Automated text generation for localisation an online game

K.A. Arthur, B. Brandt, A. Fedane, D. Hannan.
Brandt Translations,
3 Jocelyn Mall,
Jocelyn Street,
Dundalk, Co. Louth.
info@BrandtTranslations.com

Abstract – *The problem being examined arises in the gaming industry, where the addition of new and varied content is of great importance to gamers. New content addition is a cost driver and so, finding ways to allow scalable content, while managing costs will be of interest to producers. A solution is presented in the paper to the problem of localising a large volume of sentences of text with similar structures. An automated text generation tool was written in Ruby on Rails, which implemented a set of grammatical rules for each of the project languages, and generated grammatically correct text to appear in the user interface of the online game. The costs of full product internationalisation and rules based translation are compared in a simple model. We conclude that the rules based approach to automatic text generation is less costly, and scalable for this particular case.*

Keywords: String concatenation, localisation translation, cost reduction, translation review, string parsing, grammar rules, online games.

1 Introduction

Automated telephone attendants, that provide information, such as, “You have ” + ” 2 ” + ” messages” are very common. They produce sentences by concatenating parts of sentences together to form whole and grammatically correct sentences. The English language is relatively easy to work with when it comes to using rules for simple sentence formation. In languages other than English, the rules for forming simple sentences are more complex and context dependent.

The purpose of this paper is to describe the solution we implemented to manage the translation cost of the localisation of a massively multiplayer online text based role playing game (MMORPG). The problem we faced was the translation of large sets of sentences that were structured very similarly, contained similar content and that needed to be scalable. In this context, the scalability of the translation content referred to the ability to add more content as desired.

The software on which we were working was not internationalized, and the generation of much of the content visible to the user was through string concatenation, whereby strings are added together to form sentences. While this is appealing in English, and there are very few grammatical exceptions for which to cater, when we move to other languages, the localisation of such content becomes problematic.

String concatenation is to be avoided, according to internationalisation best practice (see Brandt 2010, Akkas 2010). In general, there should be one string to localize in the resources for every instance of a string appearing in the software user interface. If we were to implement best practice for this project, we would have had to rewrite significant portions of the base code, translate and review the software and then perform functional testing. Each phase would incur significant costs.

The problem we faced was to generate a solution that minimized the costs of producing translations for a large volume of words, without incurring both large translation cost and large review cost. Our solution involved analysing the text of the source sentences per language, and identifying a general structure for each language to which each sentence adhered. The individual words were translated and a rules based Ruby parser was used to output the target sentences.

We put the target languages sentences in spreadsheets so that translators could easily review the text. This avoided having to bring up each sentence exhaustively in gameplay. Some gameplay was necessary so that the translators could see a selection of the sentences in context.

2 Problem description

In this section, we describe the problem in greater detail and give specific examples of the text to be translated. We also examine the grammatical construction of the sentences and show their structure. The example languages we considered were; English, Spanish, Russian and German.

2.1 String concatenation

String concatenation is the process of adding together strings to make a whole message. The process is used in many applications, for example, in telephone automated attendant software, we have: You have + one + new message. You have + two + new messages. You have + six + new messages. In the online game, some of the example text is as follows: Hero+defeated+a Cyclops+in+The Abandoned Caves of Mumbo
Lady+was defeated by a+Cyclops+in+The Abandoned Caves of Mumbo
You are on a quest in the+The Baneful Dungeons of Bella+searching for+The Scroll of Bella

The English examples shown above demonstrate the versatility of the English language and that adding together components can make regular grammatically correct sentences. The only irregular sentence in the telephone attendant example is the first sentence. The second and subsequent sentences match the number with the plural of the word message, simply by adding the letter “s”. There is no such simple solution in other languages.

2.2 Case

Part of the reason for the complexity of the translation from one language to another is the fact that nouns may change depending on how they are used in the sentences. Case is the grammatical name for this process. (B.J. Blake 2001), states that, “Case is a system of marking dependent nouns for the type of relationship they bear to their heads”. The author goes on to say that case “marks the relationship of the noun to a verb at the clause level or of a noun to a preposition…” For our example, this means that nouns change their form (decline) depending on how they appear in the text. Case is not the only grammatical process to cause a noun to change its form. In addition, a noun can change its form to agree with the verb to match; case, number or gender. In English, case is not a significant feature of the language, compared to Latin, Greek or other Indo-European languages. Although the purpose of this article is not to discuss grammar in-depth, enough will be described of the main concepts order to implement our semi-automated solution for MMORPG localisation.

The context of our problem is the localisation of a massively multiplayer online, text based fantasy adventure game, into German and Spanish. An example of the text that needed to be translated is as follows:

You have found+an+ancient+Kovalli+copper+braclet
You have found+an+ancient+Kovalli+gold+braclet
You have found+an+ancient+Kovalli+gold+chalice
You have found+a+new+Mithican+bone+chalice
You have found+a+new+Mithican+bone+cup

There are other types of text in the game, providing similar issues, but for the purposes of this paper, we will examine only the above in detail. For ease, we will refer to the sentence as the “loot sentence”.

Examples of case categories in language are as follows: nominative, accusative, genitive, dative, locative, and instrumental cases. The following list shows examples of how these cases are used in sentences, (BBC 2001 online);

Case	**Example**
Nominative indicates the subject	"*John* buys the newspaper."
Accusative indicates the direct object:	"John buys the *newspaper*."
Genitive indicates possession:	"*John's* newspaper is cheap."
Dative indicates the indirect object	"John writes *to the newspaper*."
Locative indicates location or topic of conversation and is governed by the prepositions 'in', 'on', 'about', and 'in the presence of:	"John talks *about the newspaper*."
Instrumental has the sense of 'with' or 'by means of':	"John swats a fly *with the newspaper*."

Case	**English**	**Russian**

Nominative Case	Ivan sees the table.	Иван видит стол.
Accusative Case	Ivan closes the book.	Иван закрывает книгу.
Genitive Case	Ivan wrote a letter to a friend of Boris with a pen.	Иван написал письмо другу Бориса ручкой.
Dative Case	Ivan gave the table to Valery.	Иван дал стол Валерию.
Prepositional Case	The book is on the table	Книга—на столе
Instrumental Case	Ivan wrote a letter to (his) friend with a pen.	Иван написал письмо другу ручкой.

Table 1 showing examples of case in English and Russian (Lexiteria 1996 online).

As mentioned above, case, number and gender cause nouns to change their form. Each of the languages we consider in this paper is listed in Table 2along with the number of cases and number of genders of nouns.

Table 2 showing the list of languages, the number of noun genders and the number of cases in the language.

Language	**Number of genders (Masc/Fem/Neut)**	**Number of cases**
Spanish	3	0
German	3	4
Russian	3	6

Just to note, Spanish has 3 genders of noun, but the neuter gender is not commonly used.

2.3 Software

When logged into the online software, the user interacts with characters in the game, picking up points, loot and other items, while visiting a number of places, or “locations”, in the fantasy world. In the next sections, we will describe the “loot” and examine the treatment of the “loot” sentence in each language in turn. We identify a general structure for the loot sentence in each language and show how the sentences are generated. We implemented similar structures for the location names and the creatures inhabiting the game, but for simplicity and to avoid repetition we will describe only the loot. Later in the paper, we describe the implementation of the “rules engine” for generating sentences.

2.4 General sentence structure

An adventurer may win or find some loot and this information is presented in the game user interface. Examples of the types of sentence include; “you found an ancient bone Accardian bracelet” or “ you found a new bronze Kovalli cup”. There are structural variations within sentences, and some sentences have common words. Examples of English sentence templates are as follows:

Example structure:
  {age} {material-jewellery} {source} {loot_name}
Example sentence:
  ancient + bone + Accardian + bracelet

Example structure:
  {size}{type} of {kind}
Example sentence:
  huge + chest + of + cash

Example structure:
  {material-quality} {source} {loot_name}
Example sentence:
  magical + Gurthian + cup

The template structure can be reordered, as the word order may vary from one language to another. For each sentence template element, we must maintain a table of its masculine/feminine/neuter, singular/plural and case forms. The "rules engine" uses the given sentence structure for each language and generates the appropriate sentence. All of the sentence elements are stored in categorised tables, which are looked up as required.

In the following sections we will examine examples of sentence construction in the different languages. We will also demonstrate via a short calculation that even with a modest number of variables, the number of sentences to translate can quickly grow.

2.5 Spanish

The Spanish language has two genders for nouns, namely masculine and feminine, and no cases. The sentence structure in English for the sentence to describe treasure that you found is, for example; " ancient bronze Accardian brooch":

{age} {material-jewellery} {source} {loot_name}

In Spanish the order of the sentence has to be rearranged, to the following;

{age} {loot_name} {material-jewellery} {source}

The {age} variable declines with the gender and number of the variable {loot name}. This means that the words signifying age, such as "ancient" or "new" will vary their form depending on the type of "loot" picked up by the adventurer. The {material-jewellery} variable does not vary. The {source} variable varies depending on the gender and the number of the {loot name}.

Table 3 shows examples of the "age" attribute of the loot.

Attribute age en_US	**M Singular**	**F Singular**	**M Plural**	**F Plural**
Ancient	antiguo	antigua	antiguos	antiguas
Antique	histórico	histórica	históricos	históricas
New	nuevo	nueva	nuevos	nuevas

Table 4 shows examples of the "material-jewellery" attribute of the loot.

Attribute material-jewellery en_US	**Attribute material-jewellery es_ES**
Bone	de hueso
Bronze	de bronce
Iron	de hierro
Silver	de plata
Tooth	de marfil

Table 5 shows examples of the "name" attribute of the loot and grammatical gender (fem/masc) and number (sing/pl).

Loot name en_US	**Loot name es_ES**	**Fem/Masc**	**Singular/Plural**
Necklace	collar	M	S
Bracelet	brazalete	M	S
Earring	pendiente	M	S
Earrings	pendientes	M	P
Brooch	broche	M	S
Bangle	pulsera	F	S

Table 6 shows examples of the loot "source" name and its variations to match gender and number.

Attribute source en_US	**M Singular**	**F Singular**	**M Plural**	**F Plural**
Accardian	Acardiano	Acardiana	Acardianos	Acardianas
Gurthian	Gurtiano	Gurtiana	Gurtianos	Gurtianas
Kovalli	Kovaliano	Kovaliana	Kovalianos	Kovalianas

Table 7 shows examples of the loot "type" name and its variations to match gender and number.

Loot_type en_US	**Loot_type es_ES**	**Gender**	**Number Sing/Plur**
Hoard	pila	F	Singular
Chest	cofre	M	Singular
Pots	calderos	M	Plural

Table 8 shows examples of the loot "size" and its variations to match gender and number.

Loot_size en_US	**M Singular**	**F Singular**	**M Plural**	**F Plural**
Colossal	colosal	colosal	colosales	colosales
Huge	enorme	enorme	enormes	enormes
Interesting	interesante	interesante	interesantes	interesantes
Impressive	impresionante	impresionante	impresionantes	impresionantes
Small	pequeño/	pequeña	pequeños	pequeñas

Table 9 shows examples of the loot "kind".

Loot_kind en_US	**Loot_kind es_ES**
Cash	monedas
Loot	botín
Treasure	tesoro
Valuables	riquezas

The game provides for a random element, which varies the type of treasure found by the adventurer. The "rules engine" takes the random element to identify the type of loot found, and associates it with an item in the loot table. Other random variables identify the source, age and material characteristics of the loot. The rules engine generates a sentence consistent with the grammatical rules of the language, specifying gender and number of the loot variable, as stored in the loot table. The age and material variables must hold content that is grammatically consistent with the loot variable content.

Examples of some of the Spanish sentences constructed from the template and data tables are as follows:

histórico brazalete de oro kovaliano
antigua pulsera de plata acardiana
nuevos pendientes de marfil gurtianos

In addition to the structure given above, there are also other "loot" sentence templates used in the generation of this type of text. This provides a rich environment in the game for the user.

The next type of loot sentence to be examined has the English template structure:

{size}{type} of {kind}

In Spanish the word order of the text remains as it is in the English. The variable {size} varies depending on the gender and the number of {type}. The variable {kind} does not change. Examples of additional loot sentence templates, with sample text are as follows in

Table 10 shows examples additional loot template sentences in English.

English template	**Sample**
{size}{type} of {kind}	Huge pots of gold
{loot_sentence} of {where_found}	Ancient silver Accardian bangle of Lars
{material-quality} {source} {loot name}	Magical Accardian cup

The corresponding Spanish templates are ordered as follows:

Table 11 shows examples additional loot template sentences in Spanish.

Spanish template	Sample
{size}{type} de {kind}	enormes calderos de oro
{loot_sentence} de {where_found}	antigua pulsera de plata acardiana de Lars
{loot_name} {material-quality} {source}	copa mágica acardiana

In addition to the loot an adventurer might come across, the software also has a number of magical creatures, locations and other features, as mentioned earlier in this paper. The sentences for these features are generated analogously to the above.

Given a data set containing X number of variables for age, Y number of variables for item, Z number of material types and W variables for provenance, the sum total of combinations is XYZW. Consider the increase in the amount of content if one of the variables, for example X, is incremented by one. The increase in volume of sentences is:

$$(X + 1)\ Y\ Z\ W - X\ Y\ Z\ W = Y\ Z\ W \qquad (1)$$

When the number of variables of type age, that is X, is increased by one, then the increase in the number of sentences is: Y Z W. For even modest data sets, the final number of sentences that can be generated grows large quickly. For gamers, variety of content and continuous additions is of premium importance. New content when viewed in this way is a cost driver for the game. Finding ways of allowing scalable content, while managing to contain costs is of vital importance.

2.6 Russian

The Russian language declines nouns for gender, number and case. There are 6 cases in Russian, namely, nominative, accusative, genitive, dative, locative, and instrumental cases, see (SEELRC 2007). In addition, there are 3 genders for nouns; masculine, feminine and neuter. As for Spanish, we begin with the loot sentence structure for English, which is as follows;

English:
    {age} {material-jewellery} {source} {loot_name}
The corresponding sentence template structure for Russian is as follows:
    {age} {material-jewellery} {source} {loot_name}
In this example sentence, the structure and element order of the sentence template does not change from English to Russian, as it did for Spanish.

The loot appearing in this sentence is in the nominative case. All attributes for the noun describing the loot have the nominative form, where the ending of the attribute depends on the noun describing the loot name. For the purposes of illustration, we will display a limited amount of Russian data in the following tables.

Table 12 shows examples of the “age” attribute of the loot in the nominative case.

Attribute age en_US	Attribute age ru_RU	Fem	Masc	Neuter	Plural
ancient	древний	древняя	древний	древнее	древние

Table 13 shows examples of the “age” attribute of the loot in the genitive case.

Attribute age en_US	Attribute age ru_RU	Fem	Masc	Neuter	Plural
ancient	древний	древней	древнего	древнего	древних

Table 14 shows examples of the “material-jewellery” attribute of the loot.

Attribute material-jewellery en_US	Attribute material-jewellery ru_RU
bone	костяной
bronze	бронзовый
iron	железный
silver	серебряный

tooth	из зуба (no adjective in Russian)

Table 15 shows examples of the “name” attribute of the loot and grammatical gender (fem/masc) and number (sing/pl).

Loot name	**Loot name ru_RU nominative case**	**gender**	**number**
necklace	ожерелье	Neuter	Singular
bracelet	браслет	Masculine	Singular
earring	серьга	Feminine	Singular
earrings	серьги	Feminine	Plural
brooch	брошь	Feminine	Singular

Table 16 shows examples of the loot “source” name and its variations to match gender and number, nominative case.

Attribute source en_US	**Attribute source ru_RU – nominative**	fem	masc	neut	plural
Accardian	Аккардианский	Аккардианская	Аккардианский	Аккардианское	Аккардианские

Table 17 shows examples of the loot “source” name and its variations to match gender and number, genitive case.

Attribute source en_US	**Attribute source ru_RU – gentive**	fem	masc	neut	plural
Accardian	Аккардианский	Аккардианской	Аккардианского	Аккардианского	Аккардианских

In a fashion similar to Spanish, the rules engine for Russian generates loot, and other, sentences from data tables of translated words, ensuring that the sentences are grammatically correct. Grammatical correctness is ensured by matching the case, gender and number of constituent words in the loot sentences.

2.7 German

In general, the process for generation of the loot sentences in German follows the same structure as the Russian and Spanish languages. We will not repeat the detailed descriptions of text generation given for Spanish with German data, but due to German grammatical rules, we had to make additional changes to the template structure and rules engine.

Modern German has four declensions (nominative, genitive, dative, accusative) and three genders of nouns, masculine, feminine and neuter. Rather than discuss the exposition of the generation of loot sentences for German, we feel that at this stage, it is more valuable to discuss the exceptions arising in this language.

2.7.1 German Compound nouns

Given the English loot sentence template:
{age} {material-jewellery} {source} {loot_name}

The loot sentence template structure for German is as follows:

{age} {source} {material-jewellery} {loot_name}

Examples of loot sentences generated from this simple template are as follows:

Singular: neue akkardische Bronze-Halskette
Singular: altes kovallisches Gold-Armband
Plural: antike Frosthamer Silber-Ohrringe

Note, that the sentences chosen in this model are relatively simple, and do not contain grammatical structures containing pronouns, interrogative or negative clauses, as these were not required for the online content. In German, the template element order has to be changed to the following;

{adjective} + {source-material} + {noun}
new Frostham brass pen.

This means the material and the object will have to be grouped together for all such sentences. In German, these words form a compound noun. The formation of a compound noun has to follow certain rules. In most cases, the words for source and material become one word, such as “brasspen”. However in the exception cases, the source and material words require an additional “s” to be placed where the two words join, which for our data set was not required. If this is required, the rules engine can be adapted for these special cases.

The loot sentences in German have to be generated in different cases depending on the context. In English we have the following sentence:

You have stumbled upon a battle between two groups of adventurers over {size}{type}of {kind}
You have stumbled upon a battle between two groups of adventurers over a Huge Hoard of Cash

The German translation requires an accusative object, so that the variables {loot_type} and {loot_kind} match. Examples of the variables; {loot_type} and {loot_kind} are as follows:

{loot_type}:
Cache, Chest, Hoard, Mound, Mountain, Pile, Stash, Stockpile,
{loot_kind}:
Cash, Loot, Treasure, Valuables, Zorkmids

When words such as, “Kiste” (case) or “Truhe” (chest) are paired with “Geld” (cash), “Beute” (loot) or “Schatz” (treasure), a compound noun is formed: {loot_kind}{loot_type}. For example: Geldkiste, Beutekiste, or Schatztruhe.

When words such as, “Kiste” (case) or “Truhe” (chest) are paired with, “Wertgegenstände” (valuables) or “Zorkmids”, the following rule applies:
{loot_type} mit {loot_kind}
where {loot_kind} has to be in the dative case. This rule generates loot sentences such as;
Kiste mit Wertgegenständen
Truhe mit Zorkmids

If a noun denoting an unspecified amount is paired with “Geld” (cash), “Beute” (loot) or “Zorkmids”, the following rule applies:
{loot_type} {loot_kind}
This rule generates loot sentences such as;
Menge Geld
Haufen Beute
Berg Zorkminds

If a noun denoting an unspecified amount is paired with “Wertgegenstände” (valuables) or “Schatz” (treasure), the following rule applies:
{loot_type} von {loot_kind}
where {loot_kind} has to be in the dative case. This rule generates loot sentences such as;
Masse von Wertgegegenständen
Berg von Wertgegegenständen
Unmenge von Schatzstücken
Haufen von Schatzstücken

German was not the only language requiring special rules to process certain types of sentence, but it required most of the special cases. The difficulties encountered in automated sentence generation shows explicitly that natural languages do not easily map to one-another in a mechanistic fashion, and that for every “general” grammatical rule there exist exceptions.

2.8 Rules engine

We have discussed the sentence template structure for the languages (Spanish, Russian and German) relevant to our project, and the process for generating grammatically correct sentences from components. For the limited “loot sentence” examples, we have also specified which sentence element drives the agreement with the other sentence elements for each sentence template type. We will now look at describing the rules engine algorithm showing example from the Russian language.

The content to be localised for the online game was analyzed by linguists, who identified the grammatical rules for each sentence type in the content. From our localisation experience, we know that languages do not map to another in a unique and one-to-one fashion, and for every attempt to define a mapping process there exists exceptions.

The sequence of events one would use to generate the loot text for the template: {age} {material-jewellery} {source} {loot_name}, is as follows:

1. From the game software random element, generate the loot name. This provides the gender and number to which other sentence elements have to agree. Example: ожерелье (necklace)
2. Generate adjective of type: age. Example: древний (ancient)
3. Select appropriate nominative adjective form from adjectives list. Example: древнее
4. Generate adjective of type: source. Example: Аккардианский (Accardian)
5. Select appropriate nominative adjective form from adjectives list. Example: Аккардианское
6. Generate adjective of type: material-jewellery. Example: серебряный (silver)
7. Select appropriate nominative adjective form from adjectives list – серебряное
8. Place the words generated in the sentence template.

Example output might be the following:

{age} {material-jewellery} {source} {loot_name} – Ancient silver Accardian necklace
{age} {material-jewellery} {source} {loot_name} – древнее серебряное Аккардианское ожерелье

The Rules Engine is a piece of software that uses the grammatical rules for the target language and puts together an appropriate string based on the grammatical rules for that string. It follows the above steps. The engine has an exception list of special cases, which are different for each language and sentence template type. Naturally, if the list of special cases was a similar number to the total number of sentences, then we would not consider the rules engine to be useful. The Rules Engine generates value because the number of special cases is significantly lower than the total number of cases that can be implemented with the generic grammatical rules.

2.9 Costs

To evaluate the costs of the project, we have to consider what alternative methods are available. In our analysis, there are two methods;

- Internationalisation – where every sentence containing a variation is stored as a single string, following the rule that there be no string concatenation.
- Rules engine approach – where single words are translated and concatenated into full strings.

For simplicity, and to facilitate comparison between the two models, we assume;

- The software internationalisation costs are going to be similar to the costs for the development of the Rules engine.
- The review costs of the translations generated by both the rules method and the full internationalisation method are similar.

This means that in this simple model, all we have to do is compare word counts from the two methods to get an overal indication of which is the lower cost. In the full internationalisation case, every variation in a variable generates a sentence.

$$\text{Number of sentences} = X\ Y\ Z\ W \text{ sentences} \qquad (2)$$

Let there be N words per sentence on average, so that the total number of words is:

$$\text{Number of words} = X\ Y\ Z\ W \,.\, N \text{ words} \qquad (3)$$

For simplicity we have discounted the effect of repetition in translation memory models. The effect of translation memory is to reduce the amount of new words. In this case then the average number of words per sentence would be reduced. This effect does not remove the multiplicative nature of the word count.

In the case of the rules engine, let the number of genders be g. In sentence templates where there are singular and plural instances, such as Table 6, a factor of 2 is picked up, since the number of words to be translated is doubled. The number of words for each variable in the sentence is;

$$\#\ \text{age} = 2\ g\ X \text{ words.} \qquad (4)$$

$$\#\ \text{material} = Z \text{ words.} \qquad (5)$$

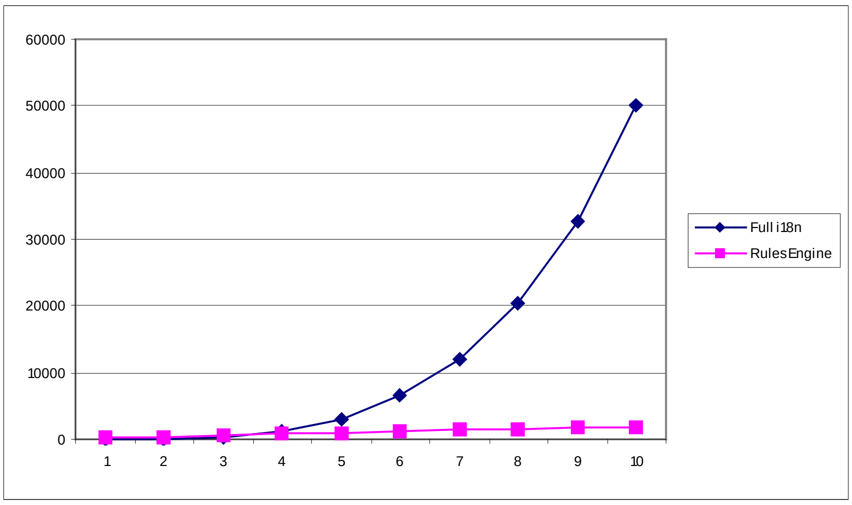
$$\#\ \text{loot name} = 2\ Y \text{ words.} \qquad (6)$$

$$\#\ \text{source} = 2\ g\ W \text{ words.} \qquad (7)$$

$$\text{Total} = 2\ g\ X + Z + 2\ Y + 2\ g\ W \text{ words.} \qquad (8)$$

To allow for multiple words in the variables, we will apply a factor of N, which is the average word count per sentence, to the total word count. The word count also picks up an additional factor when we add in the number of cases, but the total word count remains linear in the variables, X, Y, Z and W for the Rules Engine model.

Figure 1 showing the comparison of word counts for the different models.



As can be seen from Figure 1 the internationalisation model word count increases significantly faster than the Rules Engine approach which is linear in the variables. The internationalisation model, as described above, follows the Microsoft model (Microsoft 2010 online) where “Code doesn't concatenate strings to form sentences.” The implication of this is that every string that is required for presentation in the user interface is stored as a whole. The calculations above compare the word count of increasing numbers of full sentences, versus the word count of increasing numbers of single words.

It is instructive to ask how valid are our assumptions? The first assumption compares software internationalisation costs and software development costs. In the case of software internationalisation and software development the same team of resources could be used. Since cost is the rate multiplied by the time taken for the task, this means that the real comparison is between the amount of time required to perform the task of internationalisation for the product and the task of development of the Rules Engine. Both tasks would require a similar amount of development effort. Where this assumption fails, however, is in the implications of each task. Internationalising the product requires that the game is extensively functionally tested in all aspects. The testing for the addition of the Rules Engine only requires that a small part of the product be functionally tested.

The second assumption of the model is more robust to analysis. It states that the translation review costs are similar in both the software internationalisation and Rules Engine cases. In both cases the translators will review their translations both in text format, which occurs during translation, and in-context format, which occurs during game play. In the first case, the translators review their translations in something like MS Excel. For the in-context review, the translator reviews their translations in the context of game play. Howsoever the translations are generated and presented on the screen; the translator should not be able to tell how their strings are generated (if it is done correctly).

3 Conclusions

In this paper we have described the localisation problem we faced, and described the cost constraints prescribed by the client. We presented the approach taken to building a text generation application as part of an online game. The role of grammatical rules of case and gender matching of words, and the examples of special cases were described. An overview of the Rules Engine algorithm was presented. In the final section, we analysed the costs of the two available alternative approaches to the problem. Simplifying assumptions were made to facilitate comparison of the methods. In reality, a full internationalisation of the product code would require complete functional testing of the product from end to end, which was not required for the Rules Engine approach, as explained in the section above. The solution implemented above will, most likely, not work for all languages, but it does work for a sufficiently large set of languages and sentence templates for the purposes of our project. In summary, we were able to offer a cost effective method for allowing the addition of content in a scalable fashion in an online game, while managing to keep control of the key cost driver.

References

Blake, B. J., (2001), “Case”, Cambridge University Press, http://www.amazon.com/Cambridge-Textbooks-Linguistics-Barry-Blake/dp/0521014913#reader_0521014913 [accessed 22 Jun. 10]

Brandt (2010) http://www.brandttechnologies.com/media_dev_localization.shtml [accessed 23 Jun. 10]

Akkas, E., (2010) http://www.emreakkas.com/internationalization/10-internationalization-tips-for-developers-i18n-checklist [accessed 23 Jun. 10]

The Slavic and East European Language Research Center (2007) http://www.seelrc.org/ [accessed 22 Jun. 10]

BBC (2001) http://www.bbc.co.uk/dna/h2g2/alabaster/A622513 [accessed 22 Jun. 10]

Lexiteria (1996) http://www.alphadictionary.com/rusgrammar/case.html [accessed 13 Oct. 10]

Microsoft (2010) http://msdn.microsoft.com/en-us/library/cc194756.aspx [accessed 14 Oct. 10]