

The Training on “Local Language Computing Fundamentals”

Internationalization/Localization ISO/IEC 10646 (Unicode) Hints for Asian Input

2004-01-19/23
Lahore, Pakistan

Takayuki K. Sato

Chief Researcher, International Information Technology Lab.,
Center of the International Cooperation for Computerization –Japan

In this training, Sato is going to make three presentations.

Part-1: Internationalization and Localization This presentation discusses about the need of “single binary” approach for affordable localization

Part-2: ISO/IEC 10646 (Unicode) This part expresses that ISO/IEC 10646 is designed for the single binary localization.

Part-3: Japanese input method This part demonstrate the most sophisticated input method (Japanese input method), that will give a hints for designing much friendly input method for Asian characters.

INTRODUCTION

From late 70th, I was involving a localization of IT. Since then, every 4-5 years, I saw a boom of localization. Every 4-5 years, new comers started the discussion on IT localization, and went away in 1-2 years. They never come back again, then new faces come again saying the same thing, but repeating the history again. Among those booms, there are very common things. People think that localization is an independent closed technology and other people never challenged the issue. But this is not a case, this is why those people never come back the localization business again.

Today, my main discussion is “localization is very easy job, but it is very difficult business”. Peoples are coming in this business thinking that this is technical issue, technically, localization is very easy, there are many method possible to make it happened. Besides, it is very difficult to run it as business, unless having right approach, affordable localization is impossible. This is why, many people are coming in the business, and went away.

Today, I am going to discuss, why it is easy to lose money, and an approach to perform affordable localization. Also, at the end, I am recommending that national IT strategy should include a “localization” policy.

Attachment:: Importance of ICT int std-2003.doc What is CICC.ppt

Part –1 Internationalization and Localization

This is for part-1 of 3 (Internationalization and localization)

“Localization is easy job, but localization is very tough business”. “Therefore it is necessary to build up all localization activities from business view point” is a message of my presentation for 3 times of 1.5 hours.

1. Localization

At the beginning, let me try to define a localization from its objective view.

There is an e discussion of “how to provide a programming and its execution environment including an I/O of local language in local character”. And many are seeing this is localization. However, I like to say this is just a small part of localization. And this is the most easy part of localization.

There is a strong desire of every country to make the country competitive in economy, industry, socially, politically and so on..... IT is one of the key and essential elements to make country competitive. Wide range of utilization of IT is needed for making a country competitive is an idea. The most popular expression for this is to bridge a digital divide. To bridge a digital divide, an average people of the country should be able to use the world best software with in country. It should be using their own language and culture. The world best software is the key word. By using 2nd or 3^{ed} class software, it is not possible to make a country competitive. This is a key point.

Especially, at that age of world network, interoperability of data across the boarder is very important. If there is incompatibility between the data, it means another digital divide. This is why localization of the same product (not inventing similar solution) is needed.

Localization is to provide a world class IT solution in local language and culture with interoperability with outside country.

Localization is not simple capability of local language.

2. Requirements

Actually, user requirements are: as follow in order.

- The most popular (or best) in the world solution should be available.

- Operating manuals in local language

- Local language out put in local character (Romanization is not acceptable)

- With user friendly input method

- Support all cultural conventions in local ways

- User interface should be in local language

- With data processing in local custom (like sorting)

- Customize to what user want to have

- Finally, any up-grade and up-date of the original software should be available in local language and custom at the same timing as the original products at the same price as the original software (hopefully at free of price).

The process to make above available to user should be called as “localization”. Unless having that localization, user never satisfy, And the country will not become competitive. This is not simple program development with local character I/O capability.

3. Hard coded age localization

How "localization" was done in the past? What is an issue?

Let me start with the localization effort of IT solutions in traditional (or old fashioned) approach.

Practice described following was very common up until mid-80th, since this approach is too expensive, much productive methods are introduced for now already, but it is better to start from this point for better understanding of current localization process.

At early times, all programming was so called "hard coded". Every things are coded as a part of a program.

Program was tuned up very complex for better performance Also, operating manual accompanied with the product was thick and big. And up-date was very frequently done.

- Operating Manual Translation (and printing)

Manual translation, printing and furnishing to the product was the most expensive part of the localization, and it is still major part of localization.

Translation by technical writer takes long time and the quality of the translation is very critical factor. Releasing poor quality manual invites too much of bad responses from customer and created the bad reputation of the supplier. And to keep certain level of translation is very costly. In addition, the translation of manual is boring job. This has been one of the most unpopular jobs for the average engineers.

In addition, frequent up-date causes furnishing work very difficult (means very high over head, it means another cost), and most of the printed manual should be written off (another cost).

Today's on-line manual made the printing business easier, but translation work remains as same as it was.

- Out put in local character

This is very visible business that every body started as a beginning of the localization. Usually, font is located independently from other program, so it was not difficult job to find out the location of the font within a products.

The difficulty of the font is that normal engineer does not have enough sensitivity to design better shape of character, user always request better one than it is used.

Unlike independent display terminal, current technology in PC is making the font provision much independent and easy. There is no significant technical issue.

Real issue of the out put is the requirement of better quality of font and return on investment of the font provision business. Anyway expectation is "let local keep supplying the better one on suppliers own risk".

Another issue with the font is that font technology is progressing very fast. The motivation of the new technology development is new script with new behavior as well as better quality. Font development should keep following the new technology. This destroys a font developed by developing countries because they are always depends on old fashion technology. Usually, the old technology requires more complex modification on the original program.

- Input of local character

At the time of hard coded program, the key process for localization of input system was to find out all input related part out from program. Then, modify the program if it is necessary. But need of the modification at early time was small, thus the input localization was relatively easy job.

Unlike a mechanical keyboard of old teletype, electronic keyboard has made a design flexibility of keyboard much flexible.

After ISO/IEC 10646 (Unicode), normalized sequence principle is making the input program much larger and difficult job than before. I will come back to this discussion in later.

- Support all cultural conventions in local ways

Supporting cultural convention was big job at the age of hard coding. Scanning all program to find out the hard coded cultural conventions was very time consuming job. Then, all cultural convention support parts should be modified to meet with local needs.

Also, there were no standard set of cultural convention, it was up to the programmer of localization to define the local needs of the cultural conventions. This method made a quality of the support low.

- User interface should be in local language

User interface (messages out from computer) should be in local language. It was needed to scan all program to look for “hard coded messages” and replace them with “translated messages”. This is very big job with no fun for engineer. Only engineer can find out the “hard coded messages”, but engineer were poor for translation. Professional technical writer is needed for the translation, But to maintain good working relation between engineers and technical writer is very difficult. Also, this job was very costly. In addition, source code for scanning was available for localization at very late timing of the development of the original product. It was impossible to release the localized product at the same timing as the original. This cause another incompatibility problem within the world network.

- Data processing in local custom (like sorting)

Like cultural conventions, it is necessary to scan a program and find out related part, then modify them. Sometimes, the processing sub-program is shared by other processing (such as character sort with number ordering), then simple modification does not work well. Very careful modification and testing is needed

- Customizing

Customer never satisfies with the standard functionality. Customer likes to have some part of products modified for customer’s special needs. Again, scanning a program and necessary modification is needed.

If the customizing is an addition of functionalities, it is much easier than modifying some of original functionality. Testing of modified result is also big job.

Scanning and Modification is a key word for localization of “hard coded age” program.

It cost too much of money and resources for local, and it is very difficult for local to afford the cost and resource requirements. Even so, forecasting of market size and necessary cost are possible, and thus decision making is possible.

- Maintenance

Finally, any up-grade and up-date of the original software should be available in local language and custom at the same timing as the original product at the same price as the original software (hopefully at free of price).

As a reality, suppliers of the original product do not want to have a responsibility of up-dating of the modified products. They say that those are different products than what they released to the market. The responsibility of the up-dating is with the localizer. This is very large work and, further more, sometimes the modification prohibits the up-date.

Thus, scanning for up-date is far more difficult and boring job than first time localization. It costs almost the same amount of the initial localization. Besides, customers never pay for up-date and customers request real time up-date. In another word, the maintenance of localized products is revenue less business. In addition, because the maintenance should be done in short time, there is a needs of big peak of the localization capacity, then, after the

peak load, it will be a job-less status. Job load control for localization is very difficult work.

Above indicates that the localization at the "hard coded age" is "scanning, modification and testing" jobs. Those are not much difficult technical job, but very time and resource consuming job. That means costly and also it is including many management difficulties. Many company failed the business because they see the job just from technical viewpoint, those people did not see that the localization is a time and resource consuming business with very small income.

This is a key issue of the localization of hard code age.

One more fact should be recognized. A hope of the scanning and modify approach is acquisition of the technology. This practice did work well at age of main frame and/or old industries. But the practice does not work well for software.

4. Current solution (i18n/l10n approach)

Solution on above issues is to avoid "scan and modification".

- Manual

As described before, move to on line manual is one solution for printing and write-off costs.

One more movement in advanced countries is give the manual business to publishing industry. There are many textbook like manuals are published commercially and sold at book store. This is independent from the product vender. This resolves the manual issues of the venders. I am not sure this practice does work for small market or not. May be, some of new approach such as simplified manual publication and IT kiosk type approach might be invented.

- Out put

Character shape generation is independent from any specific software.

Further more, better font supply becomes independent business.

Thus, there will be no big issue on this.

Question is that if there will be font vender motivated for small market or not?

- Input

Now, input mechanism is only an independent (necessary) code sequence generation software. It is not necessary as a keyboard, but any such as voice input, hand writing, brills or any.....

Because it is independent from any, venders can invent many kinds of new approaches. And free competition may motivate many new inventions for user friendly input method.

An issue with input is too small market. It is very difficult to encourage a vender for better input method development.

- Cultural conventions in local ways

Cultural convention support should be centralized to one place, the place should be independent from any software, and it should be switch selectable for selected country/region/application. This functionality is well accepted and supported by operating system as of now. Thus there is no big concern on this. This functionality is widely known as "locale".

Only thing that local should do is to disclose a locale data for the world.

- User interface translation

Hard code of message should be changed to code independent facility "message catalog". This is a message outside of main program, and there is a language independent linkage between program and message catalog. Local can only translate the messages in the catalog and replace with the original. No scanning and modification is needed.

By mean of this, requirement of engineer is lower than hard code age, and quality translation is key (and only) objective of the user interface localization.

- Data processing in local custom (like sorting)

Use a common sub-program for all purpose. Change the common sub-program if necessity for localization is happened. Necessary. API for the common sub-program should be standardized.

This is almost like "hidden locale", and the candidate for future locale item.

- Customize to what user want to have

Customizer hook interfaces should be provided. All customization should use those interfaces.

Also, Customize and personalize facility should be separated clearly.

By doing above, there is no need to do source code dependent modification for localization.

Localization becomes a set of independent business such as manual publication, font development, input method development, message translation and so on. Those business are not inter dependent, thus the implementation (or supply) can be done separately. This is another important point.

Actually, there is no need to take a look at source code of original product, needs for engineers are minimum. Finally, testing requirement will be minimum.

- Up-date

In addition, up-grade and up-date of the code affect nothing on localized result. Only up-date of the message should be concerned. By mean of this up-date, cost of and necessary time for localization of up-date is minimum and may become affordable.

5. Conclusion

In different expression, original product should be separated in to two, one is the program code and another is empty containers for cultural dependent items. Program in this structure is named as **INTERNATIONALIZED** (I18N) program and filling process of the empty containers with local data is named as **LOCALIZATION** (L10N).

The I18N/L10N process is the way to go. (See ISO/IEC TR 11017)

Rule of thumb is: make cultural dependent data out side of program. Avid any cultural dependent part of program coding. Then localization does not require any code modification, so as code testing.

This also means that the localization is not a high technology engineering work any more.

Only condition for program part is "**single binary**". Character code for any script should work within the program equally. This is why special code ISO/IEC 10646 (Unicode) is invented.

About the ISO/IEC 10646, let me discuss about it at next section.

To make the single binary possible, all programmer must know all requirements of the world. This is not possible.

The possible way is to reflect all requirements on international standards and engineer follow the international standards. Most requirements of characters from Asian countries are already on the international standard already.

(For detail, ref. to separated paper titled as “**Importance of IT international Standard**”). From this view, developing regional standard (such as Asian) does not make sense. It is better to disclose all requirements on international standard. Then national standard should tell how to use it as the country.

6. Related Standards and Organizations

ISO/IEC 646	ASCII code
ISO/IEC 2022	Switching mechanism between coded character sets
ISO/IEC 2375	Registration of final byte for escape sequence to be used by ISO/IEC 2022
ISO/IEC 6429	Control codes
ISO/IEC 8859	8-bits cods
ISO/IEC 10646	Unicode
ISO/IEC TR 11017	Framework of Internationalization
ISO/IEC 14651	Sorting standard
ISO/IEC 14652	Definition method of cultural conventions
ISO/IEC TR 16285	Character Glyph model
ISO/IEC 15897	Registration of cultural convention

ISO:

IEC:

ISO/IEC JTC1 Joint Technical committee between ISO and IEC (Information Technology)

ITU-T

JTC1 SC2 Coded Character set

JTC1 SC22WG20 Internationalization

JTC1 SC34 Documentation

JTC1 SC35 User Interface

Unicode Inc.

Open Standard Group: Linux internationalization

W3C, IETF

CEN Standard development body for EU

ANSI US national body

CICC Center of the International Cooperation for Computerlization

See different presentation for CICC activities such as SEISA or AFIT

-----end of part-1-----

Part-2 ISO/IEC 10646 (Unicode)

This is part-2 of 3 (ISO/IEC 10646)

At localization session, it is said that single binary is a key for affordable and quality localization.

There are many technical explanation on what is ISO/IEC 10646 (Unicode). Most of them are discussing how it look like. Besides, there is no explanation why it is necessary for the single binary. This presentation focuses on “why the current design approach is selected for ISO/IEC 10646?”

It is easy to understand, if there is any script dependency in the binary code, the localization becomes costly and causes poor quality.

1. Character set adaptation process

First, let me describe three methods that adapt a program for different scripts.

Case-1: Multiple Source Code approach

Take a original program and apply source code modification to adapt the nature of different scripts.

This is multi-source approach. And most easy way to think of.

This means, program source code is script dependent and as a result binary code is script dependent.

This means it is necessary to re-adapt the code at the time of up-dating. This is a basic cause of incompatibility with the original program.

Case-2: Single Source and Multi-Binary approach

Use the same source code, but compile the same code according to the nature of character code.

This is single source and multi-binary approach.

There was a high hope that this method is a solution, but it did not work well. Different binary is still

Different program and cause an incompatibility.

Specially, tune up for the maximize performance make this method not applicable.

Case-3: Single Binary approach

Use one binary code for any scripts. This is so called single binary approach that “LOCALIZATION” of today is assuming.

Because it uses only one binary code, there is no room to create the incompatibility.

However, there is no free lunch. Compromises on character coding is necessary for singer binary approach. The compromise causes many arguments with ISO/IEC 10646 (Unicode) encoding principle. This is why an explanations on ISO/IEC 10646 is necessary.

In this session, all explanations are trying to express why the specific ISO/IEC 10646 (Unocode) is developed for single binary localization.

2. Basic Elements of Coded Character Set

There are five key elements of coded character sets.

Coding scheme

Coded elements (usually called as character, or repertoire, but those are not accurate enough)

Behavior of the coded elements

Coding order of coded elements

Name of coded elements (normally called as character name, but this is not accurate enough)

In basic, or in old days, different code means that the code with different coded elements (so as order and name). That was all about the different codes. And the behavior of the coded elements were assumed as they are the same.

3. ISO/IEC 2022 (locking shift solution)

In principle, different codes are able to share the same binary program when only the coded elements are different and coding scheme and the behavior of the coded elements are the same.

Very old approach of swapping font for localization and use the program as it is (by accident it is single binary) is based on this principle.

The approach did work well for European scripts, however the problem was very small code space (96 or 192 characters maximum), it did not work for an area where many scripts are in use..

First solution was to switch the different codes. This method is called as "locking shift" method and is known as an ISO/IEC 2022. The method worked well for data interchange purpose. However, there are two issues for data processing purpose.

One is the locking shift by it self. By taking a look at the code value it self, it is not possible to define which character it is. Scanning for find out the locking status is necessary. This is additional (but looked unnecessary) job for programmers and slows down a process.

Another issue is multiple processes. As far as the character behavior are the same among the different codes, it is possible to share the same processing program. However, if there is a character code with different behavior, then the processing program has to cover both behaviors. No one knows what kind of new behavior will be invented in future. Solution were always killed by someone's new invention. ISO/IEC 2022 did work well for European scripts, but had an issue with some of non-European scripts.

First, CJK ideographs in North East Asia region needs to have very wide code space. Mixture of the codes with different code space makes the program for data process very complex.

Special rendering and combining process in some Asian scripts requests another additional processing.

There is no way to control the behavior of newly developed character code (that are done by many different groups). It is just a registration per ISO/IEC-2375, it is impossible to let programmer know all behaviors.

On the other hand, some scripts made a compromise to be adapted to the behavior of the scripts that are already existing. This compromise allows to make (primitive) single binary possible. At this case, there is a limitation of the processing capability. Normally, compromise is made for good document processing, and in-perfect capability for data processing.

As a result of compromise, there has been a limitation of localization and still needed small source code modification. Finally, the primitive localization for Asian script does not work well in ISO/IEC 2022 approach.

4. ISO/IEC 10646 (Unicode)

ISO/IEC 10646 has been developed to resolve this problem. The ISO/IEC 10646 approach is opposite from the past approach. In the past, coded elements were almost same as what national users want to see as their

characters. And by changing the program to process the code, the localization was done. Font technology, beside, at this time was as same as other scripts. In case of ISO/IEC 10646, the compromise is made at coded elements, and localization is done with minimum of code modification. Also font is designed by using the new technology.

Form coded character set technology view, ISO/IEC 10646 is

Using coding scheme with very wide code space to place all coded elements in it, code scheme discussion is not necessary.

Put all necessary elements of the world into one scheme, no need to consider the difference of coded elements.

Make behavior of characters among the world script common, no way to have different program to process the different scripts.

Give up binary sort, thus, no need to concern the order of coded elements within the code table. Sort key based ordering is the solution. (Refer to ISO/IEC TR 14651)

World unique character name for each characters are assigned

As a result, base for single binary, base for better localization, is established.

ISO/IEC 10646 is

Wide coding space, In theory, full 31 bits area is a set. This is far larger than what the world is necessary.

In practice, about 1 million of code points are available under UTF-16 environment, this is assumed as good enough number. Note that UTF-16 is still 16 bits environment.

Unfortunately, original Unicode idea of 16 bits space (65K) is not enough for even CJK ideographs.

By using wide code space, there is no need for the "locking shift".

However, the wide code space requires different processing from traditional 7-8 bits. Transition from 7-8 to 16 bits is in progress.

Also, conversion between 31(16) bits environment and 8 bits is provided as UTF-8.

Behavior of coded elements: Traditional simple behavior does not work for many scripts in the world, There is no way to stick on the old behavior, the new standard behavior has been set, and coded elements for all scripts are re-selected according to the new standard behavior.

The key points of the new behavior are as follow:

- Unification: All the same characters in the different scripts are unified
- Assign the code for character (logical unit), not for glyph (abstract shape). (new character glyph model)
- Combining sequence and/or Normalized sequence is used, use of pre-composed character has been discouraged

Unification: Many scripts are sharing the same characters. For example, Most of German scripts are the same as script in UK. China, Korea and Japan are sharing almost similar ideograph. Nepali is with Devanagari and Urdu with Arabic, If all scripts are coded as they are, there might be multiple-coding problem, at least, confusions with the user is unavoidable. Unification between tightly related scripts avoid this problem.

Nepali is a part of Devanagari extension and Urdu is a part of Arabic extension.

Unification invites some problem. User of one script may see unnecessary characters on the code table. This reduce the credibility of the code itself, or misuse of the code. This is why national subset of the ISO/IEC 10646 for national use is necessary (need of national standard after the international standard). One more possibility is over unification and/or under unification. Japan has arguing on this issue in international discussion for more than 10 years.

Separation between Character and Glyph: In traditional view, character should have a visible shape to read (glyph) as well as logical (invisible) meaning (character in new definition), coded character set in the past was coding for the glyph in this view, but ISO/IEC 10646 is coding for the character.

For example, in case of Arabic, coding for presentation forms are not done (it is there for backward compatibility purpose, but it is for limited use). Font (intelligent font) should take care for glyph selection for the same character.

If glyph is uniquely definable always, the method is fine, if not, such as ambiguity, font can not do any job. ISO/IEC 10646 is using variation selector technology for this case.

ISO/IEC TR 16285 is an explanation on the Character and Glyph model

Normalized sequence (and combining sequence): One character in traditional view can compose several ways. If, there are multiple ways to compose the same character, multiple encoding would be caused. ISO/IEC 10646 select the method to encode the minimum and smallest elements. And sequence of the elements define the glyph. Intelligent font is a rendering engine for this case.

Again ambiguity possibility, special sequence (using formatting characters) is necessary in this case.

Above behavior is very different from what linguists in the country see the national characters. There are many objections from the countries for ISO/IEC 10646. But most of them are not accepted by ISO/IEC JTC1 SC2 because most of the objections are saying “back to the useless old coding”.

It is recommended that if resultant character on the screen is right, forget about internal to computer representation.

5. Frequently raise other objections

Above principles are not well know, thus, at the discussion of ISO/IEC 10646, there are many objections from many countries. The counter proposals are based on different principles that are usually old fashion, then most of the objections are rejected.

In addition, there are very common objections.

Character naming: Each character on ISO/IEC 10646 are named uniquely from others. The name is an identifier of the character and it is used as mapping information to other coded character sets. Often, there are voices that the naming is not culturally correct. Due to the two reasons, most of the proposal for re-naming are rejected.

One is the fact that naming to make every body satisfied (and unique) is impossible (see the name as just a stream of Characters). And another is to change the mapping keys may invite a confusion of the user.

Unnecessary characters: In many cases, due to the unification, there are un-familiar characters are coded with in a collection of a script. Countries tend to request to remove them. It is not accepted because the unnecessary characters might be a necessary characters for someone. (That is why those characters are encoded)

Coding sequence: Code sequence is not for binary sorting. There are many request for reordering for binary sort. The requests have been rejected because, sharing same character set between many scripts made binary sort order for every body impossible. And future addition may cause a problem anyway.

For ordering, multi-sort key based order is recommended. (See ISO/IEC 14651)

Printed shape polish up: Proposal of better shape for printed character is often proposed. Some of them are really valid requirements, and some are requesting just for switch for the national favored shape from other national favored shape. The later cases are not accepted. Remember that the ISO/IEC 10646 is encoding a character, not glyph or character image.

6. Is ISO/IEC 10646 perfect?

ISO/IEC 10646 is new, and not proven fully yet. There might be some un-known problem in it.

The problem will be detected as something impossible to express some local character(s)

If there is something impossible to do by using current coding, please say so. But do not say "I do not like it because it is different from current national practice".

Many people in developing country tend to think that foreigners are not understanding my culture, and tried to express the culture. It did not work well. Remember that the coding based on national practice does not work well for single binary, this is why ISO/IEC 10646 is needed.

Some times, country might receive unreasonable response from the SC2, if the requirement is really needed. Then do not give up. Japan needed more than 10 years to convince the SC2 for Japanese requirements. Don't be emotional, always express what is wrong and what is need.

CICC activities such as AFIT are willing to assist to resolve this kind of real requirements.

About CICC, I will explain at the end of this presentation.

Extra: What is standard?

One of the reason of needs of standard is as follow:

If there are many possible methods to meet the requirements, and if none of the method is superior to others and if a free of choice of the methods may cause a technical, economical and/or social disadvantage, there is a needs for single standard method defined.

This means that the solution defined in a standard is not only solution for the issue. There are many other possible ways, and those others are not inferior from what standard is defining.

Because of this reason, there are many counter proposal saying:

This method is better than current standard. The answer for the proposal is "So what?".

This method is probed to work. The answer for the requirement is "So what?"

This method is defined as national standard. The answer is "Wait a minutes".

----end of part-2-----

Part-3 Hints for Asian Input Methods

This is part-3 of 3 (Input method)

Input method.

Input methods for Asian characters are still premature. This presentation is trying to hint something for future Asian input possibilities by taking a look at Japanese input method.

I have kept saying that do not standardize a national key board layout. Besides, the first thing that many countries challenge is keyboard layout. Even assistance from international organization sometimes recommends to start with the keyboard. As a rule of thumb, if the person like to start with the keyboard, I like to say, the person might not understanding the reality of localization well.

Think note PC, there are so many different keyboard. If national standard require one way, some of the products can not support the standard. Some times, standard kills a feature of the new keyboard. Current key code mapping table is very easy to swap to any keyboard layout at any time, why it should be standardize before enough experience is obtained.

Demo – Japanese input.

Japanese input method needs to handle at least 7000 characters by using normal English like keyboard. How to make that happened?

Remember that the input method is just one program to generate a code stream to the main program. This is only objective of the input method.

Think the most primitive “ubiquitous”. It is possible, simply changing code sequence generation method (such as brills) without changing any main program.

Demo of Japanese input method includes

- To show key stroke is not directly related with the code generated
- To show multiple key strokes generate one code
- To show small number of key strokes generate a long code sequence

In case of normalized sequence input, this is often used for Asian script, sometimes a generation of a code for formatting character is necessary.

Does hitting a key for “formatting character such as NSNJ” friendly with Asian average people?

Special training for computer usage is necessary. This is not a good way to do.

Flexibility of Japanese input might hint a solution for this case.

The good friendly input method will not be invented at a night, competition between developers for long years will give the solution. This might be a real local value and local contribution. Why discourage such movement by setting a standard? Why not leave them a free competition in the market?

Conclusion of the three presentations.

- Real localization for bridging digital divide is very expensive, funding and resource requirement are very large, therefore, localization plan should be reviewed by business plan view, not from technical view.
- Naïve localization enlarge a digital divide. Be careful.
- Because of this reason, technology development for less cost, lower resource requirement and higher quality is under way.
- The direction of technology is “independent pieces”, free competition within country for much friendly solutions is expected. Do not over kill them by national policy/standards.
- For this purpose “input of local requirements for world developers” is needed. Asian wide cooperation is needed for the disclosure of requirements. CICC has been assisting Asian Countries to make it happened (AFIT, SEISA, DocSII).
- Localization is long-term commitment to the users. Once it is done, no way to get back to non-localized status. This is a reason for cost and resource requirement.
- Because of this reason, localization program should be very strategic. National IT policy should clearly state the strategy.
- For most of Asian countries, localization without outside (funding and resource) support is impossible. However, characteristic of the international supports tend to be one-shot and unplanned. Harmonization with the supports and national plan is necessary. Do not accept all offers randomly.
- Avoid good will but naïve support fro out side countries.

What is CICC?

See slide presentation

----end of part-3-----