications are appearing in the open literature (reference 10).

SUMMARY

Emerging software standards allow internationalization of telemetry systems. The increasing costs of flight tests is rapidly outdistancing the decreasing costs of customizing software for non-English languages. Non-US users must consider the desirability of specifying delivery of user interfaces in the user’s native languages.

REFERENCES


