Turning MediaWiki into an efficient localisation platform

Niklas Laxström

Table of Contents

1 Introduction

For most people Wikipedia is the Wikipedia in their language. There are over 250 Wikipedias and together they are one of the biggest and most popular websites of the world. MediaWiki, the software that supports Wikipedia, needs to be versatile in order to support the requirements of languages like Khmer, Thai, Russian and Chinese.

In order to support all these languages, the Wikimedia Foundation relies on the knowledge and the effort of its communities. The resulting internationalisation and localisation is very much based on the experience of a core group of developers that concentrate their effort on the translatewiki.net project.

This group of developers has grown in experience over the last few years and it has resulted in an increasingly sophisticated environment where a growing group of volunteers work on over 300 languages. This ever-increasing number of languages brings new challenges and as these challenges are met, our perception of what we want translatewiki.net to be changes.

With this Summer of Code project we aim to realise most of the dreams for a better localisation environment. In our community the most valuable resource is the time of our localisers. As the effort needed to maintain the localisation for a language grows, increased efficiency will help us keep Wikipedia one of the most relevant resources on the Internet

2 Word about myself

My name is Niklas Laxström, I study Language Technology at the University of Helsinki. I have completed minors in Finnish language and Computer Science. As I wanted to apply what I learn in my studies, I started the translatewiki.net project in 2005. The project became really successful and it now has its own community of MediaWiki developers and localisers.

While translatewiki.net is a collaborative project, I have written most of the Translate extension. Nowadays I'm having difficulties balancing the time the project deserves and my studies, work to earn some money and the time to do sports, read books and socialise with my friends.

3 Project plan

In this Summer of Code project, I will focus on the most relevant bugs and the most promising new features. Efficiency of the work flow both for the localisers and the administrators will have priority. Translatewiki.net has its source code in the Wikimedia Foundation's SVN repository, and will be licensed with a GPL V2 or later license.

3.1 Overview

I have divided the task into groups according to the effect they will have for better overview. Each feature also has an assigned priority and estimated size in hours. Legend: **F** = feature, **B** = bug, **C** = check. The total amount of work is estimated 302 hours, which is two months of work 37.6 hours a week.

3.1.1 Translation performance and tools (154h)

- P1 **B** (4h): Sometimes the edit area is filled with some translation, sometimes it is not
- P2 **F** (20h): Basic language independent translation memory

- P2 **F** (4h): User defined assistance languages
- P2 **F** (40h): Editing of messages using AJAX
- P2 **F** (18h) Fix plural handling in Gettext support
- P3 **F** (2h): Add translation hint to fuzzy messages
- P3 **F** (12h): Enable Gettext po file submission through web UI
- P3 **F** (2h): Add links to extension pages in mediawiki.org at Special:Translate.
- P4 **F** (8h): Quotation consistency checking on language level
- P4 **F** (24h): User defined fallback languages in MediaWiki
- P4 **F** (20h): Edit message in multiple languages

3.1.2 Administration performance and tools (74h)

- P1 **F** (10h): Export core special page aliases from command line
- P1 **F** (10h): Check for conflicting special page alias translations inside and between languages
- P1 **B** (1h): Special:Magic export fails when local underlying page is not present. Gives HTML output in form.
- P2 **F** (40h): Tool to track changes in English messages with web processing to enable fuzzying and "incompatible with lower branch" tagging
- P2 **B** (1h): Core-mostused group does not work on Windows
  - P4 **C** (4h): Audit the parsing of files in the extension for problems with newlines
- P3 F (8h): Make extension message file parser output more informative error messages than just "failed"

3.1.3 Moving i18n forward (60h)

- P2 **F** (20h): Add standard support for localisation of Magic words in MediaWiki extensions
- P2 **F** (12h): Make it possible to blacklist messages from sanity checks, as some just should not be reported
- P3 **F** (8h): Export for pages using tag `translate`, so that they can be copied to another instance of MediaWiki easily.
- P4 **F** (20h): We should pull more data out of CLDR (time zone translations, country name translations)

3.1.4 General performance (14h)

- P1 **F** (10h): Make message translation status checks fast
  - P2 **F** (2h): Fix the FIXMEs in Special:LanguageStats and make it faster
  - P2 **F** (2h): Create a special page with real-time statistics for an extension group.

3.2 Detailed description

3.2.1 Translation performance and tools

The faster a translator can translate the more he can achieve in the same time. Even though quantity can make an effect, we should also support processes that ensure that the results are of good quality. This includes translation memories and term banks as well as a review process.

P1 **B** (2h): Sometimes the edit area contains some default text and sometimes it is empty when translating new messages. This causes confusion and conflicts with Google translation gadget.

Implementation plan: Decide or research what the default text should be or if it should be empty.

Deliverables: When translating new messages, the text area is consistently empty or contains some type of default value in all message groups.

P2 **F** (20h): Basic language independent translation memory. Translation memory should help with the fully or almost identical messages.

Implementation plan: Check up Pootle's translation memory tools. If they are suitable, integrate those into translation platform. Look for alternative pre-made solutions if needed. If nothing suitable is found, drop the task.

Deliverables: Automatic or semi-automatic creation of translation memory database for messages in given groups or all messages in one language. An interface which automatically or on request suggests translations from the database for message under translation. It does not need to understand anything about syntax, morphology or punctuation.

P2 **F** (4h): User defined assistance languages. These are the languages that are shown besides the definition of a message. The knowledge of languages vary from translator to translator and so does their preferences.

Implementation plan: Provide a global user setting and a way to set list of languages in Special:Preferences. Use this setting to choose the shown languages.

Deliverables: A new setting in the preferences page. When translating messages, this setting must be honoured if translations exist.

P2 **F** (40h): Editing of messages using AJAX. The major grieve in the current system is the need to jump of from the message list to a new edit page when translating a message. There is no easy way to get back to the message. Using AJAX there is no need to move users away from the message list.

Implementation plan: Consult if needed with MediaWiki developers a way to provide way to get edit page without extra stuff. Use AJAX to get this stuff and show it in a popup or similar on top of the current page.

Deliverables: Special:Translate does not throw users away from the list, if JavaScript is supported.

Risks: This needs changes in MediaWiki core, which needs to pass tight review for code quality and correctness. JavaScript is not a very strong language of the coder.

P2 **F** (18h) Fix plural handling in Gettext support. Plural entries need to be parsed and presented to the user.

Implementation plan: Edit Gettext handler to read plural entries. Then combine all plurals into one message. Edit Gettext handler to write plural entires in correct format, expanding them into multiple entries from one message.

Deliverables: Gettext files containing plural can be read, edited and exported without problems.

P3 **F** (2h): Add translation hint to fuzzy messages. Fuzzy messages are marked with a string "!! FUZZY!!". New translators may be confused about the concept. Adding a helpful message should aid translators to act in desirable ways.

Implementation plan: Check if the edit area contains !!FUZZY!! and if it does, add a message to the existing information area.

Deliverables: A message shown for fuzzy messages, which describes the fuzzy concept and instructs how to update the message. The message should be translatable.

P3 **F** (12h): Enable Gettext po file submission through web UI. We currently allow exporting a message group into special po file for off-line translation using dedicated translation tools. It should be possible to trusted users to upload those files back into the system without a need for an administrator with shell access to take part in the process.

Implementation plan: Check if MediaWikis existing file uploading framework can be used. Provide a way for a privileged user to specify uploaded file or upload file which is then imported to the database. Because it is easy to accidentally to cause lot of damage, for example if the file is very old export, it should be restricted to certain users. Alternatively, allow anyone to upload the file, but require that someone privileged checks if the file is OK and then applies it trough web interface.

Deliverables: Possibility to upload Gettext po files. Possibility for check what is changed in the files and way to apply those changes. Invalid files or files not in the special format should be detected.

P3 **F** (2h): Add links to extension pages in mediawiki.org at Special:Translate. As the *-desc messages are used for message descriptions, there is no place to add links to mediawiki.org. In the message listing, the header contains message name and it could be made a link.

Implementation plan: Make a way for message groups to provide link with more information and display it somewhere in the interface.

Deliverables: Link to the mediawiki.org pages if any for MediaWiki extensions.

P4 **F** (8h): Quotation consistency checking on language level. Consistent quotation helps in consistency and gives a better touch. The quotation can vary between languages and even inside languages.

Implementation plan: Check if there is database of quoting styles used by languages, for example in CLDR. Estimate the number of quotes in messages that should not be localised. If the amount is low enough for this feature to be useful as-is, or with the help of the ability of blacklisting individual checks per messages, do it. Otherwise document the problems and why it is not feasible to do at this point of time.

Deliverables: A manual or extracted database of quoting styles, and warnings when other style of quotes are used in a translation.

P4 **F** (24h): User defined fallback languages in MediaWiki. Even though there is a tendency that Finnish speakers can understand Swedish – because it is a national language of Finland – there is variation between the skills and preferences of individuals. For example, KDE allows user to set their preferred fallback chain for missing translations.

Implementation plan: Make a user preference for fallback languages. Make the message getting functions to prefer this chain over the predefined.

Deliverables: A user setting and documentation.

Risk: This needs changes in MediaWiki proper and its message getting functions, which are very critical. The code meeds to pass tight review for code quality and correctness.

P4 **F** (20h): Allow editing one message in multiple languages. This is a feature for people working in multiple language variants which differ mostly in script like latin or cyrillic.

Implementation plan: Provide a primary or an alternative interface to Special:Translate for editing one message in multiple languages simultaneously.

Deliverables: A way to edit one message in multiple languages simultaneously.

3.2.2 Administration performance and tools

Administrators run the whole platform. Without administrators translators work is nothing. Translations need to be exported, source definitions updated, external translations imported and many other tasks. It is of very high importance to minimise the time needed for repetitive tasks.

P1 **F** (10h): Currently exporting MediaWiki's special page aliases is slow manual work, which is away from other things an administrator has to do. Exporting these should be automated like other mass exports.

Implementation plan: Adapt the existing export classes to be able to replace any variable definition in the message files. Write a command line script similar which updates alias definitions.

Deliverables: A command line script similar to the one which exports extensions' aliases.

P1 **F** (10h): Check for conflicting special page alias translations inside and between languages. If two different pages have a same alias, one of those will not work. It is important to make sure no such errors are committed into version control system. There is already some in-language checks for duplicate translations, but they are only shown as comments on export. Asking translators to fix them is slow and delays updates. It should not be possible to make such errors in the first place.

Implementation plan: Edit the special page to show warnings when such condition is detected or even disallow saving changes until it is fixed.

Deliverables: Translatable warnings and optionally such changes cannot be saved any more.

P1 **B** (1h): Special:Magic export fails when local underlying page is not present. It gives HTML output in form, which breaks the page source very badly.

Implementation plan: Fix the use of Xml-functions.

Deliverables: The error condition does not appear any more

P2 **F** (40h): Imeplement a tool to track changes in English messages with web processing to enable fuzzying and “incompatible with lower branch” tagging.

P2 **B** (1h): Core-mostused group does not work on Windows. This is probably due handling of newlines, which are different on Windows.

Implementation plan: Check if this is the case and fix it.

Deliverables: Core-mostused group works on Windows platform too.

P4 **C** (4h): Audit the parsing of files in the extension for problems with newlines. There are multiple places where files are handled.

Implementation plan: Check the occurrences of file handling functions like `fopen` and `file_get_contents`, their modes and any regular expressions or other functions that handle the data.

Deliverables: Every instance is checked for correct handling. There is no known bugs due to parsing of newlines.

P3 **F** (8h): Make extension message file parser output more informative error messages than just “failed”. Currently it is very hard to figure out why the parsing might have failed.

Implementation plan: Edit the code to add sanity checks which can report with higher precision what is wrong with the file.

Deliverables: Improved error messages.

3.2.3 Moving i18n forward

MediaWiki's i18n support is high quality. However as time passes, new requirements are put for the software. This is because new languages are added and people's expectation grow more demanding. We must continue improving our i18n support.

P2 **F** (20h): Add a standard support for localisation of Magic words in MediaWiki extensions. It is already supported for core. It should be noted that localisation of Magic words is not uncontroversial, but some languages still want to do it, especially right-to-left languages.

Implementation plan: Mirror the implementation of localised extension aliases for extensions. It uses a global for registering translation files. Then add support to this extension for translating them.

Deliverables: A standard way for extensions to register Magic words for translation. Support for translating them in this extension.

P2 **F** (12h): Make it possible to blacklist messages from sanity checks, as some of them are always false positives. Messages tagged as such will stay on the list of problematic messages and hinder translators ability to find messages which need to be fixed. There two main reasons for blacklisting checks: 1) a given language doesn't use particular feature like support for multiple plurals. 2) The message is in some way special and should exempt some particular check in all languages. Blacklisting both should be supported.

Implementation plan: Decide where and in which format blacklist entries are stored. Text file, php code or in a wiki page. Type 2 exempts would naturally go with the message groups they belong to, but on the other hand it may be reasonable to have all exemptions at one place. Design a syntax and

parser for it. Make the checking framework aware of it.

Deliverables: A place and syntax to disable some checks for reasons 1 and 2. It should be intuitive enough for non-programmes to be able to use it without too many difficulties.

P3 **F** (8h): A way to export for pages using tag `translate`, so that they can be copied to another instance of MediaWiki easily. This means that translated pages can be exported in standard wikicode which does not depend on Translate extensions. **What is the use case of this?**

Implementation plan: Provide source for such pages and strip incompatible mark-up if needed.

Deliverables: Accessible page source code in wikitext.

P4 **F** (20h): We should pull more data out of CLDR, like for example time zone translations and country name translations. Having this data in matrix of hundreds of languages is enormous amount of work, and it makes no sense to have translators to do it again for every project that exists.

Implementation plan: The CLDR data is provided at least in XML format, and the relevant data just needs to be extracted from those. Things like poor coverage and errors of the data and conflicting use of language codes may need ways to map language codes and local overrides.

Deliverables: A facility to query data like time zone names in addition to the current ability to query language names. It is reasonable to expect that MediaWiki itself wont use this data immediately due its shortcomings.

3.2.4 General performance

Poor performance limits what we can do with things. Sometimes performance can be achieved by caching, but sometimes there is other ways like better algorithms or better data formats.

P1 **F** (10h): Make message translation status checks fast. There must be way to query messages' translation status without loading the revision text for it. Currently this is done to check the existence of fuzzy tag.

Implementation plan: Research and decide how store message status efficiently and if it needs to be cached separately. Possible alternatives include the recently added functionality of tagging revisions and making fuzzy tag a template which puts the message in a special category.

Deliverables: Checking translation statistics for hundreds of groups should be fast enough for single web request.

P2 **F** (2h): Fix the FIXMEs in Special:LanguageStats and make it fast.

Implementation plan: Once the status is fast to query, the special page should be fast with small changes only.

Deliverables: The statistics loads in less than few seconds compared to the about 10 seconds currently.

P2 **F** (2h): Create a special page with real-time statistics for an extension group. This allows translators to find groups that need attention when the translation status is nearing completion.

Implementation plan: We already have similar special pages to look for examples.

Deliverables: Translator should see which groups need work without scanning them trough one by one.

3.3 Schedule

The purpose of this schedule is to assign tasks to each week. It is used to track that project stays on schedule.

3.3.1 Week1 (36h)

- P1 **B** (4h): Sometimes the edit area is filled with some translation, sometimes it is not
- P1 **F** (10h): Export core special page aliases from command line
- P1 **F** (10h): Make message translation status checks fast
  - P2 **F** (2h): Fix the FIXMEs in Special:LanguageStats and make it faster
  - P2 **F** (2h): Create a special page with real-time statistics for an extension group.
- P3 F (8h): Make extension message file parser output more informative error messages than just “failed”

3.3.2 Week 2 (38h)

- P2 **F** (20h): Add standard support for localisation of Magic words in MediaWiki extensions
- P2 **F** (12h): Make it possible to blacklist messages from sanity checks, as some just should not be reported
- P1 **B** (1h): Special:Magic export fails when local underlying page is not present. Gives HTML output in form.
- P2 **B** (1h): Core-mostused group does not work on Windows
  - P4 **C** (4h): Audit the parsing of files in the extension for problems with newlines

3.3.3 Week 3 (40h)

- P2 **F** (40h): Editing of messages using AJAX

3.3.4 Week 4 (38h)

- P2 **F** (20h): Basic language independent translation memory
- P2 **F** (18h) Fix plural handling in Gettext support

3.3.5 Week 5 (34h)

- P3 **F** (12h): Enable Gettext po file submission through web UI
- P2 **F** (4h): User defined assistance languages
- P3 **F** (8h): Export for pages using tag `translate`, so that they can be copied to another instance of MediaWiki easily.
- P1 **F** (10h): Check for conflicting special page alias translations inside and between languages

3.3.6 Week 6 (36h)

- P3 **F** (2h): Add translation hint to fuzzy messages

- P3 **F** (2h): Add links to extension pages in mediawiki.org at Special:Translate.
- P4 **F** (24h): User defined fallback languages in MediaWiki
- P4 **F** (8h): Quotation consistency checking on language level

3.3.7 Week 7 (40h)

- P2 **F** (40h): Tool to track changes in English messages with web processing to enable fuzzying and “incompatible with lower branch” tagging

3.3.8 Week 8 (40h)

- P4 **F** (20h): We should pull more data out of CLDR (time zone translations, country name translations)
- P4 **F** (20h): Edit message in multiple languages

3.4 Risks

3.4.1 Sickness or accident

Effect: high. Probability: low.

In a one person project the biggest risk is the person itself. Unexpected sickness or accident could put the project on halt. It is impossible to predict these things.

3.4.2 Infrastructure availability

Effect: medium. Probability: low.

This projects depends on continuous availability of tools and internet connectivity. Code needs to be periodically committed to version control systems. There needs to be communication in IRC, mailing lists, private emails, blogs and so on. All these services are highly reliable and longer unavailability is unlikely. Prolonged unavailability of any service can slow down or halt working.

3.4.3 Code review

Effect: medium, may freeze some task. Probability: low.

Some tasks involve modifications to MediaWiki itself. These changes likely need commenting and reviewing by other MediaWiki developers. Delays in such responses delays the completion of that particular task. In most situations there is some other task that can be worked on during that time.

3.4.4 Implementation of tasks that involve MediaWiki

Effect: medium, may prevent some task being completed. Probability: medium.

Some changes may turn out to be infeasible to implement in a way that is accepted in MediaWiki in the allocated time. The same also applies to other tasks, but in lesser extent.

3.4.5 Underestimated schedule

Effect: medium. Probability: medium.

The requirements in Translate extensions are not as strict as in MediaWiki. I have done a similar project in the past, which should give some experience in estimating tasks' difficulties. If some tasks

take longer than expected, there may not be time to complete all tasks. To avoid minimise the risks a weekly program is created. It is then easy to follow how good the project is on the schedule. Progress should also be told at least weekly by blogging for example.

3.4.6 Task is impossible to implement as described

Effect: medium. Probability: medium.

It may turn out that a task cannot be implemented properly when following the implementation plan. In such case it should be discussed with the mentor how to proceed.

4 Conclusion

There is lot to do in this project. There is however always more things that can't be done in this project. As we do new things, we make space for innovation. People using the project will try out the new features, experiment with them and decide if they like it or not. Then we get suggestions for doing things differently, or what they would like to see. It is an endless continuum. But if we don't do new things, people take what is available as the complete solution. Then the development stalls.

Is is to be expected that not all features will be very popular. Some will be used only by few persons or few times a year. Some will just drop out when we start doing things differently. In the end what matters is that we will grow. MediaWiki will be localised in more languages, more thoroughly and more extensively. Many other projects will do the same, and that matters when a speaker in a developing country finds out he does not have to learn yet another lingua franca to use computers and internet.

(or just write something else instead....)