

Around the Interface in 80 Clicks

by Duane Degler and Lisa Battle



The use of computers internationally has undergone a dramatic increase. The rapid expansion of the Internet has given people access to the same applications and content anywhere in the world. Because of this, user expectations are going up; users do not want to learn another culture and another language to perform work.

Computers users have become more late adopters than early adopters. They expect that software will truly meet their need” and they need a “good user experience;” it is also important that they have “no feeling of puzzlement, no loss of control” (Norman, 1998, pp. 34-36). As designers, we can’t necessarily rely on current interface conventions that appear to be “standards.” The tools have to match people’s working needs and lifestyle needs.

These user expectations are strikingly similar to the expectations we have seen for software. We believe the demand for software that is usable across languages and cultures is the beginning of the next wave of demand from users, and the next great challenge for software designers. Software can and should make an extra effort to provide a good user experience.

Designing adaptable, internationalized software is difficult in part because of the lack of guidance within the literature of software design. The software developer, the project manager, and the consumer do not have ready sources of non-academic information on this subject. Books or websites that deal with the subject tend to be focused on cultural investigation/analysis (del Galdo & Nielsen, 1996) or on detailed programming/translation guidelines (Cross, 1999; Ebben & Marshall, 1999; Sun Microsystems, 2000; W3C, 1998). How do these guidelines translate into the practical design of applications and websites? There is a need for more holistic checklists of what to think about when designing for the international user (Ishida, 1997).

As user interface designers and performance support specialists, we are not experts in internationalization. However, this subject has come up repeatedly in the course of our work. We have been asking ourselves, What do we mean by performance, and is it possible to support it internationally? As a result, we have started building our own checklist of questions and issues that need to be addressed when designing software for international audiences.

Performance-centered design and performance support are not just about helping people use software applications. Rather, they aim to help people perform real-world tasks successfully and increase the quality of the outcomes. This increases the importance of providing localized support. It isn't enough to describe in a particular language how to enter data into the software. We need to know how to perform a job successfully within another cultural context. In addition, many people are increasingly doing knowledge work. For them, it isn't enough to support a task; we also need to be thinking about supporting knowledge acquisition and use across cultural boundaries. Culturally restrictive software could potentially get in the way of that goal. The more we try to make things context relevant for one group of users, the harder it is to make things culturally relevant for others.

What Is It?

In their book *International User Interfaces*, Elisa del Galdo and Jakob Nielsen outline three levels of international user interfaces and confirm that there is insufficient expertise and experience among practitioners with implementing levels 2 and 3 (del Galdo & Nielsen, 1996). The levels are as follows:

1. Process and display the user's native language, character set, notations, and formats.
2. Produce an interface that is understandable and usable in the user's native language.
3. Accommodate users' cultural characteristics.

The terms internationalization and localization tend to become lumped together, and they have several definitions. While they are closely related, they can usefully represent two ends of a design spectrum. Internationalization creates representations that are equally understandable across different cultures. In some cases, this can be achieved through basic symbols (such as arrows, or icons based on objects in nature). However, in most cases, a symbol has to be learned, and thus the use of international symbols leads to a "one-size-fits-all" standard that takes time to become familiar with. Eventually, users can learn to understand and conform with de facto standards.

Localization involves adapting the software to the language, culture, and norms of the local culture, so there is no cognitive dissonance or need to translate. Localization is like setting preferences that are appropriate for an entire group.

There is an implied diversity in localization, because the software should support the diversity of cultures. But due to the way localization often has to be practically implemented, this is a false diversity. Translating language is not the same thing as translating culture. Today a translation of software into French is considered sufficient for all of France, Belgium, Quebec, Martinique, and several countries in north Africa. A translation into German is considered suf-

ficient for Germany, Switzerland, and Austria, just as English is suitable for Britain, Ireland, North America, Australia, New Zealand, and South Africa. While these places belong to a language group, the dialects, spelling, vocabulary, and culture are often very different, and there are different metaphors and ways of working. Translating the language only is what we call "linguification" rather than localization. There is an expectation that the user will still make the effort to bridge other cultural gaps.

The Obvious Stuff

A number of things tend to get the most immediate attention, partly because they are very irritating to users and partly because they are easier to identify and fix (although the people doing it might not think so at the time!). The purpose of this article is not to dwell on these things, which are covered in detail in other places, but they deserve a mention. Figure 1 lists these types of details (see Sidebar, page 34).

Designing for Performance

To support performance for an international audience, we need to examine the key elements of performance in relation to the design of systems. The three-element model of performance design that we use is the one introduced by Gary Dickelman a number of years ago (Dickelman, 1996). The elements are the performer, the process, and the information, all of which must be integrated (see Figure 2). Optimal performance, or the "Performance Zone," occurs when all three of those elements are working together. If a software product does not take into consideration a whole range of international issues, it can break the link between the elements.

Disconnecting the Performer

When dealing with internationalization, the primary focus is the performer, not just as an individual, but as a cultural group. The product for a performer in one cultural group may not work for performers in another. Imagine that we have taken the three-element model of performance and removed the "performer" circle, and then attempted to

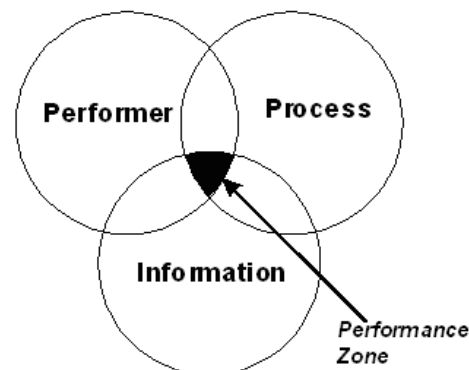


Figure 2. Key Elements to Designing for Performance.

Language Translate the application or website and performance support elements into major language groups. Allow for further translation of content elements (such as help files) locally by representatives of the user population.

Text Display Allow extra space for text to expand when it is translated into other languages. When designing screens, make sure that text can wrap without ruining the design or truncating the text. If the screen will contain a combination of message text and data, consider the effect of concatenation on the translation.

Address Format Do not lock all addresses into a “state and zip code” format. Most countries have counties, provinces, districts, or cantons (and they rarely are two letters long) and post codes that are not five sequential numbers. Include a “Country” field in addresses and allow the selection of country to influence the display of other fields.

Telephone Number Format Do not lock all telephone numbers into “area code/number” format, particularly the 3-3-4 US number format. This applies to both the field formatting in the user interface and to data validation. For example, a popular travel website surprised us by requiring the user to enter a phone number in a specific US-based format. It actually validated the field based on that formatting and returned an error message when the format was not followed, but it never openly informed us of the required format.

Date Formats Allow dates to be entered and displayed in the standard format for each country. Fortunately, operating systems allow users to set the format. Software applications should use this default, including both the sequence of numbers and the appropriate separator characters.

Currency Display the appropriate symbol to denote the local currency. Allow the symbol and any special characters to be positioned appropriately in relation to the numbers. There may also be a need for multiple currencies to be represented at the same time, to eliminate the need for manual conversion.

Units of Measure Allow data entry in units of measure that are familiar to the users and consistent with the source of the information, to eliminate the need for manual conversion. For display, consider converting data to familiar units of measure in certain circumstances if doing so would increase comprehension without the risk of miscommunication. Always be explicit about the unit of measure.

Icons and Symbols Review icons and symbols to make sure they are understandable, and replace them with local variants if appropriate. This is particularly important where icons are derived from familiar objects that are country or culture specific (such as mailboxes and tools/implements).

Representations of People Choose representations of people that are acceptable to users in each culture. Gender, race, dress style, physical setting, and nonverbal communication are all issues that affect the conscious and subconscious acceptance of messages. This issue relates not just to photos and video clips but to animations, cartoon characters, symbols, and icons as well.

Figure 1. Obvious Interface Elements.

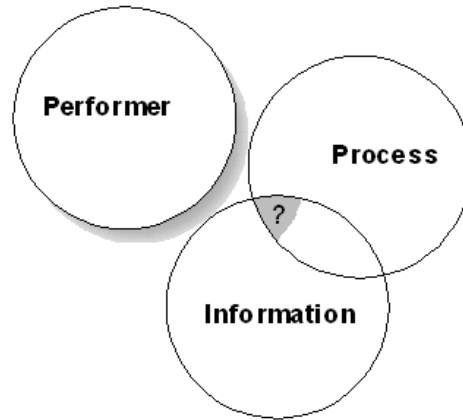


Figure 3. Disconnecting the Performer.

replace it with a different one (see Figure 3). The new circle does not fit because the parts are not interchangeable. To reconnect the performer with the process and information, we have to understand the attributes of the cultural group. To plan ahead for supporting a process with an international audience, we analyze it by asking questions in categories:

Language

- What major language groups and local languages do the performers speak? Where do they live?
- What differences exist between local languages and the major language groups? What meanings are associated with regional dialects?
- What differences exist between spoken and written language?
- What style and tone would people in this cultural group consider credible for a computer application? What style and tone would people consider helpful and friendly?
- In what other ways might the language create a distance between the user and the application?
- If the user will need to adapt to a different dialect, how can we minimize the burden?
- Is there a particular challenge in localizing for tonal languages such as Thai, which require specific written conventions to capture the shades of tone?

Time

- How do performers normally think about and represent time?
- What date formats are considered standard? Do performers deal with multiple formats?
- What day does the week start on?
- What other time periods are frequently used?
- If the application refers to special days or holidays, have the implications been considered for all countries? Will the application recognize the holidays and special days of another locale?

Cultural Expectations

- Are generic designs really generic, or do they contain implicit cultural assumptions?

- Do different ways of reading create expectations about the flow or placement of information?
- Which parts of the screen are likely to be viewed first and last?
- Do habituated skills (e.g., touch typing) conflict with any part of the design? How does the performer's equipment differ from the design requirements? How might those differences affect task performance?
- Are there local expectations about the display of information, such as for proper names? Can the software be configured to match those expectations?

Metaphor and Representation

- Will the performer be able to recognize literal representations in icons?
- Are generic visual symbols going to be familiar? Do they mean the same thing across cultures?
- If spoken language is used, what cultural biases do people have about regional accents?
- Do certain colors have political or religious significance for the performers?
- Do colors have other recognizable meanings for the performers? If so, do these meanings conflict with the intended use?
- Is it possible to configure colors to better match the performer's cultural context?

Performer's Locale

- Is the performer working in more than one country? If so, how does this affect his/her needs and expectations?
- What can be done to make their transition from one place to another as seamless as possible?

Language. Translators should avoid using words or phrases that are too specific to a locale. The choice of vocabulary, including any slang or references to specific places or events, should be as generic as possible within the major language group, so that it will be understood as widely as possible. Words that are unfamiliar or that sound foreign will likely distract users and make the application seem less supportive. The language should convey the appropriate level of authority (for example, policy statements should sound authoritative) and the appropriate tone (for example, helpful hints may be more or less formal in tone depending on the audience expectations). It is worth considering that in many languages (e.g., French), the written language and the spoken language are different—the written language is far more formal. It is important to have a native of the region, not just someone who speaks the language, working on the actual translation to influence design decisions.

Time. Another distinct difference between cultural groups is the way that they think about and represent time. In working with computer systems, users frequently deal with time elements—sorting lists in date or time sequence, entering time periods for queries, receiving reminders of due dates, scheduling appointments, making reservations, and

so on. It is important to make sure that any time elements are designed to be familiar to the target audience, since they are so often encountered. Diverging from local expectations can create significant affects and barriers. For example, we consulted on one project in which users who were responsible for a scheduling task complained about being expected to enter start and end times on a 24-hour clock. Because they were accustomed to a 12-hour clock (clarified with AM and PM), they had to stop with each entry and calculate in their heads the correct time as required by the system.

Regarding date formats, people who work internationally and have experience with different standards for date displays may actually need more help in interpreting dates than those who do not. During one international design session, we noticed that Europeans would habitually write out the name of the month for the first 12 days of the month (e.g., "12 Jan 2001"), but would switch to all numbers when writing the remaining dates (e.g., "13/1/2001"). This was a way of avoiding dates that appeared ambiguous (e.g., "12/1/2001"—is that December or January?). Designers should look for ways of removing this ambiguity without increasing the burden on the user.

Time periods are also different from one culture to another. Probably the most commonly used time period, the week, is not as straightforward as one might expect. In the US, the calendar week starts on Sunday, whereas in other countries it is common for the week to start on Monday. If the labels on a pop-up calendar are not very clear, a user might easily overlook this difference and make an error. Some applications allow the user to search for transactions entered "last week" or "this month" instead of entering start and end dates in the search criteria. Designers must consider the cultural assumptions inherent in these phrases when preparing for an international audience. A phrase such as "the current tax year" would have a different definition in different countries, while a phrase such as "the last fortnight" might not translate at all. Colloquial phrasing has been known to produce pitfalls (consider the phrase "half eight," which means 8:30 in the United Kingdom and 7:30 in some other European countries).

Special days and holidays, of course, differ from one country or culture to another. Even with the growing awareness of diversity and multicultural staff, we would avoid reference to special days if possible. There are exceptions, such as help content, training materials, or performance support tools that need to provide relevant examples and context-specific information to support the user's job. For example, a performance support system for a retail application in the United States would have to mention the Christmas season and the back-to-school season, since these are a key part of the user's work context. However, if this performance support system were made available to an international audience, the content would probably have to be completely rewritten. Other exceptions might be calendars, scheduling

applications, and collaborative software environments, where informing users of a cultural circumstance such as a holiday can be critical to performance.

Cultural Expectations. People in different countries or cultural groups tend to have different expectations of how things work. Their conceptual models are shaped by experience in different environments. Through experience, many behaviors become so skilled that a person no longer pays conscious attention to them; they become automatic. These behaviors are very likely to be culturally bound. If we are good designers, we are already trying to design things that work the way people expect them to, but the possibility of an international audience complicates this task. We believe things that may be considered generic in design are in fact generic only because of the cultural conventions of the designers and current majority of users.

Users may have different expectations about the flow and placement of information on a page. The standard of reading across from left to right is a Western cultural expectation; Semitic languages are read right to left, and the Chinese traditionally read from the top right to bottom left. We do not know how the user's comfort with the software application is affected by a change from the traditional direction of reading and writing. We suspect that even if the user adapts to a seemingly unnatural screen display, their culturally traditional reading direction may still impact eye movement. Generally, we consider the top left of a screen to be the "entry" point, with the lower right being the "exit" point—that is why common commands, controls and titles are top left, and often buttons (such as "Next") are placed toward the bottom right. Thus, cultural background may influence user performance with directional actions and may suggest the optimal placement of controls on a screen. We hope that in the future, the eye tracking research being done in the field of human-computer interaction may be able to clarify this.

People in different countries also use equipment that works differently. A prime example of this is the computer keyboard, which has a different layout in different countries. This could potentially affect performance for users working with software that was originally designed for a different keyboard—special characters or control characters may not be there. This is especially challenging for touch typists who are forced to move from one keyboard to another. For example, an American who was recently in Eastern Europe told us that she had been surprised to find a keyboard missing the familiar "@" symbol—to type this symbol, she had to learn the key combination "ALT-64." Some of the differences affect the keyboard shortcuts that designers provide to speed up tasks for expert users. For example, Figure 4 shows the different placement of the CTRL+Z key combination on the German (and various eastern European), French, and US keyboards, which could easily lead to errors.

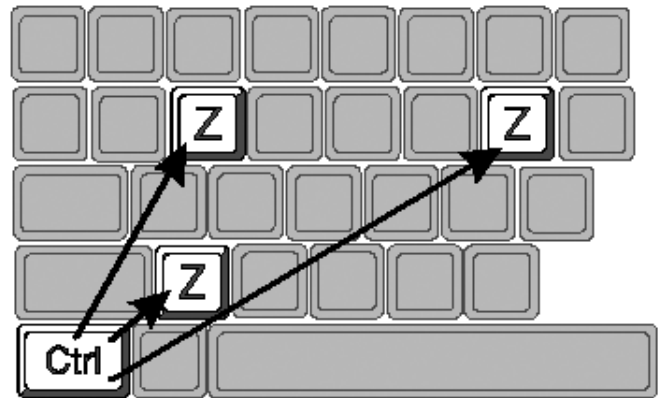


Figure 4. Variations in Key Positions.

Another performance problem with different keyboards is the potential lack of special characters needed for correct spelling of proper names. For example, Americans working in Bosnia said that the original Bosnian voter registration was done several years ago by international observers who did not correctly enter the special Slavic characters in people's names. There are two different accented versions of the letter "C" and three versions of the letter "S", but the internationals simply typed "C" and "S" (probably using American keyboards that did not contain the Slavic characters). This caused significant problems in subsequent elections, when Bosnian voters returned to the polls and were told that they were not registered to vote, because their correct names could not be found in the database. The voters were angry, and much time and effort were spent re-entering voter records.

Metaphors and Representation. Another important difference between cultural groups is their understanding of symbolism, which encompasses visual icons, sounds, colors, and metaphors. Some symbols that we consider generic are actually a matter of cultural convention. For example, a large-scale computer-based training (CBT) initiative in a multinational corporation had some symbolism that led to significant performance problems. The training department was surprised to find that while Americans were scoring high on the CBT, most Germans were failing. On investigation, we saw that the CBT required users to click on a checkmark icon for "yes" and an "X" icon for "no." In Germany, the "X" meant "yes," so many users were selecting the opposite of their intended answer. Other symbols that represent real-world objects may not be understood across cultures (for example, a literal representation of the US standard manila folder may not be recognized in other countries). With the rapid increase in web use, people are creating their own icons and mixed icon/word controls. This increasing divergence from any de facto "standard" could be a precursor to greater localization of icons (see Figure 5).

Color can convey different meaning in different cultures. It can also be unwittingly loaded with meaning. For example, in

Bosnia red is the color of the Serbs and green is the color of the Muslims. Any use of those colors can potentially stir up feelings that an outsider would not have anticipated. However, color tends to convey meaning in context, so it may be dangerous to generalize too broadly about its use. For example, in a developers' meeting, a requirements analyst cautioned the group that the use of the color red in error messages might not be good for internationalization, since the Chinese associate red with happiness. The two native Chinese developers in the meeting raised their eyebrows in surprise. One of them said, "Now I understand! This must be why the Chinese drive their cars with the tachometer in the red, because they feel so happy...until the engine stops working!"

Metaphors that are commonly understood in one culture may not translate well to another, which poses another design challenge. People who work in the performance support arena will recognize that the "coaching" metaphor has been widely used to suggest a friendly expert guide. This metaphor is frequently reinforced by images or words that tend to center around a specific sport. For example, we have seen numerous "Coach" buttons decorated with baseball cap icons and at least one performance support tool that made extensive use of the football metaphor, including green "grass" with marked field lines.

Performer's Locale. We have seen in many applications the assumption that the performer is situated in only one country. However, this is not always the case. An increasing number of people live and work in more than one international location. There is a need to provide the flexibility to support them, so that their transitions from one place to another are as seamless as possible. For example, we noticed that a popular consumer website holds a single pro-

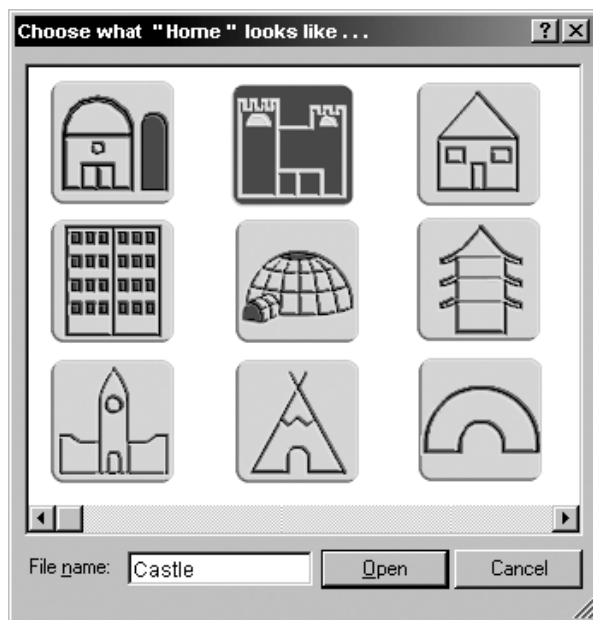


Figure 5. Customization of the Future?

file (multiple addresses, credit cards, etc.) for a person who lives and works in both the United States and the United Kingdom, but cannot distinguish where to ship from or what to charge based on the shipping address selected. The .com and .co.uk sites are different in all respects except the user's personal information, but lead the user to believe they are connected.

Disconnecting the Process

When preparing a product for an international market, it may not be enough to convert the existing product to a foreign language version. Even if the information is translated sufficiently to make it understandable, the processes themselves may not make sense to the performers or may not adequately support their needs. This is because ordinary tasks, standard practices, and work environments may differ significantly from one country to another. The process becomes "disconnected," diminishing job performance (see Figure 6). To plan ahead for supporting a process with an international audience, we analyze it by asking questions like these:

- How is this process normally done in each locale? Is it done at all?
- Is the goal or the desired outcome of the process any different in another place? Are there different standards for achievement or definitions of "success?"
- What other processes or tasks are viewed as related?
- Is the expected sequence of tasks different from one locale to another?
- How much control do people expect to have over this process? Do people expect to be guided or told what to do?
- Does the process consist of high-context or low-context tasks?
- How much does the process depend on policy? What is the source of that policy?
- Is the process affected by differences in working practice, such as work days or staffing?
- Does the process involve input from multiple countries?
- Does the process require communication among people who speak different languages or work in different countries?
- How is the process structured in the current application? In similar or competing applications? What assumptions are implied in that structure?

There are differences in standard working practices from one country to another that may affect how well a particular software application fits into the local environment. Designers should gather as much information as possible about the tasks and the work context in other countries to identify the types of differences that can be expected. For example, in some European countries it is typical to take a two-hour lunch and extend the working day later into the evening. A staff scheduling software application designed in the United States would need to accommodate this prac-

tice to fit in. The number and types of staff in the work environment may also be different. A German in a multinational corporation noted that the company's internal project staffing processes assumed a much larger and more hierarchical staff than his branch of the company actually had. One person typically took on several roles.

There is also the need to comply with different regulations in different countries. For example, a large consulting firm had difficulty rolling out a training system to its European branches because the system did not capture course completion dates and status in the way that was necessary for ISO9000 compliance. Similarly, any software purchased by the US government is required to comply with Section 508 standards for accessibility to people with disabilities. It is useful to know about these standards and practices as early as possible to make sure that the design and the business rules are configurable enough to accommodate them.

Because work is done differently in different countries, task analysis becomes very important. The expected sequence of actions, the size and scope of the task, the inputs, and the standards for success may be different. This becomes an important challenge for performance support, where the purpose is to guide the user through the task to achieve a successful outcome. For example, we sometimes introduce a sequence as a way of supporting a novice user. However, if this sequence runs counter to the local organization's expectations of how the process is normally done, it may instead increase confusion. There may also be an expectation of the degree of control that a user would want to have over the sequence. In cultures that value individuality, users may want to exert more control over their actions, while in other cultures, users may not have this expectation. There is also a need to anticipate which tasks are considered related. For example, in the US we might think of "change my address" and "change my direct deposit" as related tasks, because when people move from one place to another they are also likely to change banks. This is because in the US the banks are statewide, not national. In many other countries where there are national banks, these tasks would not be thought of as related because when people move their bank account stays the same.

We recognize that this sounds daunting. It is not practical or desirable to revisit all the tasks in a software application when preparing for an international market. To focus and prioritize this effort, we have started to identify ways of categorizing tasks as "high-context tasks" or "low-context tasks." In the translation field, others have written about high-context and low-context cultures/languages.

A high-context language is one in which there are many references to things that are not spoken directly but are assumed to be understood by all speakers (Hoft, 1995). Similarly, we have defined "high-context tasks" as those

that are more culturally bound. When we review an application, we look for high-context tasks that will need more attention to localization, low-context tasks that can be internationalized, and rules that should be made locally configurable. For example, in a training management application, the task of registering for a training course is low-context, and we would focus on making it generic enough to be understood internationally. The tasks of creating development plans and mapping learning to specific job requirements, on the other hand, are high-context, because they are tied to cultural expectations about career advancement, employment duration, incentives, continuing education, and local policy. A third task, getting manager approval for a course, is medium-context, because although it is related to cultural expectations, it is more closely tied to internal organizational rules, and must be configurable.

Although supporting the work process in another country is a primary goal, we must not forget that it is sometimes necessary to accommodate multiple countries or cultural groups at once. For example, it is not enough to support the local currency if people need to use multiple currencies as part of their standard work. A few years ago, a review of small business accounting packages showed they could be configured to accept input in one of several international currencies. However, once the user had gone through the setup process and selected a currency, that currency remained the default, and there was no ability to enter amounts in other currencies. This was a problem for international businesses, because the users were forced to manually convert the currency amounts. Similarly, in collaborative work environments it is important to support multiple language groups, because project teams and design groups are increasingly made up of international participants working in different places, different time zones, and different languages.

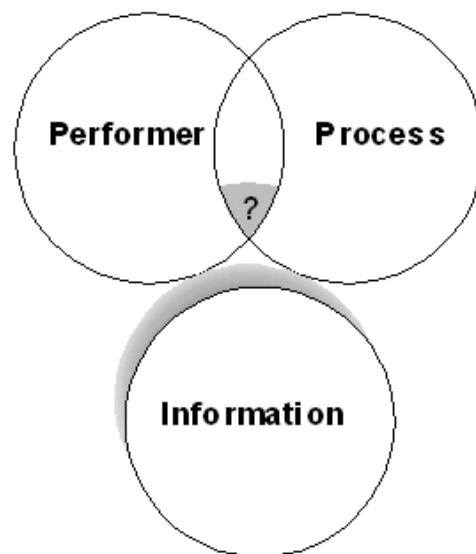


Figure 6. Disconnecting the Process.

Disconnecting the Information

Preparing information for an international market is not limited to the language translation. It also includes making sure that the content is relevant and understandable. In terms of the three-element performance model, even if the process is understandable to the performer, the content may not make sense. In this case the information becomes “disconnected,” diminishing job performance (see Figure 7). To prepare information for an international audience, we analyze it by asking the following types of questions:

- Does the information support the task?
- Can the users read the information in the language(s) provided?
- Is the meaning behind any symbolic information understandable?
- Does the layout seem clear or intuitive to users in different cultures?
- Can the users find the information, and what cognitive strategies are they likely to use?
- How much context is currently provided in alerting or feedback mechanisms? What assumptions are implied in those messages?
- Will any examples make sense to people in a different country or culture? Is it necessary to write different examples to make the point or support local guidelines, rather than translating the words?

The basic issue is the ability to read the information. Imagine that you have just gotten off an airplane in Kuala Lumpur or Delhi. The process of finding your way around an airport is familiar to you as an experienced traveler, but there is no easy way to puzzle out the written language. The “data” on the signs have no informational value to you. This is exactly the situation that prompted the development of international symbols for the physical world. These symbols have become at least recognizable enough to provide the informational value you need in that context. There is not yet an equivalent symbolic “language” for software. To some extent, this is what icons attempt to do, and some succeed by default if not by design, because certain software is now widely used around the world.

Users in different cultures apply different strategies to finding information than the familiar “searching” techniques that we find on the web. Searching is not the most effective method, but other strategies are affected by the way that people think about and categorize content, which varies from country to country. For example, the Library of Congress cataloging system is applicable only in the United States; other countries may have their own standards or no standard.

Users’ expectations of the layout and organization of information may also be different. For example, in France, reference books have the table of contents in the back rather than in the front.

Although the content organization depends primarily on the type of content and the situation in which it will be used, culture may also have an effect. For example, in Japan it would not be unusual for a user to take a manual home and read it before working with an application for the first time, while in the United States, people prefer to scan a quick-reference card and figure it out as they go, using task-specific help if they get stuck.

It is important to review warning messages and feedback for cultural relevance. The instructions that accompany a warning and specify “appropriate” action may need to be revised for different audiences. The required action (e.g., contact the help desk, or policy-specific information) may be referring to an infrastructure that is not there in a smaller country. Thus, warning messages cannot afford to be too rigid. This may be the case even when common international practices are the goal, such as when carrying out multinational drug trial research. The standardized software tools must still communicate effectively to local users.

When reviewing content, it is important to notice any local references and examples that rely on cultural context and that may not translate. We know how valuable it is to provide examples in performance support, but we must be aware that an example that makes sense in one country may just confuse users in another. For example, an application used in the insurance industry would need examples based on state regulations. However, if an insurance company were multinational, it would need completely different examples to reflect the different types of regulations that exist in countries with a single national regulation. As designers, we need to build this flexibility into the information architecture so that it is easy to pull out an example and replace it with something more appropriate.

In the past, many software companies treated localized versions of information as a secondary concern. Localized documentation and user interfaces are often released months after the initial language version, which has an effect on product distribution and support. For example, we recently saw a product that was available on the Internet for downloading, but only in English. There was a notice asking people to check back in a few months for the western European language versions. We suspect that many potential customers would forget to come back and look again. If they decide to download the English version, they may not be able to get support from local staff. For example, European users in a multinational corporation received a new version of a software package and began calling their local help desk with questions, only to find that the help desk staff had not yet received instructions on the product in their language. It affects productivity and credibility when the support staff cannot meet their end users’ needs because they do not have adequate tools available in a language they can understand.

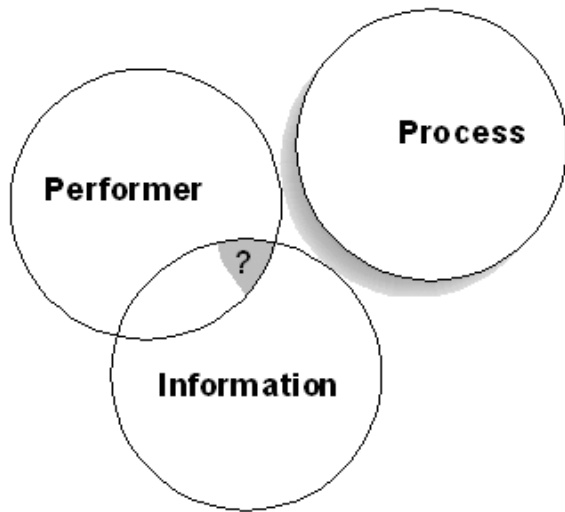


Figure 7. Disconnecting the Information.

Conclusions

Over many years of experience in the software development field, we have seen how easy it is to incorporate personal bias and local assumptions into the way that software works. Breaking those habits is certainly a challenge, one that we increasingly must meet. While we, like most designers, may have more questions than answers, there are a few things we can conclude from our explorations into internationalization. In fact, most of these principles are also general principles of good design—when they are done well, we have found that they contribute greatly to the internationalization effort.

- Question your assumptions. Avoid thinking or saying, “...But everyone knows...”
- Plan to be international from the start, and analyze the users and the environment effectively.
- Don't expect it to be easy if you do it right.
- Design applications that are configurable and modular. You can tailor business rules and formats more easily to working practices in other parts of the world.
- Look for ways to remove barriers to performance by designing for simplicity. The software should be transparent, it should not draw users' attention away from what they are doing.
- Create a content architecture that allows content types to be tagged (e.g., procedure, example) and allows examples to be replaced with localized variants. When designing performance support materials, include placeholders for local content and allow customers to add or modify content that is meaningful within their culture and organization. It may be impractical for the software provider to attempt to create all the content.
- Focus on tailoring support materials to the culture: If the software offers support, help, and advice, that support should be targeted to the user's context. Otherwise, it will increase frustration by giving information that does not make sense. The users will begin to distrust the help

because it does not relate to their needs. It is better not to provide support materials at all than it is to provide inappropriate support. 🗨️

References

- Cross, J. (1999). Internationalize with localization. [Online], msdn.microsoft.com/library/periodic/period99/vjp9922.htm
- Dickelman, G. (1996, September/October). Gershom's law: Principles for the design of performance support systems intended for use by human beings. *CBT Solutions*.
- del Galdo, E.M., & Nielsen, J. (Eds). (1996). *International user interfaces*. New York: John Wiley & Sons.
- Ebben, S., & Marshall, G. (1999). The localization process: Globalizing your code and localizing your site [Online], www.microsoft.com/TechNet/Analpln/global.asp
- Hoft, N.L. (1995). *International technical communication: How to export information about high technology*. New York: John Wiley & Sons.
- Ishida, R. (1997). Challenges in designing international user information [Online], www.xerox-emea.com/globaldesign/paper1.htm
- Norman, D.A. (1998). *The invisible computer*. Cambridge, Massachusetts: The MIT Press.
- Sun Microsystems, Inc. (2000). I18N Verification checklist [Online], www.sun.com:80/developers/gadc/i18ntesting/checklists/index.html
- W3C (World Wide Web Consortium). (1998). 18n/110n: LISA—Internationalization/localization [Online] www.w3.org/international/o-lisa.html

Duane Degler has more than 15 years' experience in organizational performance improvement, including user-centered and performance-centered system design, information and knowledge strategy, training, international system implementation, and multimedia. He currently consults on performance-centered design and knowledge management. Duane has lived and worked in Europe and the United States. He may be reached at ddegler@ipgems.com.

Lisa Battle specializes in user interface design and usability for commercial software (for which she has won a national award) and client-specific applications. Lisa has been working in performance-centered design and technical communication for more than 10 years and is currently a lead designer at Lockheed Martin. She may be reached at lbattle@acm.org.