How does the engineering workflow of a video game differ from that of other software when undertaking the translation process for Localisation?

COMP160 - Software Engineering Essay

1606119

March 30, 2017

Localisation of software has the capacity to pose many a challenge to developers, as there are so many factors and small details that come into play, with a large portions of the software needing to be completely overhauled depending on the locale, a set of "Internationalisation Standards"[1, pg.5] have to be followed by the developer if they want their software to be understandable and usable in the local they are adapting it for, this is applicable to both video games and other types of software and so my essay will compare and contrast the different steps that the developers of these types of software have to take.

1 Introduction

As technology's accessibility and uptake across the globe is forever increasing, many new emerging markets opportunities are opening up for software developers and having localised versions of their software allows for "Local users better understand and use it, attract more users, and increase software sales"[2, pg.1]. As previously mentioned, developers have to work to a set of standards to make sure that their software is understandable to a variety of markets, and through my research I have found that, due to the differences in the engineering workflow of video games and other types of software, that different organisations and groups have created several different sets of standards for different types of software, and as games require a lot more context and speech to relay the story to the player, the workflow is a lot larger, however there are some points where they overlap and flow in a similar way.

2 Integral Differences in Internationalisation Standards

As I have found through my research, compared to other types of software, video games have both a much higher amount of content, and require much more work to localise than other types of software e.g. Business, as "The linguistic port is more than simple translation; it involves use of vernacular and dialect as well as culturally appropriate interpretation of the game context" [3, pg.1], meaning the translation and it's implementation will need to use local language and the correct use of this to be fully understood; an increasing step that developers are taking is to "Cultralize" their software[4, pg.4], which goes further into the games content and removes anything that may be seen as offensive or not well received in a particular location, this has become a standard of games internationalisation, as without the correct

content being removed or replaced, it will not pass certification in that region, which extends the development time and wastes money and resources, while business software won't have any of these issues it doesn't need to be certification. Which when focusing on the modularity (Engineering Principle) of their software, game developers will need to focus on making sure that "Resource files should store all text used in the game" and that they "do not hard code text strings in the source code."[4, pg.24] as having things such as the subtitle text and the dialogue files hard coded makes it very hard to change them for localisation or certification work, which could cause issues with the code as it's deviating from it's original form; as this is noted good practice, many game developers utilise this, such as Bioware, who's alien languages are simply a couple of sound bytes repeated for every locale [3, pg.1], the only need being to translate the subtitles, which significantly cuts the workload, at this point the similarities of the workflow for both types of software really shows, but in the next section, the paper will cover the smaller the defining features of both, in particular when the translation process is undertaken.

3 The Translation Process

Due to the differences between the two types software discussed in the last section, and through the sources that were found, translation of the software's text and the changes that need to be made to said text to meet the Internationalisation standards/best practices that I covered seems to be the biggest part of the process, along with the "Cultralization" of said text, so I will now cover that in greater detail. As discussed in the previous section, as games become much larger and rely more heavily on fictional story elements to drive the narrative, these features can't simply be translated. Where as there are

sources to suggest that the use of translation software to localise software that features “highly repetitive texts” [5, pg.1] a trait that business software seems to have, allows the software’s text to be quickly changed, meaning it can be sold in other markets quicker and take hold of valuable market shares[5, pg.1], and significantly cutting the work load,and although this type of tool would be extremely useful in games development, the aforementioned additional amount of content normally found in games over business software, so this process is normally undertaken by a team of specialist employees, who would be more expensive to employ and the process would most definitely take longer, the final product would most likely be more accurate and be much more “cultralized” as it has been found that the aforementioned commercial translation software , or at least the program used in this particular paper require a “100 percent match” to successfully translate the word, which is another reason why the tool wouldn’t be particularly beneficial to game development, but it does crossover with business software as it may also feature program or locale specific terminology such as currency names[5, pg.2], so this automatic process would still need to be checked by a human but it would still cut the work load significantly, but recently there have been developments to try and make translation easier across the board.

One of these developments was the UNICODE standard [6], “which specifies the representation of text in modern software products and standards” [7, pg.6], and subvert problems such as The Korean Character Code[8], which was very controversial at the time , by collecting all of the used characters in all of the different languages and making how they are displayed and encoded the same, it makes for less work for developers as there is no need to rewrite display code for their software, and fixes the problem of locales that read right to left, as it allows for bi-directional display of said text[9],

which is very helpful to developers who might have online portions as having bi-directional text is a "Internationalisation Standard" [10] that needs to be followed by developers of softare such as business, but this also carries over to game development with features such as text boxes and subtitles.

4 Conclusion

Through developments in the standards that developers must follow and ones that affect the computing field overall, the workflows of game development and other software has been brought closer and closer together, although game development still has a much longer and more involved localisation process due to the sheer amount of content over other types of software, and no tool will ever bring it down to the a level where a human will no longer be needed.

References

[1] J. M. Hogan, C. Ho-Stuart, and B. Pham, "Key challenges in software internationalisation," in *Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation-Volume 32*. Australian Computer Society, Inc., 2004, pp. 187–194.

[2] X. Xia, D. Lo, F. Zhu, X. Wang, and B. Zhou, "Software internationalization and localization: An industrial experience," in *2013 18th International Conference on Engineering of Complex Computer Systems*, July 2013, pp. 222–231.

[3] A. Losavio, S. Polyakova, T. Hayden, and M. Losavio, "Linguistic implementations in computer game and virtual world design," in

Computer Games: AI, Animation, Mobile, Multimedia, Educational and Serious Games (CGAMES), 2014. IEEE, 2014, pp. 1–4.

[4] Honeywood, R and Fung, J, "Best practices for game localization," [Online]. Available: http://c.ymcdn.com/sites/www.igda.org/resource/\collection/2DA60D94-0F74-46B1-A9E2-F2CE8B72EA4D/Best-Practices-for-Game-Localization-v22.pdf, [Accessed: 19-March-2017].

[5] R. Schaler, "A practical evaluation of an integrated translation tool during a large scale localisation project," in *Proceedings of the fourth conference on Applied natural language processing*. Association for Computational Linguistics, 1994, pp. 192–193.

[6] "The unicode standard version 8.0 core specification," [Online]. Available: http://www.unicode.org/versions/Unicode8.0.0/UnicodeStandard-8.0.pdf, [Accessed: 29-March-2017].

[7] R. Schaler, "Language resources and localisation," in *Proceedings of the Second International Workshop on Language Resources for Translation Work, Research and Training*. Association for Computational Linguistics, 2004, pp. 27–34.

[8] D. Park, "The korean character code: A national controversy, 1987-1995," *IEEE Annals of the History of Computing*, vol. 38, no. 2, pp. 40–53, 2016.

[9] "Unicode bidirectional algorithm," [Online]. Available: http://unicode.org/reports/tr9/.

[10] "Internationalization standards," [Online]. Available: https://www.w3.org/standards/techs/i18n#w3c_all, [Accessed: 19-March-2017].