WILLIE – a Web Interface for a Language Learning and Instruction Environment

Werner Winiwarter

University of Vienna, Department of Scientific Computing,
Universitätsstraße 5, A-1010 Vienna, Austria,
werner.winiwarter@univie.ac.at

Abstract. In this paper we present WILLIE, a Web-based language learning tool for Japanese, which provides the language students with a comfortable interface to lexical, syntactic, and translation knowledge. The linguistic data is derived automatically from a large parallel corpus by using a Japanese-English machine translation system, which we developed in our previous research. The system randomly chooses translation examples to present them to the students using JavaScript to dynamically open pop-up windows with additional, clearly arranged color-coded information. WILLIE has been implemented in Amzi! Prolog, using the Amzi! Logic Server CGI Interface to develop the Web application.

1 Introduction

In our research work, we use the bilingual data from the JENAAD corpus [14], which contains 150,000 Japanese-English sentence pairs from news articles. During previous research we had implemented *WETCAT* [15], a Web-based machine translation system. It follows a rule-based transfer approach (see [12, 13, 1]) to achieve high translation quality, however, all the transfer rules are learnt fully automatically from the translation examples in JENAAD by using structural matching between the parse trees for source and target language.

With *WILLIE*, a Web Interface for a Language Learning and Instruction Environment, we wanted to fully exploit the linguistic knowledge derived from JENAAD by using the generic and intuitive rule formalism developed for WETCAT to convey detailed information about the individual steps involved in the translation of a Japanese sentence. The system randomly selects Japanese sentences so that the students can inspect them both at the surface level as well as regarding syntactic and transfer knowledge. All the linguistic information is only displayed to the students on demand via dynamic pop-up windows implemented in JavaScript. This user interface design choice was motivated by the objective not to overload the students with too much redundant data. On the contrary, we want to make the learning experience more interactive and interesting, e.g. in that the students only click on an item if they do not know the answer or if they want to verify their solutions.

It is our intention to extend WILLIE with additional functionality towards a fully-fledged intelligent language tutoring system by building on the results and experiences of other successful intelligent tutoring systems for Japanese, in particular Robo-Sensei [10] (based on BANZAI [8, 9]). Especially the use of our linguistic analysis techniques to manage sophisticated exercises by processing student input and generating meaningful feedback seems a very promising direction as indicated by several studies [4–7, 11, 17–19].

The main research issue involved in our work is thus how to bridge the gap between machine translation and computer-assisted language learning to make full use of the wealth of linguistic data contained in parallel corpora, to create a lucid explanation of the translation process, and to find the best representation format to convey this information to the language student.

The rest of the paper is organized as follows. After a brief overview of the system architecture in Sect. 2, we describe the lexical analysis and parsing of Japanese sentences in Sect. 3. Next, we provide a short introduction into our rule formalism to represent the transfer knowledge, and explain the generation of the color-coded visualization of the incremental transfer steps in Sect. 4. Finally, we show in Sect. 5 how we compute the surface form of the sentence translation as well as context-specific word translations.

2 System Architecture

WILLIE was implemented in Amzi! Prolog, which offers an expressive declarative programming language within the Eclipse Platform, powerful unification operations for the efficient application of the transfer rules, and full Unicode support for Japanese characters. In addition, Amzi! Prolog comes with several APIs, in particular the Amzi! Logic Server CGI Interface, which we used to develop our Web interface.

WILLIE's system architecture is outlined in Fig. 1. The modules adapted from the WETCAT machine translation system are shaded, all the other modules have been newly implemented for WILLIE. The student's Web browser sends CGI calls to the Web server, which calls the CGI application to return dynamically generated JavaScript documents. The CGI application consists of a C program responsible for starting the Amzi! Logic Server and loading the Prolog CGI script. All user input and CGI variables are asserted as facts to the Prolog logicbase before calling the Prolog part of the CGI Amzi! interface. This Prolog wrapper performs the necessary CGI bookkeeping functions and calls predicates defined in the Prolog script implementing the language learning tool.

Whenever the student asks for a new translation example, the system *randomly selects* a Japanese sentence from the 150,000 sentences in the JENAAD corpus. The sentence is first analyzed by the *tagging* module, which produces the correct segmentation into a list of word tokens annotated with part-of-speech tags. Next, the token list is converted into a parse tree by the *parsing* module. The *transfer* module traverses the parse tree top-down and applies the transfer rules in the rule base to transform the Japanese parse tree into a corresponding English generation tree. All intermediate trees and applied rules are stored for the display of the incremental transfer steps to the student. The final task of the

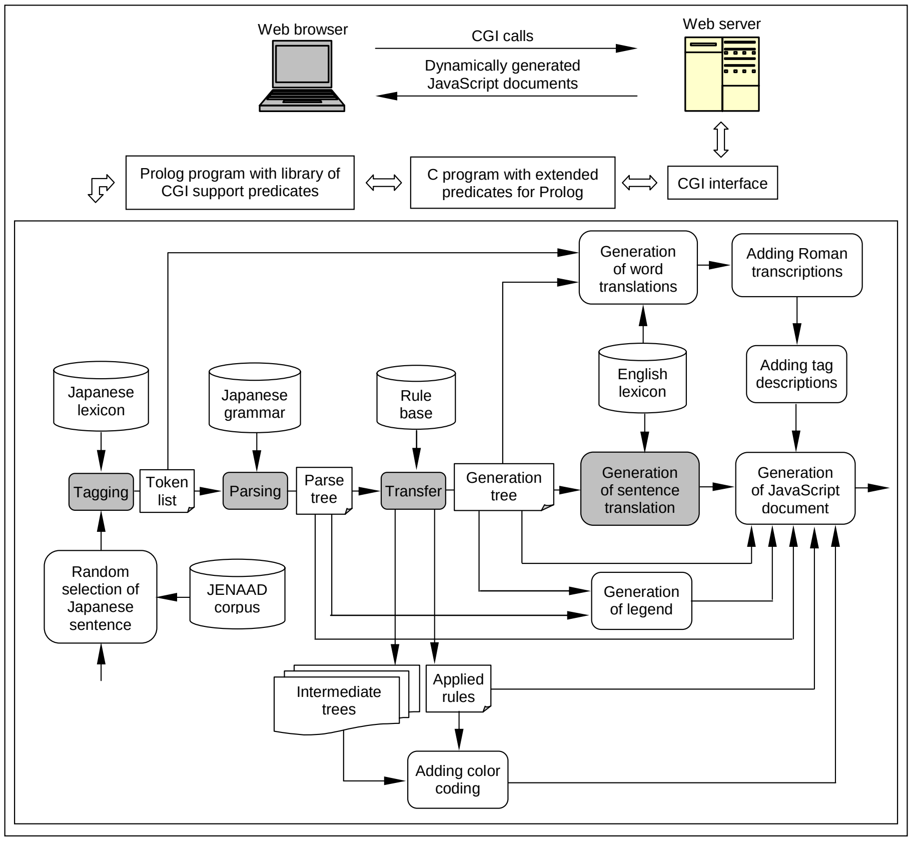
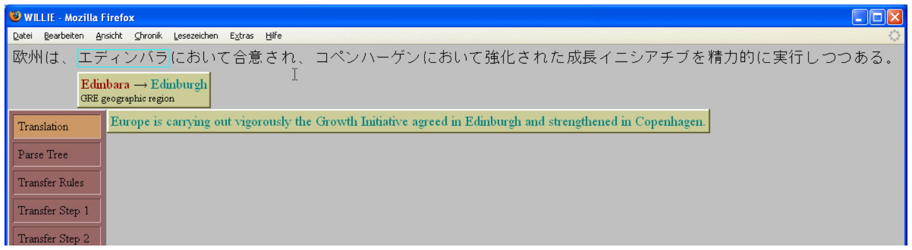

Web browser
CGI calls
Dynamically generated JavaScript documents
Web server
Prolog program with library of CGI support predicates
C program with extended predicates for Prolog
CGI interface
Generation of word translations
Adding Roman transcriptions
Japanese lexicon
Japanese grammar
Rule base
English lexicon
Adding tag descriptions
Tagging
Token list
Parsing
Parse tree
Transfer
Generation tree
Generation of sentence translation
Generation of JavaScript document
Random selection of Japanese sentence
JENAAD corpus
Generation of legend
Intermediate trees
Applied rules
Adding color coding

Fig. 1. System architecture

translation process is the *generation* of the surface form of the sentence translation by flattening the structured information in the generation tree. Although we could directly use the English translations from the JENAAD corpus, we chose not to do so in view of a planned extension of WILLIE towards the translation of free input entered as part of interactive student exercises.

We enhance the lexical data in the token list with context-specific *word translations* derived from the generation tree and add *Roman transcriptions* of Japanese words as well as plain text *tag descriptions*. To improve the comprehensibility of the tree diagrams, we dynamically generate a *legend* with descriptions of all constituent categories and feature values in both parse tree and generation tree. Finally, we apply *color coding* to the intermediate trees to convey the semantics of the applied transfer rules. All the information is gathered and combined to generate the *JavaScript document* to be sent back to the student.



欧州は、エディンバラにおいて合意され、コペンハーゲンにおいて強化された成長イニシアチブを精力的に実行しつつある。

Edinbara → Edinburgh
GRE geographic region

Europe is carrying out vigorously the Growth Initiative agreed in Edinburgh and strengthened in Copenhagen.

Fig. 2. Example of lexical data

3 Lexical Data and Parse Trees

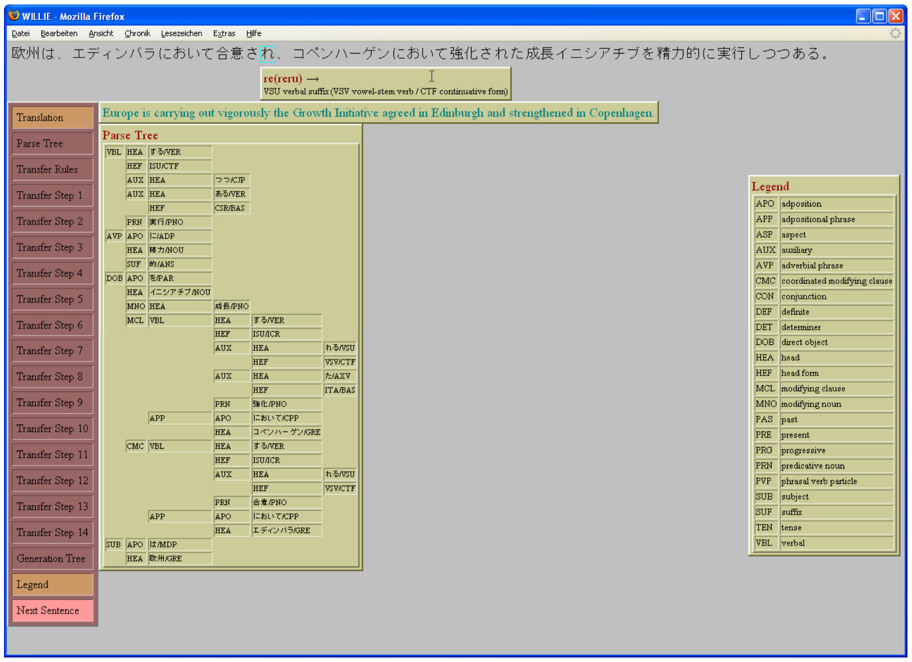
The student reaches WILLIE through a portal page, which provides instructions about how to use the system. As described before, the system randomly selects a translation example and returns a dynamically generated JavaScript page, which contains the Japanese sentence at the top and a menu bar at the left. At first, only the source sentence is visible, all lexical data is displayed in dynamic pop-up windows when moving the curser over the individual words (see Fig. 2). The English translation of the sentence can be toggled on or off by clicking on "Translation" in the menu bar.

The lexical data for each word includes the Roman transcription, the context-specific translation, and the part-of-speech tag. To compute this information, the tagging module first has to perform the correct segmentation of the Japanese sentence. For this purpose it accesses the Japanese lexicon, which was compiled automatically by applying the morphological analysis system ChaSen [3] to the JENAAD corpus. We map the numerical part-of-speech tag codes used by ChaSen to three letter acronyms and add textual descriptions. For conjugated words, we also indicate the Roman transcription of the base form and tags for conjugation type and conjugation form in parentheses (see Fig. 3). Finally, we add context-specific word translations (see Sect. 5).

By moving the cursor over "Parse Tree" in the menu bar, the student can open a pop-up window containing a nicely formatted display of the Japanese parse tree (see Fig. 3). The pop-up window remains open until the student either chooses another option from the menu bar or explicitly clicks on "Parse Tree" to close the pop-up window again. The same functionality applies to all the other entries in the menu bar except "Translation" and "Legend", which are toggled.

The parse tree is computed by the parsing module with the assistance of the Definite Clause Grammar preprocessor of Amzi! Prolog by applying the Japanese grammar to the token list. We model a sentence as a list of *constituents*, which are defined as compound terms of arity 1 with the *constituent category* as principal functor. With regard to the *argument* of a constituent we distinguish two cases:

- a *simple constituent* either represents a word with its part-of-speech tag as `atom/atom` or a feature value as `atom`,



欧州は、エディンバラにおいて合意され、コペンハーゲンにおいて強化された成長イニシアチブを精力的に実行しつつある。

re(reru) →
VSU verbal suffix (VSV vowel-stem verb / CTF continuative form)

Europe is carrying out vigorously the Growth Initiative agreed in Edinburgh and strengthened in Copenhagen.

Parse Tree

VBL	HEA	する/VER			
	HEF	ISU/CTF			
	AUX	HEA	つつ/CJP		
	AUX	HEA	ある/VER		
		HEF	CSR/BAS		
	PRN	実行/PNO			
AVP	APO	に/ADP			
	HEA	精力/NOU			
	SUF	的/ANS			
DOB	APO	を/PAR			
	HEA	イニシアチブ/NOU			
	MNO	HEA	成長/PNO		
	MCL	VBL	HEA	する/VER	
			HEF	ISU/ICR	
			AUX	HEA	れる/VSU
				HEF	VSV/CTF
			AUX	HEA	た/AXV
				HEF	ITA/BAS
			PRN	強化/PNO	
		APP	APO	において/CPP	
			HEA	コペンハーゲン/GRE	
	CMC	VBL	HEA	する/VER	
			HEF	ISU/ICR	
			AUX	HEA	れる/VSU
				HEF	VSV/CTF
			PRN	合意/PNO	
		APP	APO	において/CPP	
			HEA	エディンバラ/GRE	
SUB	APO	は/MDP			
	HEA	欧州/GRE			

Legend

APO	adposition
APP	adpositional phrase
ASP	aspect
AUX	auxiliary
AVP	adverbial phrase
CMC	coordinated modifying clause
CON	conjunction
DEF	definite
DET	determiner
DOB	direct object
HEA	head
HEF	head form
MCL	modifying clause
MNO	modifying noun
PAS	past
PRE	present
PRG	progressive
PRN	predicative noun
PVP	phrasal verb particle
SUB	subject
SUF	suffix
TEN	tense
VBL	verbal

Fig. 3. Example of parse tree

– a *complex constituent* models a phrase as a list of subconstituents.

All the acronyms used in the parse tree and the generation tree (see Sect. 5) are collected and annotated with plain text descriptions to generate an alphabetically sorted legend, which can be toggled on or off as mentioned before. For conjugated words the lexical data is divided into the two constituents HEA (head) and HEF (head form). The former has the argument base_form/part-of-speech, the latter conjugation_type/conjugation_form.

4 Transfer Rules and Steps

In our approach we have developed a very generic formalism to represent translation knowledge. We divide a translation into a sequence of translation steps, where each step is the application of one transfer rule. There only exist three different types of transfer rules: a *word transfer rule* translates the argument of a simple constituent, a *constituent transfer rule* translates both the category and the argument of a complex constituent, and a *phrase transfer rule* allows to define elaborate conditions and substitutions on the argument of a complex constituent.

All the transfer rules are actually stored as Prolog facts in the rule base. The rule base is created automatically by using structural matching between parse trees of translation examples from the JENAAD corpus. For that purpose we also have to tag and parse the English sentences from the corpus. The English lexicon used by the English tagging module has been built automatically by applying the MontyTagger [2] to the JENAAD corpus. The grammar rules used by the English parsing module are again written in Definite Clause Grammar syntax.

The acquisition module traverses the Japanese and English parse tree for a translation example and derives new transfer rules. The search for new rules starts at the sentence level by recursively mapping the individual subconstituents of the Japanese sentence. We also perform a consolidation run on the complete set of rules, which generalizes rules to avoid overtraining and to increase the coverage for new unseen data. For more details on the acquisition process and a more formal treatment of the rule formalism we refer to [16].

In the following, we list three illustrative examples of transfer rules (using Roman transcriptions for the ease of the reader):

1. WTR(shijō/NOU, market/NN).
2. CTR(MNO, MAJ, keizai/NOU, [HEA(keizai/NOU)], [HEA(economic/JJ)]).
3. PTR(CL, suru/VER, [APP([APO(ni totte/CPP), HEA(hatten/PNO) | X1])], [APP([APO(for/IN), HEA(progress/NN) | X1])]).

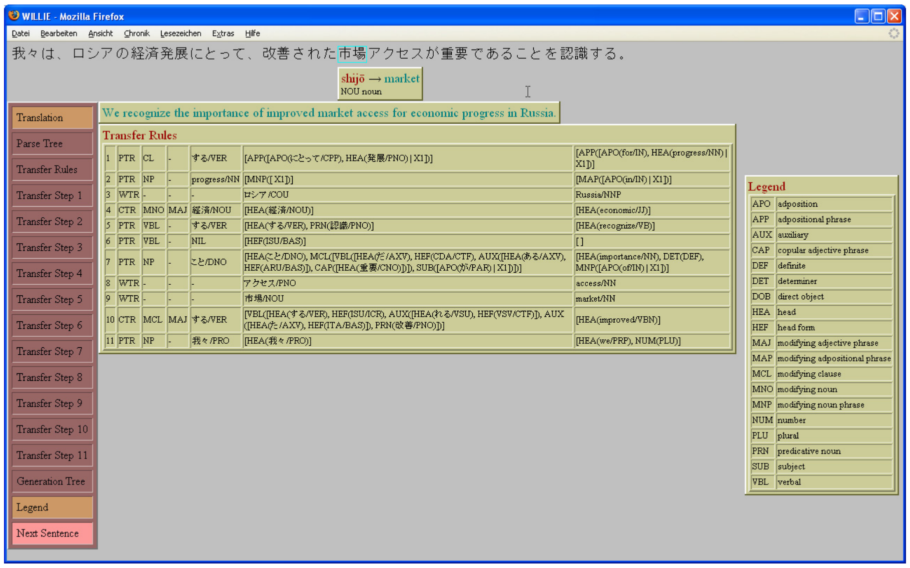
Rule 1 is the default translation of the noun shijō as the noun market. Rule 2 changes the modifying noun (MNO) keizai into the modifying adjective phrase (MAJ) economic. The third argument of the rule is the *head condition*, it is used as an index for the fast retrieval of rule candidates during transfer.

Rule 3 states that for any clause (CL) with head verb suru, the adpositional phrase (APP) "X1 hatten ni totte" has to be replaced by the adpositional phrase "for X1 progress".

The first argument is the *category condition*. CL is an example of a generalized constituent category, the other one being NP (noun phrase). The rule can be applied if the constituent category of the input is subsumed by the generalized category. The second argument is again the head condition, for a clause it is tested on the head of the verbal. Both constituent and phrase transfer rules may contain *shared variables for unification* as shown in Rule 3. This makes it possible to translate only certain parts of the input and to leave the rest intact.

One important requirement for the efficient and robust implementation of the transfer module is that the *argument condition* in the third argument of a phrase transfer rule has to be understood as a subset condition. For example, in Rule 3 it is necessary that the clause contains an adpositional phrase at an arbitrary position with the adposition (APO) "ni totte" and the head noun hatten, both again at arbitrary positions. All other elements of the clause and the adpositional phrase are appended unchanged to the translated required elements.

The language student can inspect the transfer rules used to translate a Japanese sentence by moving the mouse over the "Transfer Rules" entry of the menu bar. The rules are displayed as a numbered tabular list sorted by the sequence of their application (see Fig. 4).



我々は、ロシアの経済発展にとって、改善された市場アクセスが重要であることを認識する。

shijō → market
NOU noun

We recognize the importance of improved market access for economic progress in Russia.

Transfer Rules

1	PTR	CL	-	する/VER	[APP([APO(にとって/CPP), HEA(発展/PNO) | X1])]	[APP([APO(for/IN), HEA(progress/NN) | X1])]
2	PTR	NP	-	progress/NN	[MNP([X1])]	[MAP([APO(in/IN) | X1])]
3	WTR	-	-	-	ロシア/COU	Russia/NNP
4	CTR	MNO	MAJ	経済/NOU	[HEA(経済/NOU)]	[HEA(economic/JJ)]
5	PTR	VBL	-	する/VER	[HEA(する/VER), PRN(認識/PNO)]	[HEA(recognize/VB)]
6	PTR	VBL	-	NIL	[HEF(ISU/BAS)]	[ ]
7	PTR	NP	-	こと/DNO	[HEA(こと/DNO), MCL([VBL([HEA(だ/AXV), HEF(CDA/CTF), AUX([HEA(ある/AXV), HEF(ARU/BAS)]), CAP([HEA(重要/CNO)])]), SUB([APO(が/PAR) | X1])])]	[HEA(importance/NN), DET(DEF), MNP([APO(of/IN) | X1])]
8	WTR	-	-	-	アクセス/PNO	access/NN
9	WTR	-	-	-	市場/NOU	market/NN
10	CTR	MCL	MAJ	する/VER	[VBL([HEA(する/VER), HEF(ISU/ICR), AUX([HEA(れる/VSU), HEF(VSV/CTF)]), AUX([HEA(た/AXV), HEF(ITA/BAS)]), PRN(改善/PNO)])]	[HEA(improved/VBN)]
11	PTR	NP	-	我々/PRO	[HEA(我々/PRO)]	[HEA(we/PRP), NUM(PLU)]

Legend

APO	adposition
APP	adpositional phrase
AUX	auxiliary
CAP	copular adjective phrase
DEF	definite
DET	determiner
DOB	direct object
HEA	head
HEF	head form
MAJ	modifying adjective phrase
MAP	modifying adpositional phrase
MCL	modifying clause
MNO	modifying noun
MNP	modifying noun phrase
NUM	number
PLU	plural
PRN	predicative noun
SUB	subject
VBL	verbal

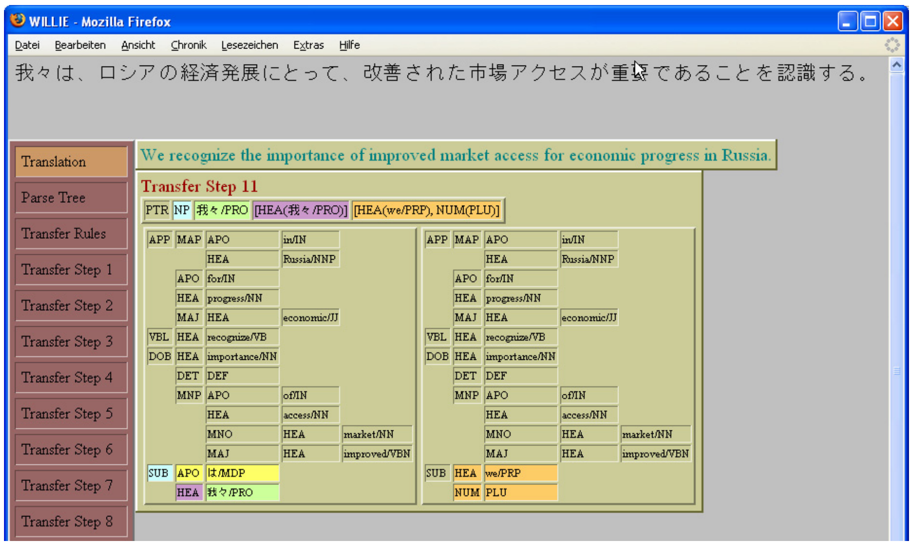
Fig. 4. Example of transfer rules

The transfer module traverses the Japanese parse tree top-down and searches the rule base for transfer rules that can be applied. At the top level we first try to find suitable phrase transfer rules. To apply a phrase transfer rule, we collect all rule candidates that satisfy the condition part and then rate each rule and choose the one with the highest score. The most difficult subtask is the verification of the argument condition because it involves testing for set inclusion at the argument level as well as recursively testing for set equality of arguments of subconstituents.

If no more rules can be applied at the sentence level, each constituent in the sentence is examined separately. We first search for constituent transfer rules before we perform a transfer of the argument. The latter involves the application of word transfer rules for simple constituents, whereas the top-level procedure is repeated recursively for complex constituents. We also perform some common standard transformations. The two most important ones are:

– the removal of redundant Japanese particles that only indicate the relationship of a phrase to the embedding phrase, which is already expressed through the category of the complex constituent,
– the addition of the coordinating conjunction “and”, which is often not explicitly expressed in Japanese.

To give the students a better understanding of the translation process, we provide them with the possibility to inspect the conditions and transformations of each rule application in detail.



我々は、ロシアの経済発展にとって、改善された市場アクセスが重要であることを認識する。

We recognize the importance of improved market access for economic progress in Russia.

Transfer Step 11

PTR	NP	我々/PRO	[HEA(我々/PRO)]	[HEA(we/PRP), NUM(PLU)]

APP	MAP	APO	in/IN	
		HEA	Russia/NNP	
	APO	for/IN		
	HEA	progress/NN		
	MAJ	HEA	economic/JJ	
VBL	HEA	recognize/VB		
DOB	HEA	importance/NN		
	DET	DEF		
	MNP	APO	of/IN	
		HEA	access/NN	
		MNO	HEA	market/NN
		MAJ	HEA	improved/VBN
SUB	APO	は/MDP		
	HEA	我々/PRO		

APP	MAP	APO	in/IN	
		HEA	Russia/NNP	
	APO	for/IN		
	HEA	progress/NN		
	MAJ	HEA	economic/JJ	
VBL	HEA	recognize/VB		
DOB	HEA	importance/NN		
	DET	DEF		
	MNP	APO	of/IN	
		HEA	access/NN	
		MNO	HEA	market/NN
		MAJ	HEA	improved/VBN
SUB	HEA	we/PRP		
	NUM	PLU		

Fig. 5. Example of transfer step

For each transfer step we show the applied rule and, side-by-side, the intermediate trees before and after the application of the rule (*source tree* and *target tree*). There exists a menu entry for each individual transfer step in the menu bar so that the students can slide the mouse over the entries to get an animated view of how the Japanese parse tree gradually changes into the completely translated English parse tree, i.e. the generation tree to compute the final sentence translation.

As can be seen in Fig. 5, we use color coding of source and target trees to convey the semantics of a rule application to the student. We use the following colors for this purpose:

– *blue*: category condition,
– *green*: head condition,
– *violet*: argument condition,
– *orange*: translation of required elements,
– *yellow*: standard transformation.

In the example in Fig. 5, one can easily see that the constituent category SUB is subsumed by NP, the head condition wareware/PRO is satisfied, the argument condition is identical to the head condition, and the translation of the Japanese pronoun wareware is therefore the personal pronoun we/PRP with an additional feature value NUM(PLU) to indicate number plural. Finally, the modifying particle WA/MDP, marking the subject of this sentence, is eliminated.

The main problem with computing the correct color coding is to find the exact location in the intermediate trees where to apply the coloring. The same is true for the generation of the context-specific word translations (see Sect. 5).

Since the original intended use of the system was for machine translation, flexibility and robustness were the two main requirements. Unfortunately, this implied that there remained no trace of the original word order in the resulting generation tree because the order of subconstituents in the arguments of complex constituents could have been arbitrarily rearranged through the application of phrase transfer rules.

We had previously added some limited position data to the generation tree to use the information about the word order in the Japanese sentence to deal with the translation of a sequence of several subconstituents with identical categories (e.g. several modifying adjective phrases), however, the computation of color coding and word translations required a more thorough redesign of all existing modules to incorporate detailed position information.

Starting from the tagging module, we added a position index to each word token beginning with 1 for the first word. The parsing module preserves this position data by adding it to the argument of each simple constituent, e.g. HEA(shijō/14/NOU). During rule acquisition, we also have to learn additional information about the exact word mapping if we translate a phrase through the application of a transfer rule. For example, Rule 3 is now stored as:

```
PTR(CL, suru/VER,
[APP([APO(ni totte/X1/CPP), HEA(hatten/X2/PNO) | X3 ])],
[APP([APO(for/X1/IN), HEA(progress/X2/NN) | X3 ])]).
```

This means that we use again shared variables for unification to preserve the position data during the application of rules by the transfer module. If there is no corresponding word in the other language, then free variables are used, i.e. they do not affect the satisfiability of the rule. Finally, it is also possible to map one Japanese word to several words in English by using the same variable, e.g.:

```
PTR(VBL, suru/VER,
[HEA(suru/X1/VER), PRN(jikkō/X2/PNO)],
[HEA(carry/X2/VB), PVP(out/X2/IN)]).
```

In Japanese grammar, there exist predicative nouns (also called "sahen nouns") that can be used together with the verb suru ("to do") to derive a new verb, like "to integrate" from "integration". In this example, the predicative noun (PRN) jikkō is translated as "carry out", i.e. a phrasal verb with the phrasal verb particle (PVP) out. Therefore, in this situation, a free variable is assigned to the position data of suru, and the position of jikkō is propagated to both carry and out so that later the complete context-specific translation "is carrying out" is assigned to jikkō.

During transfer we store all the intermediate trees with the position information intact. In addition, we number the applied rules and store them together with additional position data depending on the rule type:

- *word transfer rules*: the position of the simple constituent,
- *constituent transfer rules*: the positions of the head elements of the old and new complex constituent,

– *phrase transfer rules*: the positions of the head elements of the complex constituent before and after applying the rule to its argument.

In addition to the applied transfer rules, we also have to store the information about which standard transformations were performed during transfer to guarantee the correct color coding as shown before. To be able to assign each standard transformation to the correct transfer step, we indicate the rule number and the token that has been removed or inserted. In addition, we keep track of the position of the eliminated element, or, for inserted elements, the position of the head of the embedding phrase.

All this information is used by the color coding module to assign the correct color to the individual cells of the tabular display. For that purpose we perform a top-down traversal of the source and target tree and look for the constituent to which the rule is applied. Depending on the rule type, we have to distinguish the following cases:

– *word transfer rules*: the argument of the simple constituent is painted violet in the source tree and orange in the target tree,
– *constituent transfer rules*: the category is painted blue in the source tree and orange in the target tree,
– *phrase transfer rules*: the category is painted blue in the source tree, except for rules at sentence level,
– *standard transformations*: the category and the argument is painted yellow in the source tree for eliminations, in the target tree for insertions.

If the anchor constituent has been found for a constituent or phrase transfer rule, then the argument condition and its translation is analyzed recursively. The categories and arguments of all matching subconstituents are colored violet in the source tree and orange in the target tree, except for the argument of the head condition, which is colored green in the source tree.

5 Translations

The final task on the way to a completely translated Japanese sentence is the generation of the surface form of the English sentence as character string.

The input to the generation module is the *generation tree*, i.e. the final parse tree after applying all transfer rules. The generation tree can be inspected side-by-side with the original Japanese parse tree by moving the mouse over the corresponding menu entry.

The generation module traverses the generation tree top-down and transforms the argument of each complex constituent into a list of surface strings. The list is computed recursively from the subconstituents and flattened afterwards. The correct surface form for words with irregular inflections is computed by accessing the English lexicon.

In addition to the sentence translation, as mentioned before, we add context-sensitive word translations to the lexical entries in the token list. In a first processing step, we flatten the information in the generation tree by traversing it top-down and asserting a dynamic fact for each simple constituent indicating the position, the constituent category, and the argument. In addition we assert dynamic facts for complex constituents that influence the generation of word translations, e.g. genitive noun phrases.

Next, we process the token list from left to right. The default treatment for a lexical entry is to use the English token if a single dynamic fact exists for that list position. Otherwise, the word translation is left empty.

If there exist several dynamic facts for a position, they are used to generate the correct surface form of conjugated words as well as concatenate the individual elements of phrases.

Most problems with the assignment of word translations to the correct positions in the token list have already been resolved through the extension of the acquisition module (see Sect. 4). Therefore, there remain only few special cases to be dealt with, e.g. for predicative nouns, the features regarding conjugation have to be collected from the next position for the verb suru to produce the correct surface form. Finally, we also have to ensure the correct placement of:

– words inserted through standard transformations,
– commas and other punctuation marks,
– words inserted due to complex constituents, e.g. "to" in a modifying to-infinitive clause.

6 Conclusion

In this paper we have presented a Web-based language learning tool for Japanese, which has been developed based on an existing machine translation system. WILLIE randomly selects translation examples and displays detailed information about lexical, syntactic, and translation knowledge by using dynamic JavaScript pop-up windows and color coding. We have finished the implementation of the system including a first local prototype configuration of the Web server to demonstrate the feasibility of the approach.

Future work will focus on making WILLIE available to students of Japanese studies at our university to receive valuable feedback from practical use, in particular regarding usability. In addition, we are planning to make a demo version of WILLIE publicly available in the near future.

Although WILLIE is already a very useful tool for language students, we also see it only as a first step towards the larger aim of developing an intelligent language tutoring system. We want to extend WILLIE so that it is possible for the student to enter free input to be analyzed. This should then be embedded into interactive exercises with meaningful feedback.

Acknowledgements

We are deeply grateful to Amzi! inc. for supporting our research with a complimentary Professional Edition of Amzi! Prolog + Logic Server.

References

1. Hutchins, J.: Machine translation and computer-based translation tools: What's available and how it's used. In: Bravo, J. M., ed.: A New Spectrum of Translation Studies, University of Valladolid (2004) 13–48
2. Liu, H.: MontyLingua: An End-to-End Natural Language Processor with Common Sense. MIT Media Lab (2004)
3. Matsumoto, M. et al.: Japanese Morphological Analysis System ChaSen Version 2.0 Manual. NAIST Technical Report, NAIST-IS-TR99009 (1999)
4. Nagata, N.: Intelligent computer feedback for second language instruction. Modern Language Journal **77(3)** (1993) 330–338
5. Nagata, N.: An effective application of natural language processing in second language instruction. CALICO Journal **13(1)** (1995) 47–67
6. Nagata, N.: Computer vs. workbook instruction in second language acquisition. CALICO Journal **14(1)** (1996) 53–75
7. Nagata, N.: An experimental comparison of deductive and inductive feedback generated by a simple parser. System **25(4)** (1997) 515–534
8. Nagata, N.: BANZAI: An application of natural language processing to Web based language learning. CALICO Journal **18(4)** (2002) 583–599
9. Nagata, N.: BANZAI: Computer assisted sentence production practice with intelligent feedback. Proc. of the 3rd Intl. Conf. on Computer-Assisted Systems for Teaching and Learning Japanese, San Diego, USA (2002)
10. Nagata, N.: Robo-Sensei: Personal Japanese Tutor. Cheng & Tsui (2003)
11. Nagata, N., Swisher M. V.: A study of consciousness-raising by computer: The effect of metalinguistic feedback on second language learning. Foreign Language Annals **28(3)** (1995) 337–347
12. Newton, J., ed.: Computers in Translation: A Practical Appraisal. Routledge (1992)
13. Somers, H., ed.: Computers and Translation: A Translator's Guide. John Benjamins (2003)
14. Utiyama, M., Isahara, H.: Reliable measures for aligning Japanese-English news articles and sentences. Proc. of the 41st Annual Meeting of the ACL, Barcelona, Spain (2003) 72–79
15. Winiwarter, W.: WETCAT – Web-enabled translation using corpus-based acquisition of transfer rules. Proc. of the 3rd IEEE Intl. Conf. on Innovations in Information Technology, Dubai, United Arab Emirates (2006)
16. Winiwarter, W.: Automatic acquisition of translation knowledge using structural matching between parse trees. Proc. of the First Intl. Conf. on the Digital Society, Guadeloupe, French Carribean (2007)
17. Yang, J., Akahori, K.: Development of computer assisted language learning systems for Japanese writing using natural language processing techniques: A study on passive voice. Proc. of the AIED-Workshop on Intelligent Educational Systems on the World Wide Web, Kobe, Japan (1997)
18. Yang, J., Akahori, K.: An evaluation of Japanese CALL systems on the WWW. Comparing a freely input approach with multiple selection. Computer Assisted Language Learning **12(1)** (1998) 59–79
19. Yang, J., Akahori, K.: Error analysis in Japanese writing and its implementation in a computer assisted language learning system on the World Wide Web. CALICO Journal **15(1-3)** (1998) 47–66