

te testing experience

The Magazine for Professional Testers

printed in Germany

print version 8,00 €

free digital version

www.testingexperience.com

ISSN 1866-5705

Agile Testing

أهلاً وسهلاً
BENVENUTO

ברוכים הבאים
ISTEN HOZTA

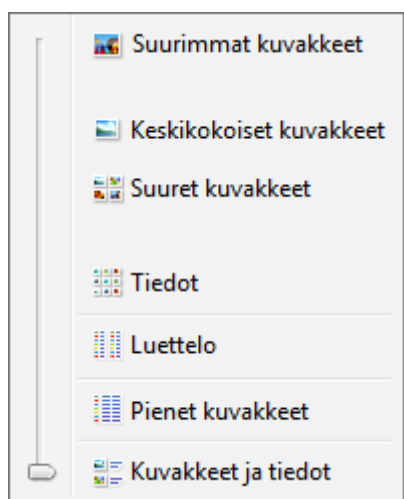
ДОБРО ПОЖАЛОВАТЬ
TERVETULO

The Importance of Language

by Michal Kolář

Introduction

Take a look at the following screenshot and answer the question: “What is wrong with this well-known menu?”



Do you think perhaps that menu items two and three (*Keskikokoiset kuvakkeet* meaning “Medium icons” and *Suuret kuvakkeet* meaning “Large icons”) are not aligned well and have too much space around them? Well, believe it or not, these layouts are not wrong, or at least this layout truly reflects the original English-language layout used in Vista. Give it another try. Still nothing?

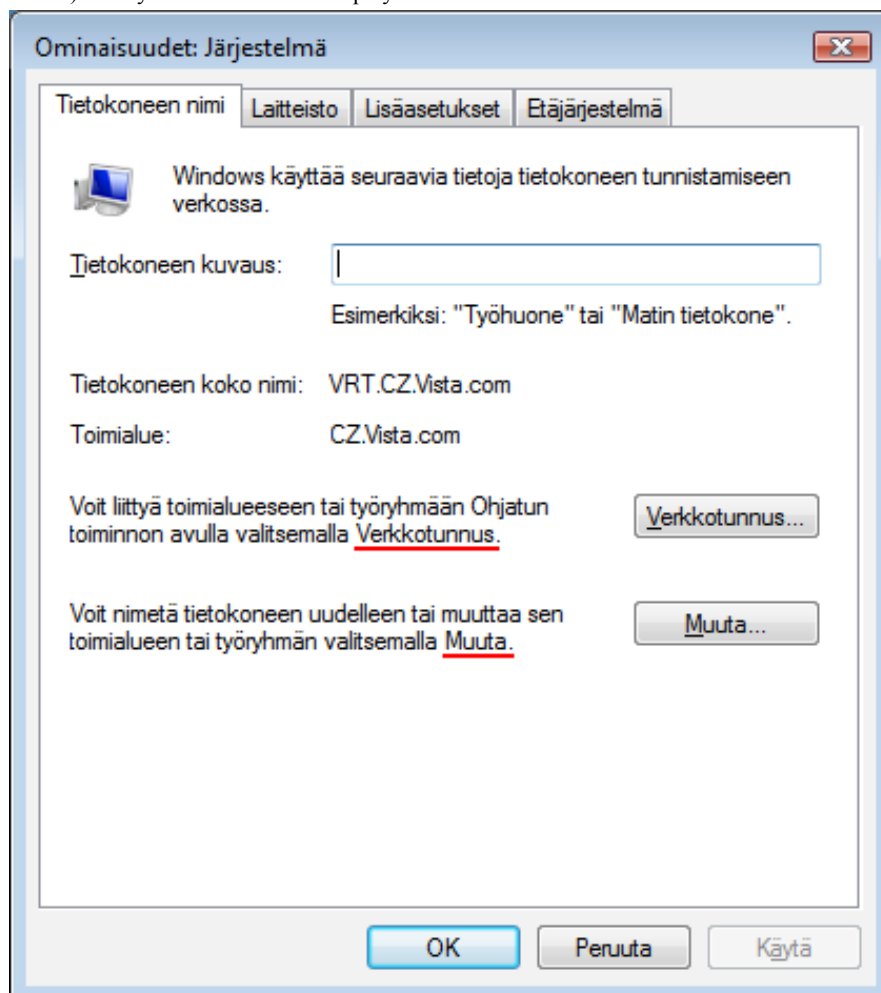
Unless you were born in the Land of a Thousand Lakes (or learned Finnish somewhere along the way), you have no chance to discover the problem. The problem is that some text labels are transposed – their text is swapped with other labels while the icons remain in their original positions. So, the text for “Large icons” is next to the medium-sized icons, and the text for “Medium icons” is next to the large-sized icons. There is almost nothing that regular functional or localization testing can do to detect such an error, but – as always – there is a way to deal with this kind of problem: **linguistic testing**.

You might be a world-renowned producer of a large software application which is localized in twenty languages, or you might be a stand-alone developer with a great small tool and you are thinking of localizing your application to break into global markets – in both cases linguistic testing is something you should definitely be aware of. It’s not enough to have your translated resource files go through a QA process by another linguist (though, having translated resource files reviewed by someone who didn’t work on localizing in the first place is a good idea). Many UI elements are displayed

only in the live compiled software and may actually require some significant infrastructure to be running before they appear. You need to see these live and in context, just as the menu displayed above.

What linguistic testing is and what it is not

Linguistic testing is sometimes confused with localization testing. You can visit Wikipedia and search for definitions there, but let me offer really simplified definitions of each:



- **Localization testing** focuses on the correct functionality, appearance and completeness of the localized product.
- **Linguistic testing** takes care of ensuring the correct language rules are being used and focuses on correct in-context linguistic usage.

The border between these two types of testing can be a bit blurred and there are several cases where a localization test engineer can easily report a problem that would normally be expected to come from a linguistic reviewer. And vice versa.

In this typical dialog box, the button names the user is expected to click on if the described action must be taken are noted (here, *Verkkotunnus* meaning “Domain” and *Muuta* meaning “Change”). But the typical software localization process may not be straightforward. The individual UI elements may come from different resource files; some may come localized by another translator, some may be recycled from a previous version of the same product and “locked” for any changes. That’s why verification of the consistency with the real button label is something which is expected from both a localization test engineer and a linguistic reviewer. An inconsistency within UI (such as button label conflicting with information in text) may be introduced easily in the localization process, and in practice occurs much more often than one may think. The value of a linguistic reviewer in such cases is much higher for languages that are perhaps too “exotic” for the localization test engineer to be expected to notice a possible error: for example Korean for a European tester or Hebrew for a Chinese test engineer.

Depth of linguistic testing

There is always a dilemma that must be decided at the very beginning of a software localization project – you need to know what level of detail is important to you and where to draw the line of not needing to take care of every single detail. To ensure linguistic quality, there are basically two different approaches that affect how testing will be completed, the target quality, and of course the cost. Those two approaches are:

- A. In-context review** – With this approach, the reviewer just skims the dialogs boxes, menus and pages and verifies that the localized content corresponds to the expectations of a user. For example, in an e-Learning course for children, there might be a picture of a sauceman; the reviewer verifies that the content next to the picture is really about cooking and not, for example, about African frogs.
- B. Linguistic analysis** – In this approach, a



professional linguist goes through all the strings displayed on the screen and carefully observes all language aspects and verifies that the linguistic conventions regarding consistent and appropriate use of language, proper use of terminology, correct style and tone are strictly followed, and that the localized text corresponds to the correct functionality of the UI.

To decide which approach is best for a given project, we need to think of the real users. Who they are? If the software application is an automotive diagnostics analysis tool, then it will most probably be used by people who do not care about every single language aspect followed to the letter and therefore it might be appropriate to choose an in-context review approach. This type of review can be a more cost-effective approach compared to a more thorough linguistic analysis, which should always be considered when testing educational software, retail applications, etc.

Each approach requires different testers. As outlined above, the in-context review can be more cost-effective, because this type of testing does not necessarily have to be done by linguistic professionals or by native speakers of the target language. It is a perfect job for part-time employees, company employees who study the language in question or have experience with the foreign country (through marriage, work experience, etc.), or, even students from local universities who study and like the language.

On the other hand, for the more thorough linguistic analysis you usually need “the best you can get.” You might think, “Great, I’m in luck here” if the language that you would like to do linguistic testing on is your mother tongue (or someone on your team’s), but hold on. Are you absolutely sure that you understand every single aspect of the language? Are you really enough of a linguistic expert? Personally, I was pretty good in school with languages, but now as a computer testing guy I would be very afraid of using the remnants of ten-year-old knowledge to test the linguistic quality of a product – even in my native language.

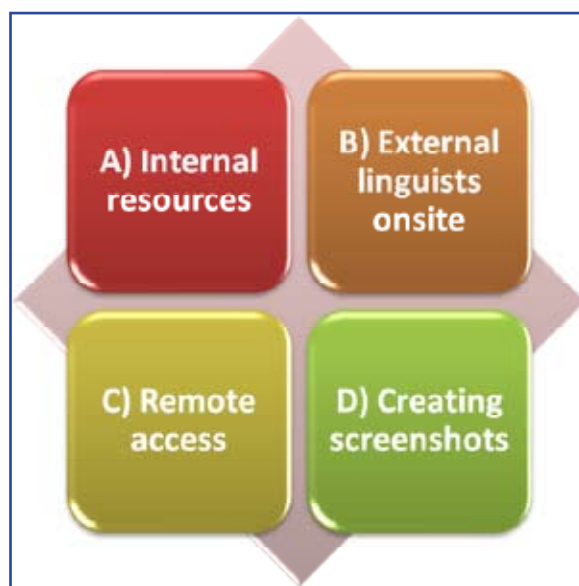
In most cases, though, a native speaker is wholly appropriate for linguistic analysis. Only in very special cases (such as testing software for, let’s say, post offices or any other public institution), where 100% correctness is an absolute must, a native speaker with a linguistic degree must be found. But, be ready for the fact that such people have considerably different expectations to students when it comes to how they expect to be compensated for their work.

It is often the case that there are limited resources for less

widely-spoken languages and there might be a situation where you might have to consider using the same person who translated the product to do the linguistic testing as well. This does not necessarily have to be a problem, and for an in-context review this solution is fine. However, for detailed linguistic analysis it is definitely not suitable, and, the added value of such testing with the same resource is very poor.

Picking the right approach

There are some “best practices” that can be followed in order to perform linguistic testing. No matter if you do it yourself or if you ask a vendor to do it for you, there are basically four approaches that can be considered when it comes to linguistic testing:



- A. There are internal resources within the company** – This is probably the most comfortable solution, but it may not necessarily be the cheapest one. Maybe your senior developer can speak Hindi and would be willing to help out with testing. However for in-context review you would be just fine with a student who would be happy to take on the task at a cost of, say, 30 per cent of your senior developer’s cost. The student – even with some management costs – might be the more cost-effective solution.

There can also be a situation where you need to perform linguistic testing on ten languages in total, with only one language covered internally by your employees. Think twice about whether it is good to split the work and test each language in a different way. The overall efficiency may suffer and you may find out in the end that utilization of one of your internal employees did not actually pay off.

- B. External linguists brought onsite** – Probably the most reliable solution and one where you have full control over the testing process is to have linguistic resources come onsite. With onsite linguistic testers, you can adjust project

details as soon as issues appear and you can also minimize downtimes by taking advantage of proximity to resolve any problems as they are noted. However, this type of testing is extremely time-intensive in the planning phase and requires a rock solid project plan that predicts in advance that the work will be ready for testing on a particular date and no later (because you need some time to contract all those onsite linguists.)

For example, if you need to coordinate ten external linguists for ten languages to perform five-day linguistic testing onsite at your premises, you should definitely have at least three different testing timeslots accounted for in your planning, and try to allocate groups of linguists in those timeslots. It is impossible to agree on a single testing timeslot that would suit all the linguists as well as your production team.

- C. **Setting up remote access** – Sometimes it is not possible to bring linguists onsite. The nature of your product might mean you don't need "top secret" approaches, and it's acceptable to allow for remote access. Or, you may simply be just out of office space. A solution is to set up the right infrastructure and grant remote access to your linguists.

This is a good approach for projects where you need to test very specific languages or you need to test a large number of languages and there is no chance to bring linguists onsite. It also gives you more flexibility, as remotely-connected linguists are more willing to accept some changes in the schedule should tasks slip. With this approach, you miss the benefits of direct support and you must rely more heavily on the linguists to do exactly what they should. Sometimes this approach cannot be applied because of technical reasons (too complex infrastructure, target device does not support remote connection, etc.).

- D. **Creating screenshots** – This final method is very suitable for technically demanding products where it is not possible to bring all linguists onsite. You can capture screenshots of all the screens you want to test and send them to the linguistic testers. The screenshot approach still provides some context to the testers, which is sufficient for full understanding. It is also possible to create a visual screenshot map so that the linguists can clearly see what actions must be taken to get to that particular dialog/menu/page and therefore increase awareness of the context for the linguists. Screenshotting is always a quite time consuming activity, but often can be automated by using automation tools and preparing a script which is later run on all language versions, generating screenshots in a few minutes (or hours at most). Here, the more languages you

are testing, the better. Preparing automation scripts for one or two languages may not pay off, but for ten or more languages the situation is certainly different.

Preparation of screenshots requires more work from the technical team but eliminates confusion with more complex products where problems can be expected if the product is tested using remote access. However, for projects with a very large number of screens that need to be checked (200+) it may not be the perfect solution, as the number of screenshots may cause confusion for the linguist and special attention must be paid to the organization of the project.

What else needs to be done?

No matter what method is chosen, and no matter what quality level you targeted, it is always essential that the reviewer understands the whole concept of the tested product; that he/she knows what it is used for; what are the key features; what has changed since the last release; who is the target user of the product and any other things that may help him/her to understand the background. This level of detailed understanding is very important for the reviewer to be able to verify the context consistencies, and it is vital for the linguistic testing approaches where reviewers are not onsite, or when they are working outside of your normal business hours.

Sometimes there is a push to combine the roles of linguistic and localization testers and have everything done by a single person at the same time. Tempting, but difficult to fulfill. Linguistic reviewers are usually liberally-educated, while localization engineers are typically more technically orientated. There is a chance of combining localization testing and the more simple in-context review, but as mentioned above, the more languages that are to be tested, the more complicated this approach is.

Another important thing is to clarify workflows and set firm rules before localized product testing is started. A list of detailed instructions must be prepared so that the reviewers can clearly see what they should verify on each dialog box, menu, etc. , and just as important, what they should not bother with. A number of organizational things must be clarified as well: Where to log defects? How to report the progress? Who is the contact person? Another aspect is to define the process to follow later when the defects are fixed and verifying that the fix has made it to the final product. Is it still necessary to have a language specialist review the fixed element again, or is it enough that the defect really seems to be gone?

Conclusion

Linguistic testing of a localized product may not be an easy undertaking. There are several aspects that complicate matters and it is often inefficient for development companies to "do it on their own."

There is no "best way" that can be recom-

mended from the scenarios outlined above. Each has some advantages; each has some disadvantages. The most suitable solution needs to be selected based on the project type, schedule, budget and company preferences.

There are many localization companies that offer this type of service and are able to deliver everything as a single package, and this is probably the future of linguistic testing – it is something that is performed by specialized companies with a strong tradition in providing linguistic services. This is the right approach for anyone who cares about quality. Increasingly, more and more organizations, large and small, recognize that the model "we'll give it to some of our local representatives" does not meet their needs and therefore they seek professional localization companies that deliver professional quality at a reasonable price.



Biography

Michal Kolář, Test Manager at Moravia Worldwide's Testing and Engineering business unit, joined the company in 2000. Michal started off with the company providing localization testing quality assurance for Moravia's key software clients, later assuming responsibility for testing quality control processes. In his current role, Michal is manager of a testing unit at Moravia headquarters in Brno, Czech Republic, where he maps and monitors production processes applying Event-Driven Process Chain (EPC) methodologies. He has also initiated the implementation of new technologies such as virtualization. During his career, Michal has prepared numerous testing and QA training courses and has trained hundreds of test engineers. He is fluent in Czech and English, with a working knowledge of Russian.