

Lost in Translation:

Exploring Challenges in Software Localisation

Ethan VanderLugt
Penn Roberts

Abstract

Computers have fundamentally shaped the modern world. As applications make their way around the globe, providing software in a user's native language is an essential focus in software development. It serves a profound cultural purpose and acts as a critical accessibility measure. However, the global expansion of software presents massive, interconnected challenges for developers and linguists, proving that treating localization as a simple afterthought is a recipe for disaster. By examining historical precedents and modern development frameworks, this paper demonstrates that true localization requires developers to consider internationalization as a main focus.

Table of Contents:

1. A History of Localisation
2. Language as Accessibility
3. Suddenly... Linguistics!
4. Robots Failing to Capture the Human Spirit
5. Breaking the (Text) Wall
6. Jumping to Conclusions

Introduction

- Users who do not know the language software is written in cannot readily use said software.
- Ergo, to increase the potential user base for one's software, one must undergo the localisation process.
 - This process is deceptively difficult to do well.

History of Languages in Computing

- ASCII was the defined standard of translating bits into readable text on screen
- Many characters from other foreign languages were excluded from the ASCII protocol, severely limiting the readability of the software in certain areas
- Later down the road, Unicode was used to include the text within these languages, opening the door for better compatibility towards language support.

Determining the Supported Languages

- Determining what languages we localise software for is a vital component of the localisation process.
- What languages we support determines who can use our software, making language settings an important accessibility feature.

Determining the Supported Languages, cont.

- Things one needs to consider when determining support:
 - Who's actually going to be using the software?
 - How many people who use the software will actually be using it in that language?
 - How difficult is it to do the translation?
- TL:DR - Cost-benefit analysis + Legal Stuff

Complications of Software Translation

- Direct word-for-word translation, largely, does not work, because language isn't a set list of concepts that everybody has different words for.
- Additionally, many languages either have completely different grammatical structures, or rely on different fundamental concepts about the nature of existence. (SVO vs SOV)
- Basically, linguistics makes localisation much more complicated than many think.

Linguistics Infodump Time!

- There are a bunch of words that exist in one language and don't in another - schadenfreude, saudade, синий and голубой, etc.
- Conjugation rules vary from language to language.
- Differing grammatical structures in different languages.

Robots Don't Understand Culture

- The *other* issue with direct translation is that there are some things that just don't translate.
- Idioms, pop culture references, and turns of phrases frequently don't cross the language barrier.
- This can lead to localisations that feel awkward, stilted, and like an afterthought to users in those languages.

Graphical Considerations

- Translation can lead to graphical issues, as characters from varying languages may have varied sizes in their characters
- This will lead to the graphical aspects of the application to fail to adjust well to changes in language structure
- Considerations regarding how user interfaces need to be adjusted to fit within the barriers of multiple languages must be considered to avoid disaster.

Graphical Considerations, cont.

- Examples of localisation that can mess with the UI:
 - Words that are different lengths (e.g. 'settings' vs. 'einstellungen')
 - Different writing systems (not all fonts account for all sets of characters; left- vs. right-starting writing systems)
 - Colors and Symbols may have different meanings in different places

Conclusion

- Software that fails to consider the languages of all users will fail to be understood by those users.
- Machine Translation tends to disregard cultural context, leading to awkward translations that throw off users.
- Choice of language support should not depend solely on profitability, but on who will most benefit from your software.

Thank you!

Contact

evanderl@mtu.edu

perobert@mtu.edu

References

- Bass, M. (2004). Global Software Development Process Research at Siemens. "Third International Workshop on Global Software Development (GSD 2004)" W12S Workshop - 26th International Conference on Software Engineering, 2004, 8–11. <https://doi.org/10.1049/ic:20040304>
- Beckett, G. H., & McGivern, L. (1999). *Dilemmas in designing multimedia software for learners of English as a second or foreign language*. International Society for Technology in Education.
- Cabezas, C., Dort, B., Resnik, P., & Maryland univ college park inst for advanced computer studies. (2001). *Spanish Language Processing at University of Maryland: Building Infrastructure for Multilingual Applications*. Maryland univ college park inst for advanced computer studies.
- Hau, E., & Aparicio, M. (2008). Software internationalization and localization in web based ERP. *Proceedings of the 26th Annual ACM International Conference on Design of Communication*, 175–180. <https://doi.org/10.1145/1456536.1456570>
- Keniston, K. (1997). *Software localization: Notes on technology and culture*. Program in Science, Technology, and Society, Massachusetts Institute of Technology.
- Matthew. (2024, November 25). *Creating multi-language support in full-stack development | by Matthew | Medium*. Creating Multi-Language Support in Full-Stack Development. <https://medium.com/@mdburkee/creating-multi-language-support-in-full-stack-development-3066d6abf833>
- Maumevičienė, D. (2017). Lokalizacija Kaip Komunikacijos Aktas. *Vertimo Studijos*, 4(4), 95. <https://doi.org/10.15388/vertstud.2011.4.10576>
- Mick, G., & Pym, A. (2015). *The role of revision in English-spanish software localization*. Universitat Rovira i Virgili.
- Ressin, M. (2015). *An empirical examination of interdisciplinary collaboration within the practice of localisation and development of international software*. University of West London.
- Rivas-Ginel, M. I., Gautier, L., Corpas Pastor, G., Froeliger, N., Seghiri, M., Monti, J., & Faria Pires, L. de. (2023). *The ergonomics of Cat Tools for video game localisation*.
- Taylor, D. (1992). *Global software: Developing applications for the International Market*. Springer-Verlag.
- YAZICI, Y., & HAYTA, P. (2021). Investigation of the relationship between computer programs and foreign language used in graphic design process. *Yildiz Journal of Art and Design*, 8(1), 43–52. <https://doi.org/10.47481/yjad.811071>