The double whammy of game localization

Jacob Stempniewicz

Jacob Stempniewicz is VP of marketing at localization provider Andovar. He speaks four languages and has lived in Asia for over a decade.

Game localization suffers from a double whammy. It combines some of the hardest challenges of software localization with the need for creativity often reserved for literary translation. At the same time, some consider it a less serious type of work “because games are for kids.”

As a result, game localizers don’t always get the recognition they deserve.

But are games really just for kids? Anyone with actual exposure to the gaming industry knows this to be false — things have changed tremendously since the time of Pac-Man and Super Mario and so has the typical gamer. According to US Entertainment Software Association (ESA)’s "Essential Facts" report from 2015, the average game player is 35 years old, while women age 18 or older represent a significantly greater portion of the game-playing population (33%) than boys age 18 or younger (15%). Big-budget games cost as much or more to produce than top Hollywood blockbusters, and the gaming industry makes more money than the music and movie industries put together.

With that out of the way, let me explain the double whammy game localizers face.

The whammy of software localization

Games are software, so it should come as no surprise that widely-known software localization issues also apply to game localization. Most of them could be avoided or mitigated if the game development team had someone with game localization awareness or with knowledge of foreign languages, but that is rarely the case. Some of the most common are:

Separating text from code. This is the starting point of any localization project. Before everything else, you first need to export all content that needs to be localized while
leaving actual game code intact. If translators are sent files containing translatable text mixed with code, they are sure to change or delete code elements by accident, which usually breaks the game. Tools can automate this task to an extent, but if the game developers did not consider localization from the start, it can still be very problematic. Afterward, you should be able to insert localized content back into the game.

**String concatenation problems.** Little has given localizers more grief than string concatenation. Anyone who’s ever encountered them will shudder at the term. A quick explanation for everyone else: “to concatenate” means “to link together” and in software development it applies to composing strings from dynamic elements (variables), for example: “[character-name] has reached level [lvl-number].”

This works great in English, but squeezing languages with more complex grammars into such constraints tends to be difficult if not impossible. Changing how strings are composed (concatenated) in different languages is usually out of the question and localizers try to come up with workarounds, which don’t always look good.

**English-centric design.** In the English version of the game you may find sentences such as “You have won!” What can be simpler to translate than that three-word statement? The problem is, in many languages the wording will change depending on the gender and number of addressees, which means it’s impossible to give one translation covering all situations like the English one. English grammar is comparatively simple and lacks features common in other languages, such as complex verb conjugation, noun-verb-adjective agreement, declension of nouns, strict formality levels and many others.

**Terminology consistency.** In both software and game localization, consistent use of terminology in all languages is crucial. You can’t call a spell “Lightning Bolt” in the game manual, “Bolt of Lightning” in the user interface, and “Electric Shock” in a quest description. This would not only make a poor impression on players, it can confuse them and impede their progress. A famous example comes from the English translation of *Street Fighter 2*, where players were told they “must defeat Sheng Long” to progress in the game. Little did they know that “Sheng Long” is a mistranslation of “Rising Dragon Punch,” a special move the player needs to use rather than a (nonexistent) new opponent. The best way to manage terminology is through termbases where important terms are stored along with descriptions and examples of usage.

**Text expansion.** When translating from English to other European languages, you are likely to end up with strings too long to fit in the layout created for English. This is because they often require more words, or longer ones, to say the same thing. When translating from Chinese, Korean or Japanese into English, the problem becomes even more severe, since these scripts are not alphabetic and pack a lot of meaning into each character. The common suggestion to simply “keep the length of strings the same in translation” may sound reasonable at first glance, but turns out to be almost impossible without ugly shortening of words or removing parts of text. A better solution is to adjust the sizes of buttons and fields, or even localize the main user interface elements early on, so game
developers can design the layout accordingly.

**Hard-coded text.** Saving images with uneditable (also known as “hard-coded”) text may seem like an innocent practice, but becomes a royal headache when you need to localize. In the flurry of development, especially if it was handled by various teams or outsourced, source files can be hard to locate. However, without files where text is on a separate layer from the graphics, you’ll need to manually edit the images or even recreate them from scratch. These solutions take more time, cost more and the results don’t always look good.

**Testing.** Unlike document translation, localization of software and games requires in-depth testing of the localized version before it can be published. This means following a structured test plan to review all areas of the game to check for translation that’s incorrect when seen in context, strings going out of bounds and broken functionality. In addition to that, you may want to check for display issues on devices with different screen sizes and resolutions (such as smartphones), compare different versions of the operating system, make sure the game works fine for users with low internet bandwidth or high latency, as well as a score of other issues that can only be found in real-life gameplay. In the case of expansive, big-budget AAA titles, testing can take months and requires specialized software for bug-tracking and reporting.

As if the above weren’t enough, there’s also the second whammy.

**The whammy of creativity**

Games are so much more than just software, and their purpose is not simply to get a task done. Rather, they are interactive stories trying to immerse the player in a new world full of adventure and fun. Granted, some titles are more ambitious than others in this regard, but games have a capacity for immersion surpassing even that of movies or books. It can be maintained in a localized version, but only if the wheels of imagination are not derailed by display issues or awkward word choices. Here are some ways this can be tackled:

- **Game familiarization.** Many games are set in a fantastic universe populated by imaginary creatures. It’s difficult to start translating such content until one understands the game world and the rules that govern it. Translation teams should have time to familiarize themselves with the game and reference materials such as story outline, concept art, design documents, gameplay videos, character bios and so on before translation starts.

- **Context.** A sure-fire way to get nonsensical translations is to send translators nothing except a spreadsheet with rows and rows of strings and a column for “target language.” Instead, strings should be grouped in a logical manner (Menu, UI, Dialogues, Items) and ideally contain annotations describing their meaning in the game. Isolated words can often be translated in different ways, and it’s impossible to choose the right one without context. Such issues will not be found by a spellcheck or an editor looking at a list of strings, but will be discovered once they’re imported into the game.

- **Cultural adaptation.** A game that’s very successful in its original market may struggle in others, even if the translation is accurate. This touches on the broader meaning of localization; not just translating something into a new language, but also adapting it to the new market in other ways. This is most commonly needed in the case of games developed in Asia and published in Western countries or the other way round. Another issue is country-specific cultural sensitivities, such as Germany’s ban on Nazi symbolism in gaming. The type of work required to make a game successful in a completely new cultural environment goes far beyond what’s needed in other localization projects.

What’s worse, you have to be ready for a possible backlash from fans protesting that the game was changed too much! An example of this is the fan outrage at the localization of a game called *Fire Emblem Fates* from Japanese, which they felt amounted to censorship and robbed them of the original experience.

**Challenge accepted?**

Game localization can be a very serious and demanding undertaking. It can not only rival, but even surpass the challenges found in other types of localization. I hope that those who accept the challenge will step into the ring aware of the difficulties and well prepared for a tough fight.

---

**Game Localization and Global Publishing**

LAI offers game market analysis, localization and publishing services to help game developers identify the best markets for their games and publish them globally.

Established in 1993, LAI has an extensive network of gamers, local market experts and partners in all major gaming markets worldwide.

**LAI Global Game Services**

info@lai.com • www.lai.com

Silicon Valley • Beijing • Tokyo