Research Report

Multilingual Generalization of the ModelCreator Software for Math Item Generation

Derrick Higgins

Yoko Futagi

Paul Deane

Research & Development

April 2005
RR-05-02

Multilingual Generalization of the ModelCreator Software for Math Item Generation

Derrick Higgins, Yoko Futagi, and Paul Deane

ETS, Princeton, NJ

April 2005

As part of its educational and social mission and in fulfilling the organization's nonprofit charter and bylaws, ETS has and continues to learn from and also to lead research that furthers educational and measurement research to advance quality and equity in education and assessment for all users of the organization's products and services.

ETS Research Reports provide preliminary and limited dissemination of ETS research prior to publication. To obtain a PDF or a print copy of a report, please visit:

www.ets.org/research/contact.html

EDUCATIONAL TESTING SERVICE, ETS, and the ETS logo are registered trademarks of Educational Testing Service.

Copyright © 2005 by Educational Testing Service. All rights reserved.

Abstract

This paper reports on the process of modifying the ModelCreator item generation system to produce output in multiple languages. In particular, Japanese and Spanish are now supported in addition to English. The addition of multilingual functionality was considerably facilitated by the general formulation of our natural language generation system, which proceeds in three stages, whereby the first two stages are largely language-independent. This multilingual capability is very promising for the domain of educational assessment because, when combined with *item modeling*, it provides the potential to produce a large pool of automatically constructed items, with difficulty estimates and balanced forms for speakers of different languages.

Key words: Natural language generation, Spanish, Japanese, item modeling, natural language processing (NLP)

Acknowledgements

We would like to thank Aurora Graf, Kathy Sheehan, and Isaac Bejar for insightful discussion regarding this project. Cecilia Rosenblum was instrumental in ensuring the quality of the Spanish language generation system. ETS also has our gratitude for its support of this work.

Overview

This research report concerns the development of a multilingual adaptation of the ModelCreator item generation system. Whereas the original version of ModelCreator produced test items in English, the newest version uses a common generation system to produce items in English, Spanish, or Japanese.

ModelCreator is a system for automatic item generation (AIG; cf. Irvine & Kyllonen, 2002; Bennett, in press), specifically the creation of mathematical word problems. Deane (2003) is a general description of the motivation, aims, and progress of the ModelCreator project. In the current report, we will reprise Deane's major points regarding the general design of ModelCreator, but the remainder of the report focuses on the goal of multilingual adaptation of the system.

In particular, we will describe ongoing work to extend ModelCreator's capacity to Japanese and Spanish. While English and Spanish are closely related languages, Japanese substantially differs from English in syntax, morphology, and the formal realization of semantic distinctions. A long-term goal of the project is to enable the system to handle a larger set of languages, but our experience with Spanish and Japanese has been revealing as these are the first set of adaptations we have undertaken. Moreover, they represent the range of languages that our underlying model can potentially deal with. While Spanish represents a sort of lower bound in the amount of development effort required to add a new language, Japanese provides an estimate of the upper end of the difficulty spectrum because Japanese and English represent entirely different historical and typological language groups.

Test Adaptation and Item Generation

One of the major topics of this paper is *test adaptation* (Stansfield, 2003; Hambleton & Patsula, 1999; Hambleton & Kanjee, 1995), which is the modification of an existing test for application to a new group of individuals who typically speak another language than that of the original test. The specific contribution we hope to make here is the facilitation of automatic, multilingual item generation, and the ability to generate test items of the same content in multiple languages.

While addressing this task is of great importance for test adaptation, it is still only a part of the complete set of considerations involved in adapting a test. As Hambleton and Patsula (1999) and Sireci and Berberoğlu (2000) point out, having two forms of a test, where one is a translation of the other, does not guarantee that the test has been suitably adapted for a given population. Cultural factors must also be taken into consideration in the adaptation process, and steps must be taken to ensure construct equivalence between the two forms.

The main advantages that ModelCreator has to offer in the domain of test adaptation are as follows:

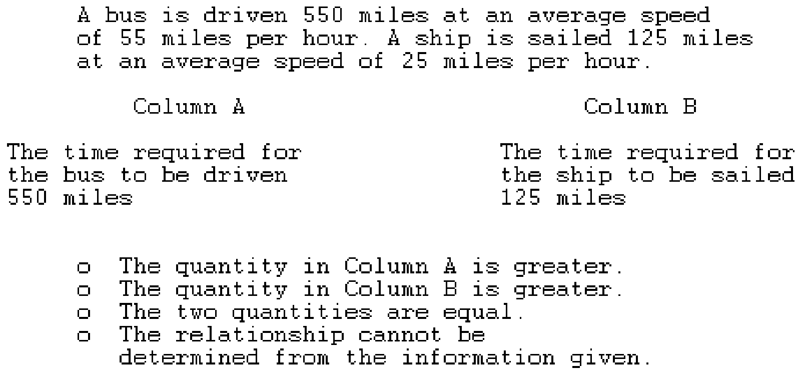
- Because we generate all linguistic variants of an item from the same abstract item model, there is no asymmetry between these variants, and the task of establishing uniformity of content across languages is accordingly simplified. (For example, we do not need independent treatment of active-voice and passive-voice variants of the same basic item.)
- The automatic nature of the generation of items in different languages guarantees a uniform linguistic correspondence between test items in different languages, which would be difficult to achieve using manual translations.
- The relative ease with which new languages and new content can be incorporated into ModelCreator has beneficial consequences for the entire test adaptation process, because it enables test developers to focus on the more subtle issues of adaptation, such as cultural appropriateness, rather than the gross outlines of translation.

Beyond test adaptation, of course, the ModelCreator generation system may simply be used as a way of generating items in each of the *single languages* in its repertoire (which is the aim of Bejar, Fairon, & Williamson, 2002). Even in a setting where there is no need for the cross-linguistic generation capability made possible by ModelCreator's architecture, the generation of test items in Japanese or other languages will be of value.

The Problem Domain: Generation of Math Word Problems

The ModelCreator item creation system is constantly undergoing modifications to extend the set of questions that it can produce. The initial set of items added to the system was distance-rate-time problems. Since that first stage of development, the English-language generation capacity of the system has been extended to produce interest-accrual problems, while work-rate problems are currently being added. At present, only distance-rate-time items are supported in Spanish and Japanese.

All of these items are in the *quantitative comparison* (QC) format. QC items have a *stem*, in which general information relevant to the item is presented, followed by two columns, which express quantities the test taker is asked to compare. The task presented to the examinee is to choose whether the first quantity is greater, the second quantity is greater, the two are equal, or their relative magnitude cannot be determined. Figure 1, a screen shot from the English version of ModelCreator, illustrates this format:



```
          A bus is driven 550 miles at an average speed
          of 55 miles per hour. A ship is sailed 125 miles
          at an average speed of 25 miles per hour.

              Column A                        Column B

     The time required for           The time required for
     the bus to be driven            the ship to be sailed
     550 miles                       125 miles


          o  The quantity in Column A is greater.
          o  The quantity in Column B is greater.
          o  The two quantities are equal.
          o  The relationship cannot be
             determined from the information given.
```

Figure 1. **Distance-rate-time item produced by the ModelCreator system.**

As Deane (2003) pointed out, the distance-rate-time domain is particularly well-suited for an item generation system because the verbal material is taken from a fairly circumscribed and well-defined domain, which lends itself to modeling because of the straightforward underlying mathematical construct (D = R × T), and the fact that the elements of the test construct (such as distance and time) can be mapped fairly directly to elements in the eventual item (such as distance and rate *expressions*). The same holds true of the interest and work problem item types that are currently being added.

Template-Driven Generation vs. Natural Language Processing

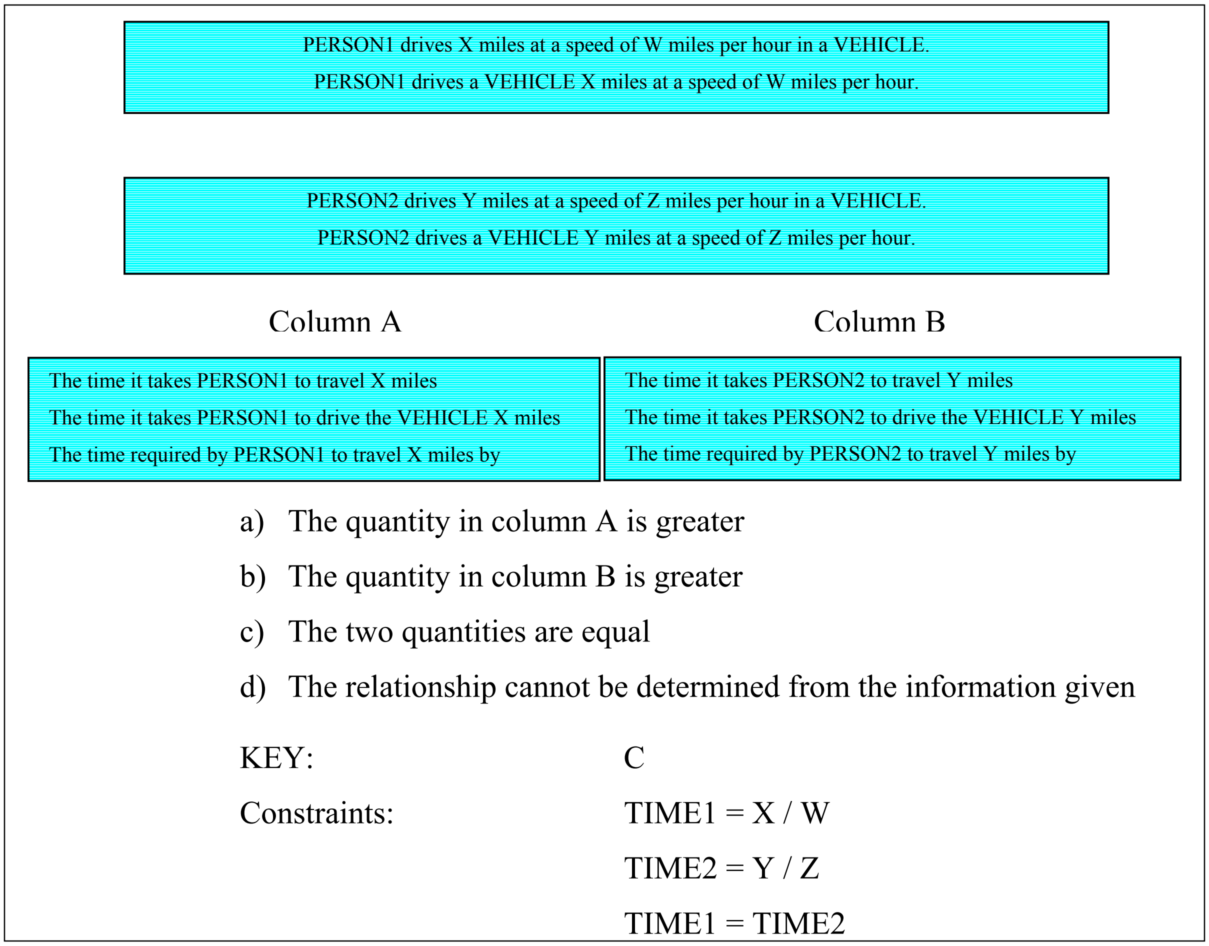
Much previous work in AIG, including Singley and Bennett (2002), has used item templates as a primitive in the generation process. An item template basically consists of a sentence frame, with all details of wording determined beforehand, in which words of certain types may be substituted from a list. The coverage of the item generation system under such an approach would be jointly determined by the set of templates and the set of word lists available for substitution.

Figure 2 illustrates this with a simple example of a template-based approach to creating distance-rate-time word problems in the quantitative comparison format. In the stem of the item and each of its columns, a template sentence must first be chosen from each shaded box. As shown at the bottom of the figure, a completed template may also specify the key of items derived from it, and any constraints that the variables of the template must satisfy. The filler lists in Figure 3 are then used to populate the templates with English words. Since there are three template choices for each of the four sentences, and four choices for the person name and vehicle type, this simple example already can produce 3 × 3 × 3 × 3 × 4 × 4 × 4 = 5,184 distinct items, even disregarding the possible variation in the numbers used for rates and times. (Of course, not all of these item instances may be equally intersubstitutable. Names and vehicle types are likely to function as *incidentals*, which do not affect difficulty, whereas the specific numbers used in an item are more likely to be *radicals*, which may be relevant to the item's difficulty. See the first chapter of Irvine & Kyllonen, 2002, for additional discussion.).

Figures 2–3 represent what is known as an *item model* (cf. Bejar et al. 2002; LaDuca, Staples, Templeton, & Holzman, 1986), a specification for a class of isomorphic items that have comparable content and the same psychometric properties. Particular items derived from the same item model may be referred to as *instances* of the model. Templates are one way of specifying an item model (by manually creating all of the linguistic content that is common to items derived from the model), but it is not the only way. As we shall see below, ModelCreator's fundamental innovation is its capacity to automatically construct item models.

A template-based approach is attractive for its initial simplicity; a large number of new items can be created with minimal effort. All that is required is the specification of a template, along with the possible variations in each slot. Given this immediate payoff from the template-based approach, moving beyond this to more sophisticated natural language processing (NLP) techniques requires some justification.

Templates run into problems when the domain is extended beyond the simple range of items covered in this example. First, the development effort expended in creating a set of templates does not help to reduce the effort required in creating new templates. (This is the issue of *linguistic generalization*.) Because templates are composed of sentences that are all but complete and thus do not include generalizations over grammatical structures, the model designer must start from scratch in creating new templates. The ModelCreator approach we describe below addresses this problem by allowing for the automatic construction of item models, which are in essence like templates.



PERSON1 drives X miles at a speed of W miles per hour in a VEHICLE.
PERSON1 drives a VEHICLE X miles at a speed of W miles per hour.

PERSON2 drives Y miles at a speed of Z miles per hour in a VEHICLE.
PERSON2 drives a VEHICLE Y miles at a speed of Z miles per hour.

Column A	Column B
The time it takes PERSON1 to travel X miles The time it takes PERSON1 to drive the VEHICLE X miles The time required by PERSON1 to travel X miles by	The time it takes PERSON2 to travel Y miles The time it takes PERSON2 to drive the VEHICLE Y miles The time required by PERSON2 to travel Y miles by

a) The quantity in column A is greater

b) The quantity in column B is greater

c) The two quantities are equal

d) The relationship cannot be determined from the information given

KEY: C

Constraints: TIME1 = X / W

TIME2 = Y / Z

TIME1 = TIME2

Figure 2. **Example template for a mathematics quantitative comparison item.**

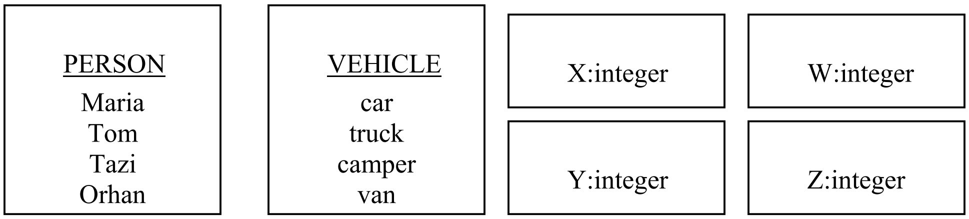


Figure 3. **Example lists of fillers for template slots.**

Second, the structure of a template-based item model makes it difficult to encode the dependencies among the filler items chosen (the *interslot dependency* problem). For example, it is reasonable to create items that involve cars traveling at 35 miles per hour, but not at 400 miles per hour, whereas jet airplanes are more likely to travel at 400 mph than at 35 mph. Since the filler items for template slots are chosen independently of one another, these generalizations are not easily represented. It is true that a set of constraints may be added to the specification of a template in order to account for these dependencies, but this straightforward approach is fraught with complications when attempted on a large scale. The relevant set of constraints (whether it has to do with the expected speeds for different classes of vehicles or the class of motion verbs that may be used with an explicitly expressed endpoint, etc.) must be specified independently for each template used for generation. In addition, no set of constraints incorporated into a template-based generation system can control for the sort of dependencies that may hold between the words chosen for a template slot and the grammatical structure of the item. Since the basic linguistic structure of the template is fixed, any words that are incompatible with it simply have to be excluded.

Third, developing a large-scale generation system based on templates can lead to a system that is excessively fragile. To cover a large fragment of the domain under consideration, a large set of templates may be required, which can quickly become unmanageable. Finally, and most importantly, a template-based approach to generation is not ideal for generation in multiple languages, since the templates must be developed independently for each language.

The ModelCreator Approach

The approach to generation taken in the ModelCreator project introduces an additional degree of abstraction and automation, compared with a template-based system. The user need only specify high-level generation parameters via the graphical user interface, rather than the full set of options comprising an item model, as in a template-based approach. This basically amounts to automatic generation of item templates, a move necessitated by the problems of scalability and self-consistency we have noted with the manual production of templates.

In fact, there are intermediate levels of automation between the manual specification of templates and a fully automatic system for template generation as found in the ModelCreator project, just as there are intermediate steps between manual item writing and schematic use of item models specified by templates. The work of Fairon and Williamson (2002) represents a sort of “third path” between a straightforward template-based approach to generation and the deep generation done by ModelCreator. Their system is basically template driven and uses canned text for much of the content it produces; however, it does have some local rules that allow it to enforce grammatical constraints such as subject-verb agreement in its automatically generated text.

The approach to generation taken in the ModelCreator project requires a more intensive initial development effort than a template-based approach, but is also more scaleable and robust. It is scaleable in that it can more easily be extended to handle a wider range of constructions within a language or to entirely new languages and content areas. It is robust in that the back-end linguistic processing takes care of such niceties as subject-verb agreement and gender concord, so that malformed item instances are less likely to result from carelessly constructed models. Through the use of natural language processing techniques, this system also solves the problems of interslot dependency and linguistic generalization alluded to above.

The problem of interslot dependency is addressed in part by using a frame-semantic representation (cf. Fillmore 1968, 1985; Baker, Fillmore, & Lowe, 1998; Fillmore & Baker, 2001) for the vocabulary of the item domain. This *frame database* indicates the semantic relationships between different participant roles in the activity described in a given item type, so that only permissible combinations are produced. For instance, a distance-rate-time event frame might contain the role types AGENT, VEHICLE, DISTANCE, TIME, RATE, ORIGIN, and GOAL, among others, reflecting our knowledge that a traveling event involves a person navigating a vehicle from some origin to an endpoint, covering a certain distance at a certain rate. In addition to specifying these roles, our frame semantic representation indicates constraints on the lexical elements that may fill each role. For instance, if the RATE role is chosen to be greater than 35 miles per hour, the VEHICLE chosen should not be a bicycle, and if the verb of motion used is *sail*, the VEHICLE role must be filled by some sort of watercraft.

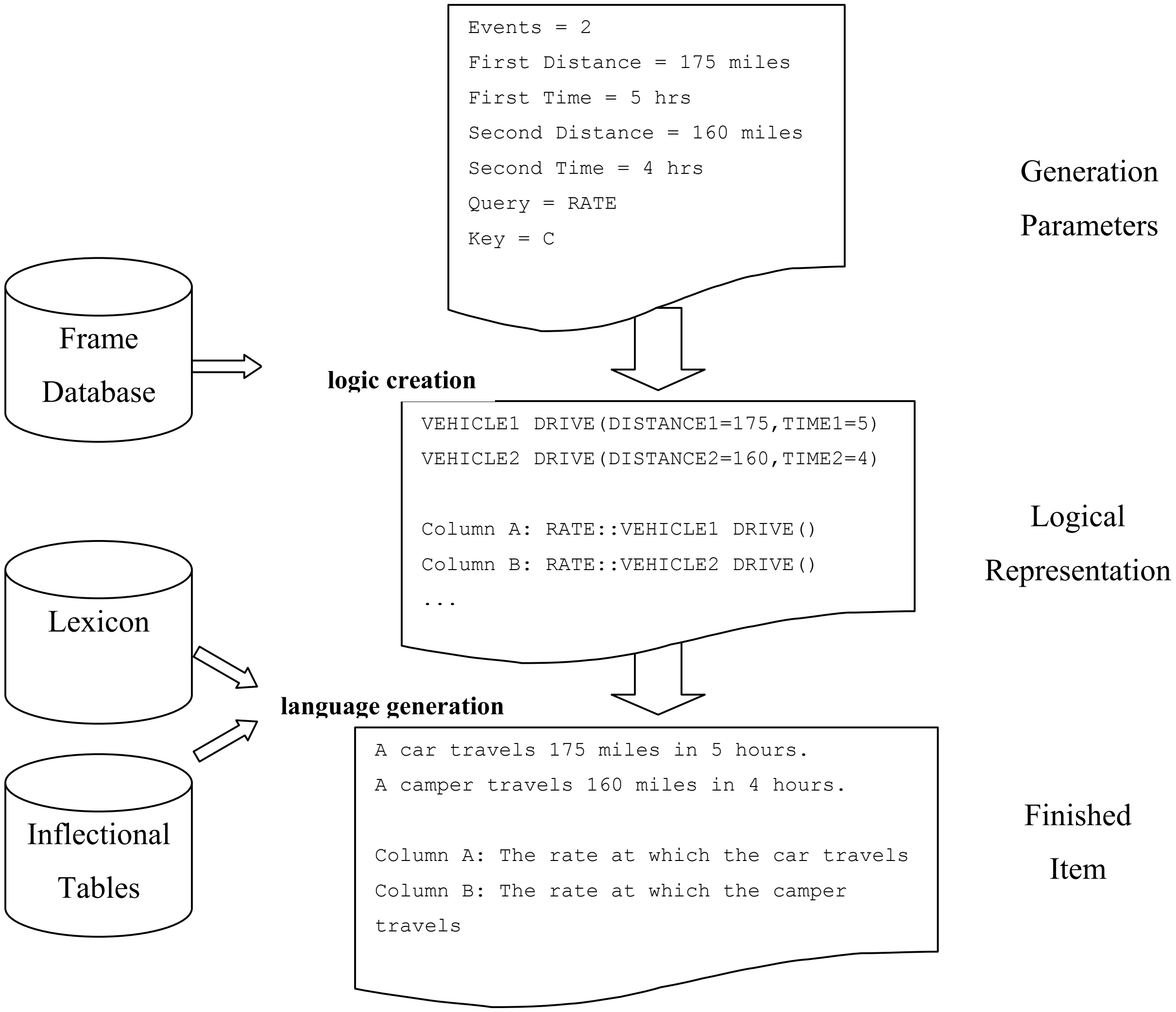
This frame database consists of an inheritance hierarchy of frames, in which a more specific frame may inherit most of its values from a more general one. For instance, we can implement a general frame that expresses the properties of travel events conducted by air. This frame will indicate, among other things, that the vehicle involved in an air travel event can be a jet or an airplane, that the typical motion verb for these events is *fly*, and that a typical endpoint of travel is an airport. We could also introduce a more specific “helicopter-travel” frame that inherits all of the properties of the more general air travel frame, but augments these with attributes that hold only for travel by helicopter. For example, a helipad is a fine landing point for a helicopter, but not for any other sort of aircraft.

In addition to the filler dependencies that arise from semantic factors and are handled by the ModelCreator frame database, there are dependencies that arise from more formal (less content-related) aspects of language. For example, if the element filling the VEHICLE role (say, *camper*) has been mentioned previously in the item, then subsequent uses must refer to it as *the camper* rather than *a camper*. In Japanese, a VEHICLE used as a direct object requires the case suffix を (*-o*), while as a subject it requires another case suffix: が (*-ga*). Such facts can be handled directly by writing them into templates, as in Figure 2. However, doing so is a less than ideal solution because general facts about case marking and definiteness have to be encoded anew for each template sentence that is added to the generation system.

Such are the complications and ad hoc solutions that a large-scale template-based approach would require in order to get acceptable results for a reasonable range of grammatical phenomena. To avoid such problems, ModelCreator employs a full-fledged linguistic model, which determines the correct grammatical structure to produce on the fly. This automation of the process of producing templates helps to alleviate their deficiencies. In particular, automatic generation of item templates entails a preexisting categorization of them, which constitutes a solution of the linguistic generalization problem. In addition, the interslot dependency problem is solved by our use of a semantic frame database that specifies the admissible participants for various event types and reasonable constraints that may be placed on them.

In a first pass, general information about the item type desired is used to construct a pseudological representation, which is largely language-neutral, but represents the participants involved in the event to be described and how this information is packaged in the stem and query expressions of the item. From this logical representation of the item, a linguistic generation step produces the completed item, using linguistic analysis to produce the sentence forms for the item. This process is represented in Figure 4 below.

The *generation parameters* in Figure 4 consist of a set of choices made through the program interface by choosing from graphical menus, checking boxes, and so forth. They include such aspects of the distance-rate-time item as the number of vehicles traveling, the distances involved in each event, and whether the item should query the examinee for a rate, time, or distance quantity.



```
Events = 2
First Distance = 175 miles
First Time = 5 hrs
Second Distance = 160 miles
Second Time = 4 hrs
Query = RATE
Key = C
```

```
VEHICLE1 DRIVE(DISTANCE1=175,TIME1=5)
VEHICLE2 DRIVE(DISTANCE2=160,TIME2=4)

Column A: RATE::VEHICLE1 DRIVE()
Column B: RATE::VEHICLE2 DRIVE()
...
```

```
A car travels 175 miles in 5 hours.
A camper travels 160 miles in 4 hours.

Column A: The rate at which the car travels
Column B: The rate at which the camper
travels
```

Figure 4. **ModelCreator's generation process.**

These item parameters are considerably more abstract than the information found in an item template, and therefore more easily incorporated into a point-and-click style user interface. This allows us to use an entirely menu-driven interface for ModelCreator, which gives the user fine control over the specific details of items the system generates, yet does not require programming skills or other technical knowledge on the user's part.

The level of *logical representation* in Figure 4 roughly corresponds to the information that must be specified in an item template. In template-based approaches, the templates are individually authored, a process that can be time-consuming and requires considerable expertise. Using ModelCreator, however, these templates are generated automatically given a specification of the generation parameters. The logical representation is the output of a *logic creation* process, which packages the information for the item into discrete propositions and determines which information will be provided in the stem of the item and which will be part of the options in each column. The semantic frame database is applied at this stage to determine constraints on which participant roles can occur together. For example, a TIME expression occurring together with a VEHICLE expression and a DISTANCE expression has two different permissible expressions. One way of packaging the information is to use a DRIVE-type predicate (e.g., *The car* ***drove*** *100 miles in 2 hours*). Another is to use both a REQUIRE predicate and a DRIVE predicate (*The car* ***required*** *2 hours to* ***drive*** *100 miles*).

Finally, the finished item is created from this logical representation by means of our *natural-language generation* (NLG) component. This process adds the linguistic substance to the logical outline. It uses the lexicon to look up appropriate terms for each of the logical elements, which is analogous to filling out the slots in a template. However, this slot-filling is done in a linguistically informed way, using our knowledge of the language's structure to ensure that the language generated for the item is well-formed. Among other things, this involves using our inflectional tables to ensure that each word appears in the correct form to reflect tense, case, gender, or whatever other features may influence its realization.

The Strategy for Multilingual Generalization

The design of the generation process described above, in which a logical representation of the content is generated as a first step and language-specific elaboration of this logical structure is performed afterwards, has clear consequences for the strategy to be pursued in extending the item generation system to a new language. Ideally, the generation of a semantic representation ought to be a largely language-independent matter, although as we shall discuss below, in some cases the structures encode some language-specific phrasing options. The bulk of the work, then, for extending the ModelCreator system to a new language consists in redesigning the component of the system that fleshes out the frame semantics for a given item, adding the lexical and grammatical "meat" to the semantic skeleton.

Modifications for Japanese

In this section, we present a broad outline of the modifications that were necessary in order to produce a working NLG program to generate test items in Japanese, starting with the English version of the ModelCreator software. On the whole, it is remarkable how localized these changes were in the ModelCreator code, given the great differences between English and Japanese.

Historically and typologically, English and Japanese are radically different languages (Comrie, 1987). There is no known genetic relationship between the two languages; even the most distant known relatives of English, such as Farsi and Sanskrit, are members of the Indo-European family, to which Japanese does not belong. Structurally, while Japanese does not have features such as an ergative-absolutive case system or noun incorporation, which are perhaps the most exotic from the viewpoint of an English speaker, it nevertheless represents a very different linguistic type from English and other Western European languages. Where in English, nominal case is marked only on pronouns and even there only in a very restricted paradigm, Japanese has a full-fledged case system. In English, "functor" categories such as verbs and prepositions generally precede their arguments, so that the normal phrase order for an active declarative sentence is subject-verb-object (SVO). In contrast, Japanese is a head-final language, in which the default word order is subject-object-verb (SOV), and "prepositions" follow the noun (so they are actually *postpositions*).

Despite extensive linguistic differences such as these, though, the development of Japanese ModelCreator did not require complete duplication of the effort expended in developing the English system. Rather, only certain areas of the program code needed to modified. Thanks to the very general architecture of the original English ModelCreator system, the modifications for Japanese could be restricted to the bare minimum, targeting those respects in which the grammar of Japanese differs from that of English. Note that in a template-based generation system, there is no clear distinction drawn between language-specific grammatical generalizations and more universal aspects of item structure (such as how information is packaged in the different textual parts of an item).

Japanese Grammatical Structure

In this subsection, a more specific description of differences between Japanese and English is given. Since it is impossible to discuss every single difference between the two languages, we will limit ourselves to those parts of Japanese grammar that anyone attempting sentence generation both in English and Japanese must address. We will start with the writing system and then move on to more syntactic and morphological differences.

Writing system. Besides grammatical differences, one major difference between English and Japanese is the writing system. English employs an alphabet, in which one or more letters represent a segment (a single speech sound). For example:

(1) English alphabet:

a pp le — conventional spelling
| | |
[æ p l] — phonetic representation

In this example, a letter *a* stands for the segment [æ], a sequence of letters *pp* for [p], and *le* for [l] (or *l* stands for [l] and *e* is silent).

On the other hand, Japanese uses two writing systems, which are both different from an alphabet, simultaneously: *Kanji*, or Chinese characters, in which each character represents one or more syllables as well as some semantic content, and a *syllabary*, in which each letter represents a single syllable rather than a segment. The following is an illustration of the system:

(2) Japanese writing system:

a. *Kanji:*

Each complete character represents a meaning:

for example, 手 "hand" ↘   ↙ 寺 "temple" [ji]

$\boxed{持}$ "hold; have" [ji]

b. *Kana*

Each symbol represents a syllable, rather than a segment:

for example, く る ま "car"

[ku ru ma]

In (2a), a simplified form of an existing *Kanji* 手 "hand" becomes a part of another *Kanji* 持 "hold; have" (in this case, the left half). Here, "hand" is contributing a part of the meaning of 持 "hold; have" in that it has something to do with hands. At the same time, another existing character 寺 "temple" becomes the right half of 持, contributing the sound [ji].[1] For a noncomposite *Kanji* like 手, one must learn its pronunciation(s), for there is nothing in the character to provide a clue. Even for composite characters like 持, one must learn various pronunciations to which no clue is found in the character itself.

On the other hand, *Kana*, a syllabary, is closer to an alphabet in that it is a writing system that represents sounds, not meaning. There are two variants of *Kana*: *Hiragana* and *Katakana*. In the last few decades, *Katakana* has come to be used almost solely for loan words. Chinese loan words are still usually written in *Kanji*, but their pronunciation is almost always written in *Katakana*. Loan words from other languages are always in *Katakana*. The major difference between *Kanji* and either system of *Kana*, as mentioned above, is that each *kana* represents a syllable, which, in the case of Japanese, is a combination of one optional onset (syllable-initial) consonant and a nucleus (the most sonorous and prominent part of a syllable; in Japanese, a vowel). The modern Japanese syllabary uses separate letters for optional coda (syllable-final) consonants, of which there are only two possibilities: ん for a nasal consonant, or small っ (as opposed to full-size つ) for a geminate consonant. *Kana* has also come to mark long consonants by either using the vowel symbol which actually represents the long vowel (for vowels [a, i, u]) or a vowel symbol that represents the sound that is higher than the actual long vowel, making the long vowel look as if it is a diphthong ([u] for [o] and [i] for [e]). The example in (2b) is a simple one, in which each syllable represented by a *kana* is CV (consonant-vowel).

There is a basic division of labor between *Kanji* and *Kana* in the Japanese writing system in normal circumstances. *Kanji*, whose main characteristic is representing meaning, are used for content words and morphemes such as nouns and the (partial) stem of verbs, adjectives, and adjectival nouns. *Kana*, which represent sounds, are used for function words/morphemes, such as case markers, postpositions, various suffixes such as tense, aspect, negation, and so forth. The following is an example to illustrate this division of labor:

(3)　ジョンは、　白い　車を　持 っている。
John-TOPIC　white　car-Acc　have-STATIVE-PRES
"John has a white car."

First, let us look at the content words. ジョン "John" is a foreign name, thus is written in *Katakana*, one of the two types of *Kana*. 白 "white" is an adjective stem, thus it is in *Kanji.* 車 "car" is a noun, and 持 "have" is a verb stem, so they are also written in *Kanji*. Now turning to function morphemes, the topic marker は, adjective inflection い, the accusative marker を, the stative aspect suffix いる and the verbal conjugation form that is required by the suffix って are all written in *Kana*. However, if we are writing for younger children, for example, lower-grade elementary-school children, all or much of this writing is in *Kana*. This is because more complex *Kanji* are generally taught at a higher grade level. This means that different age groups require not only different vocabulary in their reading material, but a different proportion of *Kanji* in the writing system for the same vocabulary, starting from all-*Kana* for very young children.

Word order. One of the major differences between English and Japanese is the basic word order. English is a so-called SVO language. Thus, the arguments of a verb other than the subject in English always follow the verb. For example:

(4)	John	drove	a car	to the airport.
	S	**V**	O	Goal

On the other hand, Japanese is an SOV language, in which arguments of a verb always precede the verb. The following example is the Japanese counterpart of (4) above:

(5)	ジョンが	車を	空港に	運転していった。
	John-ga	kuruma-o	kuukou-ni	untenshite-it-ta.
	John-NOM	car-ACC	airport-to	drive-go-PAST
	S	O	Goal	**V**
	“John drove a car to the airport.”			

In the English example (4), the verb is in the middle of the sentence, while in the Japanese counterpart (5), it is at the end of the sentence.

Nouns: Case and topic markers and pre- vs. postpositions. These examples also show other syntactic and morphological differences between English and Japanese. One such difference is the presence of “(syntactic) case markers” on nouns in Japanese but not in English. There are three nouns in each example: *John/ジョン, car/車*, and *airport/空港.* In (4), none of the nouns are marked for case, while in (5), *ジョン* and *車* are marked nominative (NOM) *が* (*-ga*) and accusative (ACC) *を* (*-o*), respectively. The other noun, *airport*, is preceded by the preposition *to* in English, while its counterpart, 空港,is followed by the postposition *–ni* in Japanese.

Modern English is not particularly rich in morphology. Syntactic case marking is still observable in its pronoun system—for example, *I* for subject, *me* for object, and *my* for possessive—but regular nouns and names are only marked for the possessive: *John's* as opposed to *John* (either subject or object). On the other hand, Japanese nouns and names, as well as pronouns, are almost always followed by a case or topic marker unless they are followed by a postposition. (Note 空港 in (5), which is followed by a postposition, thereby ruling out a case/topic marker.)

There are four case markers in Japanese: nominative *-ga*, which typically marks subjects; accusative *-o*, which typically marks the direct object; dative *–ni*, which marks the direct object of some verbs and indirect object of other verbs; and genitive *–no*, which marks the possessive. Japanese also has a topic marker, *-wa*, which is often used to mark subjects instead of *–ga*, as well as topic noun phrases.[2]

As is typical of SVO languages, English has prepositions such as *at, to, from, for, of, by*, and so on which always precede their complement nouns. Japanese, as is typical of SOV languages, has postpositions, which always follow their complement nouns (Greenberg , 1966). While there is some semantic correspondence between English prepositions and Japanese postpositions, it is far from complete. This is to be expected when dealing with two or more languages, because languages differ in lexicalization of semantic concepts. This is true even between two closely related languages and is certainly true between such unrelated languages as English and Japanese.

One of the major differences in noun phrases between English and Japanese, which pertains to the distance-rate-time problem type, is the relative clause construction. In English, a head-initial language, a relative clause *follows* the head noun, and there is a relative pronoun that refers to the head noun (although sometimes the relative pronoun is dropped):

(6) the car, [$_{\text{relative clause}}$ **which** John drove __]

Here, the relative clause which refers to the head noun *car*. In a declarative sentence, "the car" would be in the underlined direct object position. Japanese differs from English in two major points: Being a head-final language, a relative clause precedes the head noun; and there is no relative pronoun, so the relative clause looks like there is a gap:

(7) [ジョンが__運転した $_{\text{relative clause}}$] 車
John-NOM __ drive-PAST car

While the exact construction of relative clause is a target of much debate in the field of linguistics, it is beyond the scope of current task to go into it. The important thing is to recognize the difference in word order and the lack of relative pronouns.

Verbs in Japanese. Unlike English, Japanese verbs are not free morphemes. They must be followed by at least one suffix, namely tense, which is either nonpast or past:

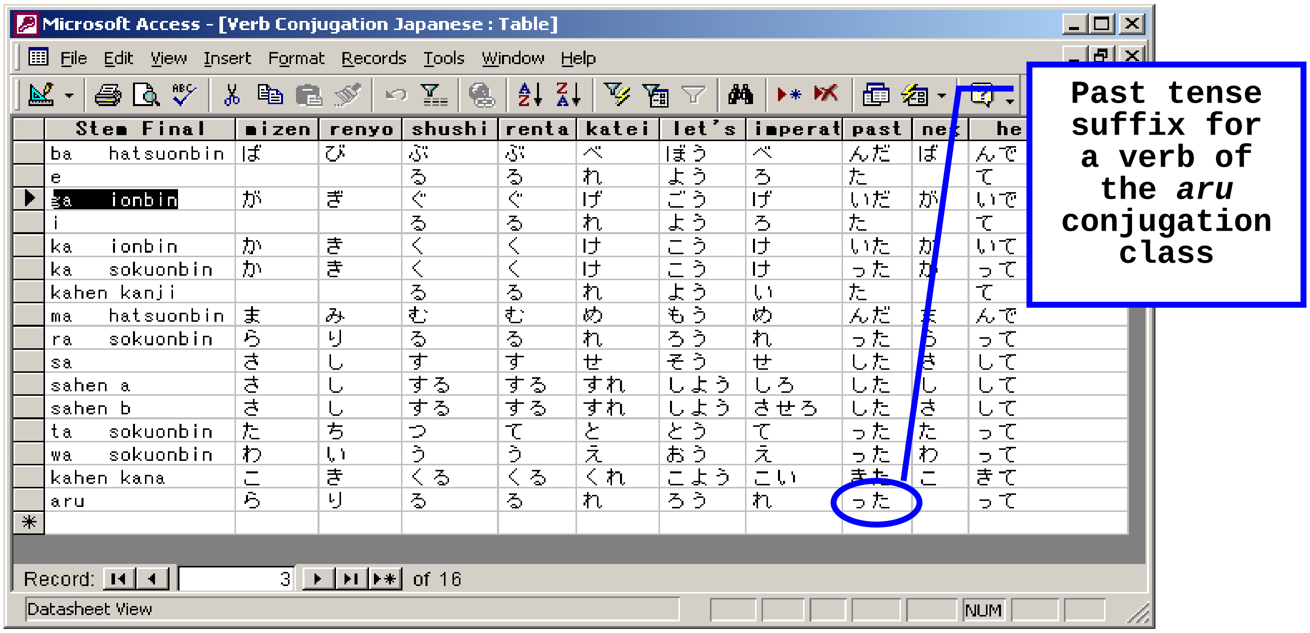
(8) *nom-* "drink":

a. Mizu-o nom-u -Nonpast (present, future, habitual etc.)
water-ACC drink-NONPAST
"[I] drink water."

b. Mizu-o non-da -Past
water-ACC drink-PAST
"[I] drank water."

c. *Mizu-o nom-Ø - No tense suffix (impossible)

Besides tense, Japanese verbs conjugate for various syntactic and/or morphological contexts, such as before negation, before a polite suffix, before a noun, in an imperative sentence, and so on. Traditional Japanese grammar recognizes 6 conjugated forms. For the purpose of adapting ModelCreator for Japanese, we have extended the number to 10, in order to handle a wider and more complete set of verb forms. At the same time, the exact form of each conjugated form depends on the stem-final sound, which can be either a consonant or a vowel. In all, there are 16 such conjugation types. The following figure, Figure 5, presents the Japanese verb conjugation table from the Microsoft Access database in which we stored this morphological information. This table lists the appropriate suffix for a verb form, depending on its conjugation class and the desired inflectional category.

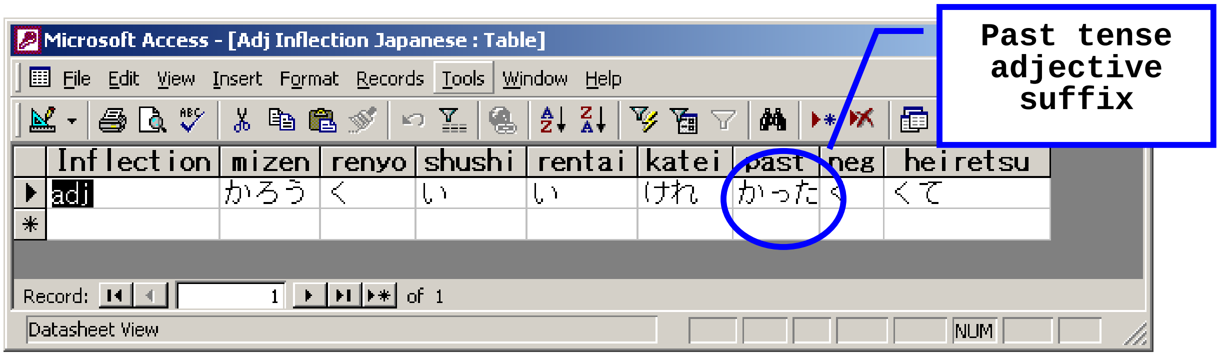

Microsoft Access - [Verb Conjugation Japanese : Table]
File Edit View Insert Format Records Tools Window Help
Record: 3 of 16
Datasheet View
NUM

Stem Final	mizen	renyo	shushi	renta	katei	let's	imperat	past	ne	he
ba hatsuonbin	ば	び	ぶ	ぶ	べ	ぼう	べ	んだ	ば	んで
e			る	る	れ	よう	ろ	た		て
ga ionbin	が	ぎ	ぐ	ぐ	げ	ごう	げ	いだ	が	いで
i			る	る	れ	よう	ろ	た		て
ka ionbin	か	き	く	く	け	こう	け	いた	か	いて
ka sokuonbin	か	き	く	く	け	こう	け	った	か	って
kahen kanji			る	る	れ	よう	い	た		て
ma hatsuonbin	ま	み	む	む	め	もう	め	んだ	ま	んで
ra sokuonbin	ら	り	る	る	れ	ろう	れ	った	ら	って
sa	さ	し	す	す	せ	そう	せ	した	さ	して
sahen a	さ	し	する	する	すれ	しよう	しろ	した	し	して
sahen b	さ	し	する	する	すれ	しよう	させろ	した	さ	して
ta sokuonbin	た	ち	つ	て	と	とう	て	った	た	って
wa sokuonbin	わ	い	う	う	え	おう	え	った	わ	って
kahen kana	こ	き	くる	くる	くれ	こよう	こい	きた	こ	きて
aru	ら	り	る	る	れ	ろう	れ	った		って

Past tense suffix for a verb of the *aru* conjugation class

Figure 5. **Verb conjugation table.**

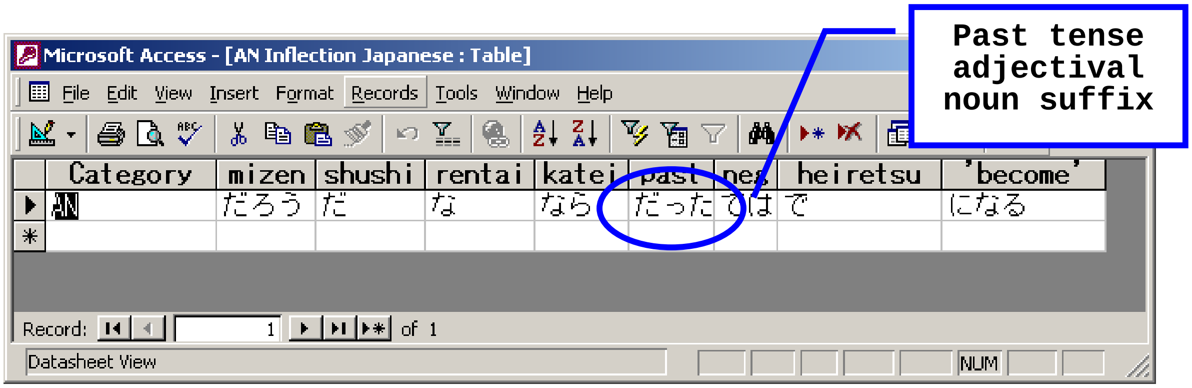
Adjectives and adjectival nouns. It was mentioned above that complete one-to-one correspondence of members of a lexical category is rare between two languages. This is true of lexical categories themselves. Japanese has so-called adjectival nouns (or "nominal adjectives"), whose syntactic distribution resembles that of adjectives, but whose morphological pattern more closely resembles that of nouns. It is necessary to list separate inflectional forms for adjectives and adjectival nouns. The following two figures, Figure 6 and Figure 7, show the adjective inflection table and adjectival noun inflection table, which similarly list the appropriate suffixes for different forms of adjectives and adjectival nouns:


Microsoft Access - [Adj Inflection Japanese : Table]
File Edit View Insert Format Records Tools Window Help
Record: 1 of 1
Datasheet View
NUM

Inflection	mizen	renyo	shushi	rentai	katei	past	neg	heiretsu
adj	かろう	く	い	い	けれ	かった	く	くて

Past tense adjective suffix

Figure 6. **Adjective inflection table.**



Microsoft Access - [AN Inflection Japanese : Table]
File Edit View Insert Format Records Tools Window Help
Record: 1 of 1
Datasheet View
NUM
Past tense adjectival noun suffix

Category	mizen	shushi	rentai	katei	past	neg	heiretsu	'become'
AN	だろう	だ	な	なら	だった	では	で	になる

Figure 7. **Adjectival noun inflection table.**

Modifications to ModelCreator's Lexical Resources

Lexicon. The source for the lexicon was taken from ipadic (Matsumoto et al., 1999), a freeware dictionary developed and distributed by Nara Institute of Science and Technology. Ipadic was originally packaged with several NLP tools for Japanese such as ChaSen[3] (morphological analyzer) or Kakasi (kana-kanji converter). It consists of several EUC-encoded text files, one file for each lexical category. (EUC is a type of encoding for Japanese.) In each file, each line contains a lexical entry and the information that goes with that entry (its category, inflection type, etc.), as well as information related to the organization of the dictionary.

(9) An example entry from *ipadic*:

(品詞(**形容詞 自立**))((見出し語(**あさい 3206**))(読み**アサイ**)(活用型 **形容詞・アウオ段**))

gloss: (category(**adjective free**)) ((entry (あさい 3260)) (pronunciation アサイ) (inflection **adjective auo**))

The letters/characters in bold face indicate lexical information, while those in regular type are organizational information. For the purpose of Multilingual Math ModelCreator, we decided to use Microsoft Access for easy manipulation and organization of large number of entries and UTF-8 for Japanese encoding. Thus, it was necessary to modify *ipadic* by stripping superfluous information, adding occasional supplementary information, and converting as much information as possible into English/ASCII.

The first step was to strip the original dictionary of any information that was unnecessary for ModelCreator. For example, the parentheses and numbers in the original entry needed to be removed, as well as superfluous spaces. This was done using Perl scripts and *Nimax*, a Japanese version of the UNIX text editor *emacs*. A single tab replaces a string of unnecessary information, making the entry more suitable for importing to Microsoft Access. After this stripping, the example entry in

(9) looked like the following:

(10) 形容詞 自立[tab]あさい[tab]アサイ[tab]形容詞・アウオ段

The deleted words such as "category" or "pronunciation" would be replaced by a column label in Microsoft Access.

The second step was to convert category and conjugation/inflection information into English (ASCII), so that those who do not read Japanese can understand the structure of the lexical database. Thus, the entry became:

(11) **adj_free**[tab]あさい[tab]アサイ[tab]**adj**

The information contained in this new entry is, respectively:

(12) **category**[tab]**word entry**[tab]**pronunciation**[tab]**inflection type**

That is, in (11), the category of the word is "adj_free," the word entry is あさい, pronunciation is "アサイ," and the inflection type "adj."

The third step was to convert the pronunciation (underlined アサイ in (10) and (11)) into ASCII transcription. Thus, the final entry became as follows:

(13) **adj_free**[tab]あさい[tab]**asai**[tab]**adj**

Note that the pronunciation entry was originally in *Katakana* as in (10) and (11) (underlined).

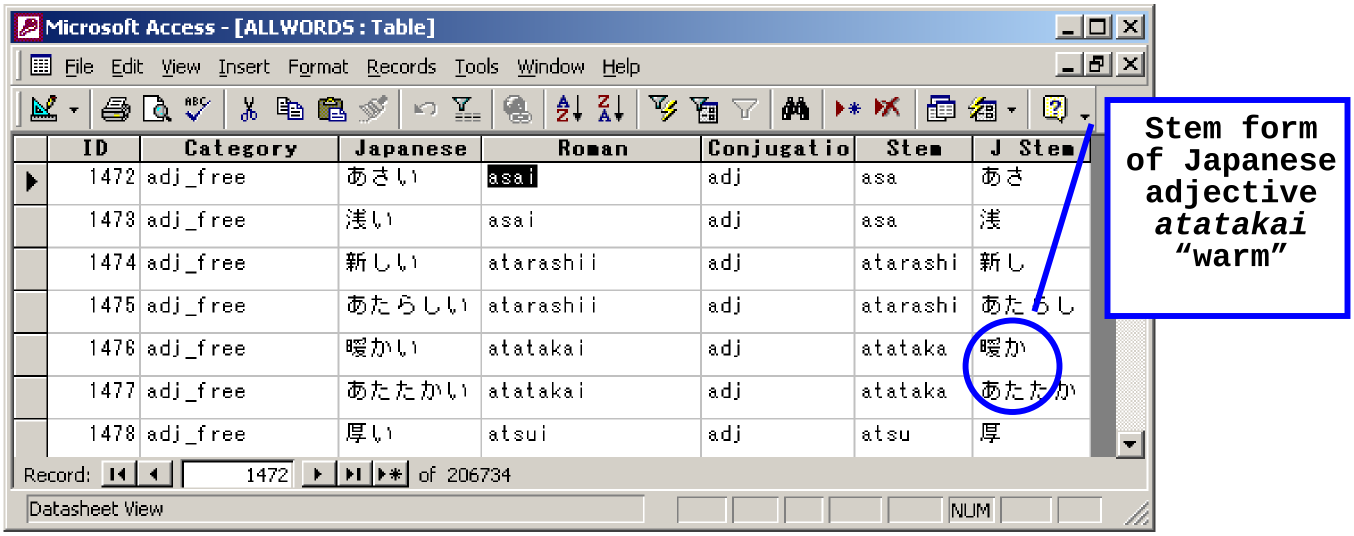
In order to avoid having to manually convert each pronunciation entry into ASCII as in (11) (underlined), a freeware program called *Kakasi*[4] was used. *Kakasi*, a component of the Japanese full-text search engine *Namazu*, is a program that converts between ASCII representation and *Kanji, Kana,* or combinations thereof, and also between *Kana* and *Kanji*. There were well over 200,000 entries, so this was much more efficient than manual conversion, even though there were quite a few errors in conversion due to most *Kanji* having more than one possible pronunciation.

At this point, the text dictionary files were imported into Microsoft Access, so that the query function of the software could be used to further process the dictionary.

The fourth step was to add information about the stem of words that belong to lexical categories that inflect/conjugate, such as adjectives and verbs. Here, the term *stem* is not used in a strict morphological sense, but rather as that part of the written word that does not change throughout conjugation/inflection, which is more appropriate for this project. As is mentioned above, adjectives, adjectival nouns, and verbs have various inflectional forms according to their position in the sentence and the suffixes that follow them (e.g., tense, aspect, negation, etc.). It is possible to list the fully inflected/conjugated forms of each verb/adjective/adjectival noun. However, inflection/conjugation in Japanese is very systematic, so it is easy to simply enter the stem and inflection/conjugation type for each word and attach inflectional/conjugation suffixes to it. The latter is the method we chose for this project.

In the Access database, a column was added for the stem in ASCII (*Stem)* and another for the stem in Japanese (*J stem)*. The entry form of verbs/adjectives/adjectival nouns is a nonpast form. To derive the desired Japanese stem from the entry form only required using a query to identify the correct category, then copying the entry form in the Japanese column into the J Stem column minus the last *kana*. Adjectives and adjectival nouns, fortunately, have only one type of inflection each, thus we encountered no problem. Deriving the stems in ASCII was more involved than for verbs because there are several types of verb conjugation, and moreover, as we have shown above, one *kana* may correspond to one or two segments. There are verbs whose last syllable is onset-less (only the nucleus). This means that simply copying the ASCII stem under "Roman" column minus the last letter or last two letters across the board would not do the trick; rather, some verbs need to have the last letter eliminated to arrive at the stem, others need to have the last two letters eliminated.

When all the stems were derived, all the lexical-category-based dictionaries were concatenated into one large dictionary, a snapshot of which appears below in Figure 8:


Microsoft Access - [ALLWORDS : Table]
File Edit View Insert Format Records Tools Window Help

ID	Category	Japanese	Roman	Conjugatio	Stem	J Stem
1472	adj_free	あさい	asai	adj	asa	あさ
1473	adj_free	浅い	asai	adj	asa	浅
1474	adj_free	新しい	atarashii	adj	atarashi	新し
1475	adj_free	あたらしい	atarashii	adj	atarashi	あたらし
1476	adj_free	暖かい	atatakai	adj	atataka	暖か
1477	adj_free	あたたかい	atatakai	adj	atataka	あたたか
1478	adj_free	厚い	atsui	adj	atsu	厚

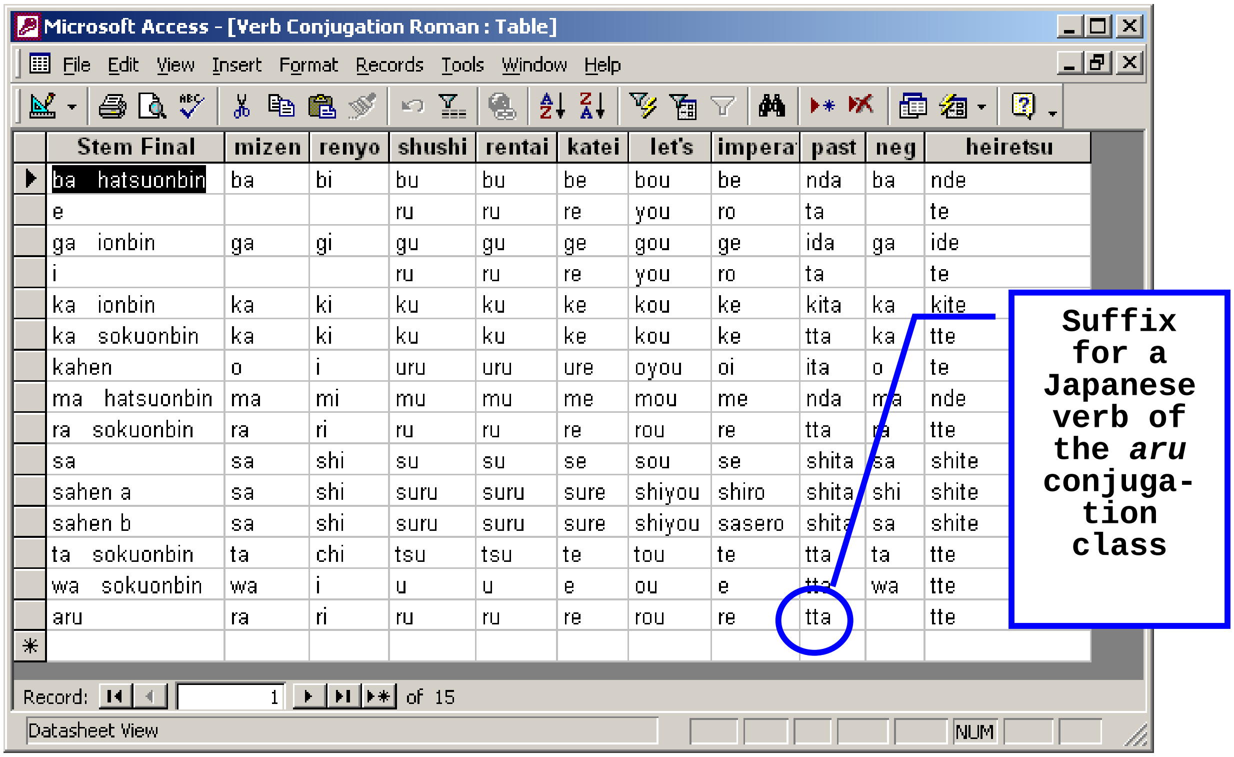
Record: 1472 of 206734

Datasheet View NUM

Stem form of Japanese adjective *atatakai* “warm”

Figure 8. **Finished dictionary.**

Once the main dictionary was finished, we created two conjugation/inflection tables each for verbs, adjectives and adjectival nouns: one in Japanese and one in ASCII. The conjugation/inflection table in Japanese for each category is seen above in Figure 8. The verb conjugation table in ASCII looks like the following in Figure 9:


Microsoft Access - [Verb Conjugation Roman : Table]
File Edit View Insert Format Records Tools Window Help

Stem Final	mizen	renyo	shushi	rentai	katei	let's	impera	past	neg	heiretsu
ba hatsuonbin	ba	bi	bu	bu	be	bou	be	nda	ba	nde
e			ru	ru	re	you	ro	ta		te
ga ionbin	ga	gi	gu	gu	ge	gou	ge	ida	ga	ide
i			ru	ru	re	you	ro	ta		te
ka ionbin	ka	ki	ku	ku	ke	kou	ke	kita	ka	kite
ka sokuonbin	ka	ki	ku	ku	ke	kou	ke	tta	ka	tte
kahen	o	i	uru	uru	ure	oyou	oi	ita	o	te
ma hatsuonbin	ma	mi	mu	mu	me	mou	me	nda	ma	nde
ra sokuonbin	ra	ri	ru	ru	re	rou	re	tta	ra	tte
sa	sa	shi	su	su	se	sou	se	shita	sa	shite
sahen a	sa	shi	suru	suru	sure	shiyou	shiro	shita	shi	shite
sahen b	sa	shi	suru	suru	sure	shiyou	sasero	shita	sa	shite
ta sokuonbin	ta	chi	tsu	tsu	te	tou	te	tta	ta	tte
wa sokuonbin	wa	i	u	u	e	ou	e	tta	wa	tte
aru	ra	ri	ru	ru	re	rou	re	tta		tte

Record: 1 of 15

Datasheet View NUM

Suffix for a Japanese verb of the *aru* conjugation class

Figure 9. **Verb conjugation table in ASCII.**

Frame database. The frame database stores semantic frame information. As is discussed above, it is structured in such a way that a more specific frame can inherit many of its properties from a more generic one. It was necessary not only to "Japanize" the lexical items but also to modify the relationships among some frames to move from English to Japanese

Modification of the frame database required less labor than creating the dictionary. It consisted of:

1. Changing the lexical items to Japanese
2. Suppressing the roles that have no equivalence in Japanese
3. Finding the ID number of each lexical item in the above dictionary and inserting it below each lexical item
4. Changing the inheritance properties of each model where necessary

Tasks 1 and 2 were carried out simultaneously. A role would be suppressed if the syntactic structure for which the role is provided is ungrammatical, either in the language as a whole or in the distance-rate-time (DRT) context. For example, Role 12 lists verbs of transporting passengers, such as *fly*, as in "The pilot flew the passengers to Rome." In Japanese, the direct object of such verbs as *drive* or *fly* has to be the vehicle. Transportation of passengers would be expressed by a participle construction, closer to "The pilot flew an airplane, carrying the passengers." Another example is the passive construction, such as "A truck <u>was driven</u> at 25 mph." A passive does exist in the language, but it is not used in the DRT context. Nominalization of a vehicle-operating event into a noun is also rare in Japanese, if not impossible. For example, the event expressed by "John drove to NYC" may be nominalized as "John's <u>drive</u> to NYC" in English. This is not a common phenomenon in Japanese;[5] the nominalization needs to apply to the whole sentence, thus requiring a completely different construction from English. Thus, the roles corresponding to nominalization were suppressed.

Once Task 1 and 2 were done, Task 3 involved simply going into the dictionary, finding the word, and copying the unique ID into the frame database. Task 4 took some time to finalize. The inheritance properties between frames cannot be the same between the English and the Japanese frame databases, largely because the proper combinations between a vehicle and verbs (whether the vehicle is the subject or the object) are rarely (if ever) the same between any two languages. For example, in English one can drive a car, a bus, a motor boat, or a sleigh; on the other hand, Japanese equivalent of the verb *drive* (*unten-suru*) can be used for the first two, but not for the latter two and their cousins. This is an example in which a Japanese verb has more specialized use than its English counterpart. In order to have the correct vehicle-verb pairing, inheritance between an automobile-type frame and motorboat- or sleigh-type frames needed to be blocked in Japanese and appropriate verb(s) introduced in the sleigh-type frames.

Modifications to ModelCreator's Grammatical Generation Rules

As shown earlier, Japanese syntax differs substantially from English syntax. Clearly, these differences had to be reflected in the course of developing a Japanese version of ModelCreator.

Dominant word-order pattern. First, Japanese is an SOV language with postpositions, whereas in English, verbs and prepositions precede their objects. Implementing this difference was relatively simple, given the ModelCreator architecture, in which largely language-neutral logical structures are used as the basis for generating test items. An eventual verb phrase, such as "drove 40 kilometers" in English or "４０キロメートル運転した" in Japanese will be represented at this logical level by a predicate expression such as DRIVE(DISTANCE=40). When a logical phrase such as this is parsed by the language generation engine, the only syntactic change needed for Japanese (besides, of course, changing the words!) is the directive that the verb realizing the logical element DRIVE should be printed at the end of the phrase, rather than at the beginning. The handling of Japanese postpostions is analogous. When, during the generation process, the conversion from logic demands that a prepositional phrase be output, the Japanese version of the code will output a postposition at the end of the phrase, whereas the English version will output a preposition at the beginning.

Figures 10 and 11 illustrate how the logical structure of a sentence (in fact, an entire item) can be carried over largely unchanged form English to Japanese in many cases, while the realization of the logical relations is language-specific. (The birdcage-like structures below are meant to schematically represent the logical structures that ModelCreator produces.) Disregarding the order of arguments, this logical structure is common to the English and Japanese sentences below. In each language, the information about the travel event is packaged into a biclausal structure (with each clause represented in the diagram as a single nexus at which the lines connected to logical arguments meet—the roof of a birdcage). The embedded clause expresses the vehicle and distance slots of the event, connected by a TRAVEL-class verb. The main clause in each language forms a query regarding the time required for the travel event. (The dashed line connecting the VEHICLE argument to the higher clause indicates that this is a *control* structure [cf. Rosenbaum, 1967; McCawley, 1998; Chomsky, 1981; Chierchia, 1984], in which a single element can function grammatically as an argument of multiple clauses.)

In mapping this logical structure to a finished sentence, though, the Japanese ModelCreator system must respect the SOV word order of Japanese (VEHICLE-DISTANCE-TRAVEL), whereas English ModelCreator selects SVO order (VEHICLE-DISTANCE-TRAVEL).

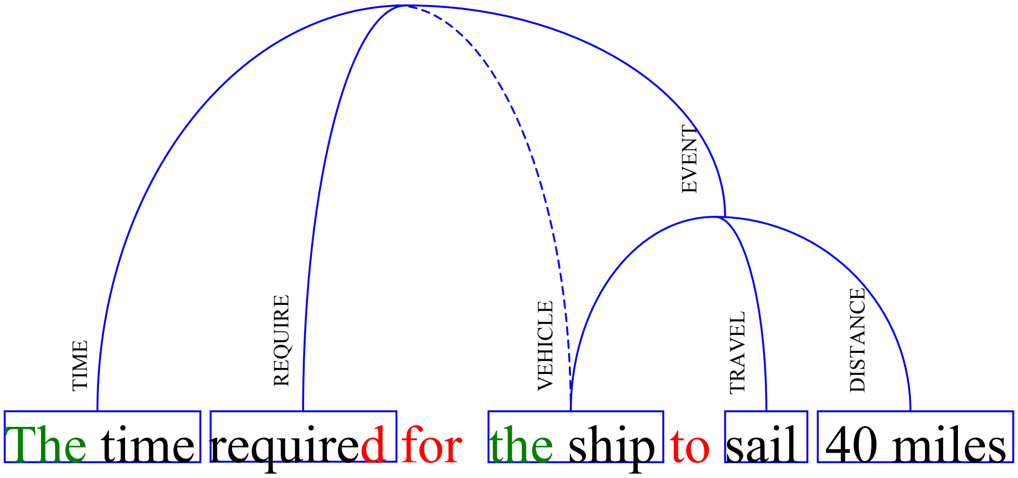


Figure 10. **English mapping, logic to surface.**

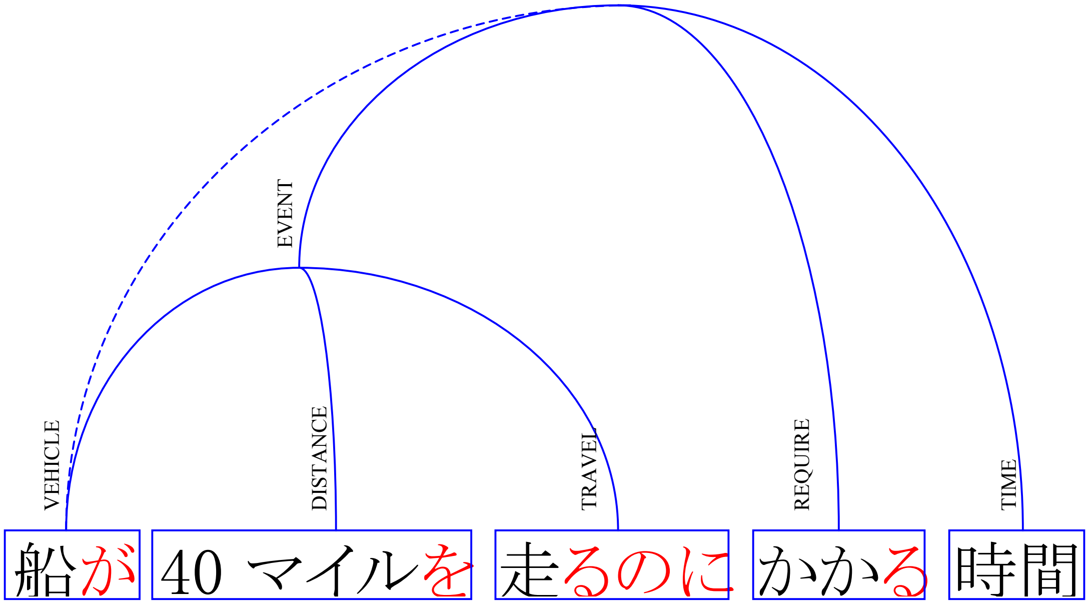


Figure 11. **Japanese mapping, logic to surface.**

Inflection. Second, Japanese differs from English in the inflectional paradigms of nouns and verbs (and in the case of Japanese, verbal nouns and adjectival nouns). Here, the term inflection refers to the use of affixes (suffixes in particular for Japanese and English) to mark syntactic or semantic relationships within a sentence. In the case of English nouns, this comprises the distinction between singular and plural forms, and in the case of English verbs, the marking of tense with suffixes like –ed and –en, and also subject-verb agreement marking in the present tense (e.g., “She walks” vs. “They walk”).

In Japanese, the number of nouns is not morphologically marked, which slightly simplifies the task of natural language generation. However, Japanese does systematically mark nouns for case, so that direct objects receive a different suffix (を) from subjects (which are usually marked with が), and oblique arguments to a verb may be marked with a number of other suffixes. Some internal bookkeeping had to be introduced to handle case marking for Japanese nouns, but fortunately, a good deal of this mechanism already existed in the English version of ModelCreator.[6] In order to assign the proper case marker to a Japanese nominal, we need to keep track of the grammatical function (such as subject, object, or indirect object) assigned to each noun during generation. As it turns out, this information was already present in the English version of the software, to support such linguistic options as the passive voice.

In English, the direct object of a verb is not marked with a case suffix, but it is still useful to identify which argument is the direct object during generation. For one thing, certain verbs have restrictions on what sorts of expressions they can have as a direct object. (*Reach* takes a destination as its object, whereas *drive* can take a vehicle or a distance expression, rate, or perhaps even a time expression.) For another, in producing passive sentences, we have to know which argument of a verb would be the direct object in an active sentence, so that we can suppress it, and assign it the subject role.

While the Japanese verbal paradigm differs from the English one, these differences turn out not to demand substantial changes in the grammatical rules used for generating Japanese. For both Japanese and English, the only tense distinction made by ModelCreator is between one tense used for past events and another used for ongoing action (past and present tense for English and past and nonpast for Japanese). Japanese has no subject-verb agreement, so again in this case Japanese inflection turns out to be simpler for our system to handle than in English inflection.

Discourse marking. Third, the two languages differ in how the discourse status of entities mentioned in an item is expressed. In English, the definiteness of noun phrases is primarily indicated through the choice of determiner, so ModelCreator uses rules concerning which entities are mentioned in each part of the item in order to determine whether a noun such as *ship* should be output as *a ship*, *a certain ship*, or *the ship*, for example. In Japanese, however, definiteness is not marked by means of a determiner, although under certain conditions related to the discourse status of entities named, the topic marker は is used instead of the subject marker が. Since these conditions do not coincide with those for any of the English determiner choices, ModelCreator's discourse module had to be largely replaced for Japanese.

Differences relating to logical structure. Fourth, while the pseudological representation we produce as an intermediate step toward generating complete test items is intended to be largely language-independent, some cases have arisen in which it seems that even this level must differ for English and Japanese. For instance, the English system uses a subordinate clause in query expressions such as "the speed maintained by a truck traveling 4 kilometers in 2 hours," which is generated from a "biclausal" logical representation that contains two separate propositions—one headed by a predicate to eventually be realized as the verb *maintain* and another headed by a predicate that will be expressed as *travel*. In Japanese, however, this content is best expressed using a single clause, without introducing a higher verb such as *maintain* in English.

In addition, English and Japanese sometimes differ in the order in which arguments to a predicate are expressed such as, whether the distance expression or time expression comes first in a verb phrase. Since the order of arguments is specified at our logical level, it is difficult to adequately cover the linguistic facts without changing the logic for Japanese. On the whole, though, the modifications to the logical level for Japanese were quite restricted in scope, so that this level can still be regarded as quite general.

Special cases. Finally, as is common in linguistics, once we have done our best to describe the domain in question using general grammatical rules, there remains a residue of idiosyncratic cases that must be handled in an ad hoc fashion. As it happens, Japanese prefers for a distance expression to precede a time expression in a verb phrase when the subject of the clause is a vehicle. However, when the vehicle is the object, and the subject is a human agent (the driver), the time precedes the distance. No general grammatical principle suggests itself to capture facts like these, so in developing Japanese ModelCreator, they simply had to be encoded as such.

Of course, this sort of arbitrariness is to be found in English grammar as well. We use plural agreement with the quantifier *all* (e.g., “All cats hate water”), but singular agreement with *each* (“Each cat hates water”). If this sort of idiosyncrasy were the rule, rather than the exception, the general architecture of ModelCreator would be of little value; fortunately, this is not the case.

Interface Modifications

In addition to these substantive changes to the underlying item generation system, a number of modifications to the user interface for the program were necessary in porting ModelCreator’s functionality to Japanese.

Unicode. First, the set of characters used in writing Japanese is much larger than the English alphabet. As mentioned above in the general discussion of Japanese grammar, Japanese uses two syllabaries (*Hiragana* and *Katakana*), as well as an ideographic set of characters borrowed from Chinese (*Kanji*). To present items to the user in Japanese, we thus needed to Unicode-enable the ModelCreator application, which involved rewriting the text-processing routines of the program to handle multiple-byte characters and adding Unicode support to the display engine.

Line breaking. Second, whereas English text conventionally separates words with spaces, a Japanese sentence typically does not include any explicit delimitation of words. To accommodate this option for displaying Japanese, it was necessary to implement some smart text handling behind the scenes, so that word boundaries would be respected for line breaking, but not explicitly marked in the finished item.

Interface parameters. Third, some of the generation parameters available for English are inappropriate for Japanese, given the different structure of the two languages. In such cases, the parameter was simply dropped from the Japanese version of ModelCreator or modified to behave in a way that makes more sense for Japanese. One example of a parameter that had to be modified for the Japanese version of the program is that of sentence structure. In the English version of ModelCreator, there are four possible options for the basic structure of a clause used to express a travel event:

- Active voice, agent and vehicle expressed (e.g., “Claude drove a van 50 miles.”)

- Active voice, only vehicle expressed (e.g., “A van traveled 50 miles.”)
- Active voice, only agent expressed (e.g., “Claude drove 50 miles.”)
- Passive voice, only vehicle expressed (e.g., “A van was driven 50 miles.”)

In the Japanese version of the program, the final two options above have been removed. The passive structure in Japanese is not as natural for this context as it is in English, and we decided therefore to disable it entirely. The third option, in which the vehicle type is suppressed, is also unacceptable in Japanese. Note that there is a strong dependency between the vehicle type and the verb chosen in the sentence. (“Claude drove to Phoenix” implies a land vehicle, whereas “Claude flew to Phoenix” assumes an aircraft.) In Japanese, this dependency cannot comfortably be left implicit without mentioning the vehicle.

Other generation options which needed to be modified in the Japanese version of the program include the use of event nouns in event descriptions (e.g., “Stacy made a trip of 100 miles” rather than “Stacy traveled 100 miles”) and the periphrastic statement of speeds in the form “a speed of 100 miles every 2 hours.” The linguistic resources that are used to express a given content will vary to some extent from language to language, so we need to be circumspect in deciding which generation options to allow for each new language. However, languages are likely to agree on a large number of these possible options, so that much of the work of developing the English version of the ModelCreator system could be carried over directly to Japanese. For instance, English and Japanese both permit travel events to be expressed in active sentences with the agent expressed or left implicit, as shown above. Both languages allow these sentences to appear in multiple tenses. Both languages allow times to be expressed as an absolute duration (e.g., 5 hours) or as a sequence of clock times (e.g., from 1:00 to 6:00). Both allow rates to be explicitly qualified as average speeds or constant speeds.

Default parameters. Finally, the change from English to Japanese as the target language demanded a rethinking of the default generation parameters for the program. In particular, the default units for distance and rate in English were set to miles and miles per hour, respectively. In Japanese, these were set to kilometers and kilometers per hour, respectively.

In English, the default structure for describing a travel event is the passive (e.g., “A truck is driven for 5 miles at 60 miles per hour.”). In Japanese generation, though, this structure is not available, as noted above, so the default structure uses the active voice.

Insights gained from the modifications. Because we started out with the English version of ModelCreator, there were quite a few cases where we had to translate from English to Japanese. Despite the fact that ModelCreator's architecture is fairly language-neutral, this still means that the structure of the English language was sometimes imposed on Japanese, making some of the Japanese output less than optimal and causing some linguistic content to remain uncovered. These are possible clues to where the language-neutrality of the ModelCreator can be further improved.

Summary

All in all, the modifications to create a working version of the ModelCreator software for Japanese were relatively minor, given the vast difference in grammatical structure between Japanese and English. The most comprehensive adaptations were in the areas of lexical resources for Japanese (such as a dictionary of names, a table of inflectional suffixes, and the Japanese frame database), the language-specific generation rules that flesh out ModelCreator's logical structures, and finally, the user interface for the program. While this process was far from trivial, it was much more straightforward than any strategy for producing a Japanese version of a template-based NLG program could be. A template-based system would require the manual translation of each template, and grammatical and semantic restrictions on the filler elements for each template slot would have to be devised and stated anew for the new target language.

Modifications for Spanish

In this section, we describe the changes made to ModelCreator in the course of adding Spanish functionality. Compared to the changes needed for Japanese, these were relatively few in number and small in scope. Of course, Spanish is a language much closer to English, historically and typologically, so it is not surprising that more of the functionality developed for English could be carried over for Spanish in ModelCreator.

As with Japanese, the major categories of modifications that we made to the basic program for Spanish involved the lexical resources we used (namely, the dictionary and frame database) and changes to the grammatical generation rules. Before discussing these modifications in detail, however, we provide a short survey of the grammatical characteristics of Spanish, highlighting differences from English that are relevant for this generation project.

General Grammatical Structure

English and Spanish are both Indo-European languages, which puts them together in a broad class containing such other languages as Russian, Sanskrit, and Farsi. However, Spanish and English are even more closely related typologically than their genealogy would suggest. Spanish belongs to the Romance subfamily of Indo-European, and while English is a Germanic language, it has been heavily influenced by Romance through extended contact with French during the Norman invasion of England.

Both Spanish and English are SVO languages, in which the subject of a transitive clause precedes the verb, while the object follows it. Both languages have prepositions rather than postpositions. The largest difference in word order between the two languages is the fact that adjectives in Spanish typically follow the noun, whereas in English, adjectives precede it.

Both languages exhibit subject-verb agreement (though the English paradigm is less extensive), and neither has overt case marking of nominals. However, Spanish nouns have inherent gender (masculine or feminine), which is not a feature of English.

In short, Spanish is a much more similar language to English than Japanese, which immensely simplified the task of accommodating it in the ModelCreator interface.

Modifications to Lexical Resources

Lexicon. The Spanish lexicon is derived from the MULTEXT Lexicons version 1.0, distributed by European Language Resources Association (ELRA/ELDA). It contains word forms for five European languages: English, French, German, Italian, and Spanish. Among these languages, Spanish has the greatest number of word forms listed (510,710 word forms), in part due to the fairly extensive morphological system of the language. (The full paradigm is explicitly provided for each entry.) The word forms for a given language are contained in a single text file, in which each line contains a word form, its lemma, and a lexical tag in a tabulated format. Figure 12 presents a few example lines from the paradigm of the verb *recorrer* (travel). Without going into too much detail, the codes for the first line indicate that *recorriéramos* is a main verb (**Vm**) in the subjunctive (**s**) mood and imperfect (**i**) tense, with first-person plural (**1p**) agreement inflection.

recorriéramos	recorrer	Vmsi1p-
recorriésemos	recorrer	Vmsi1p-
recorrió	recorrer	Vmis3s-
recorro	recorrer	Vmip1s-

***Figure 12.* Example of Spanish entries from MULTEXT Lexicons version 1.0.**

Because the file is already in ASCII characters, it did not need to go through the same clean-up/conversion process used for the Japanese source.

Frame database. The original English frame database was modified to work in Spanish, as in the case of Japanese. However, not all of the four steps used in converting the Japanese database, repeated below, were necessary for Spanish:

1. Changing the lexical items to Japanese
2. Suppressing the roles that have no equivalence in Japanese
3. Finding the ID number of each lexical item in the above dictionary and inserting it below each lexical item
4. Changing the inheritance properties of each model where necessary

Specifically, Step 3 was not necessary, because Spanish does not use two-byte characters. Instead, the orthographic forms of each lexical item could simply be listed in the file.

As members of the European branch of Indo-European language family, overall grammatical difference between Spanish and English is not as great as between English and Japanese. This meant fewer changes in frame database than for Japanese. They mostly consisted of: (a) the lack of nominalization of DRT-type verbs such as *fly – flight*, *arrive – arrival,* or *depart – departure*; and (b) the lack of some of the vehicle-specific verbs, such as *John bicycled to Rome*. The Spanish counterpart of (b) is a verb corresponding to *go, travel,* and so on, plus instrumental PP, "*en* ('by') vehicle," a construction that is quite common in English as well. This necessitated an additional role for the verb in this construction and changing the inheritance properties of frames. Difference (a) above necessitated Step 2, suppressing the roles corresponding to nominalizations that have no appropriate counterpart in Spanish.

Once the initial modifications were made, which consisted primarily of changing the vocabulary, a native speaker was consulted as to the appropriateness of the Spanish words and some basic sentence constructions. Afterwards, necessary changes were made as the grammatical generation rules became developed, in consultation with the native speaker.

Modifications to Grammatical Generation Rules

Because of the grammatical differences, the rules used to generate the finished text for the Spanish version of ModelCreator had to be modified somewhat from those used for English.

Word order. The dominant word-order pattern is SVO, as in English. However, English and Spanish word order does not correspond in all respects. Most notably, Spanish differs in that adjectives typically follow the nouns that they modify, whereas in English, adjectives precede the noun. In fact, matters are not entirely so simple, since certain adjectives in Spanish precede the noun (typically those which describe inherent qualities of an entity).

For this and other minor word order differences between English and Spanish, accommodation had to be made within the generation rules.

Inflection. The biggest grammatical difference between English and Spanish for the purposes of natural language generation is the area of verbal and nominal inflection. Nouns in Spanish have inherent gender (masculine or feminine), which must be kept track of in the course of generating a sentence. Adjectives agree in gender and number with the nouns that they modify, and verbs agree in gender and number with the subject noun.

The necessity of associating each noun with its gender is addressed by means of the Spanish dictionary described above. Whereas in the English version of ModelCreator, the dictionary was only used for person names, Spanish ModelCreator makes more extensive use of lexical resources because of the more complex inflectional information used for each word.

Miscellaneous differences. Finally, there are a number of minor details in wording between Spanish and English that needed to be addressed in the transition. For example, the phrasing of time expressions is somewhat different in Spanish: *three o'clock* translates as *las 3:00*, using a definite article. Ultimately, an exhaustive review of the content generated by the system is the only way to be sure that no aspect of grammar has been overlooked in the transition.

Interface Modifications

In regard to the interface as well, adding Spanish required fewer changes than adding Japanese. Spanish and English both use the Latin alphabet, and although Spanish does have some accented characters not used for English, these do not require a change in the codepage used to represent text. Therefore, there was no analogue to the difficulty encountered with Japanese, in response to which Unicode support was added.

In fact, the only changes we needed to make to the interface in adding Spanish support fall into the category of *default parameters*, as discussed above for the Japanese integration. A few of the default generation options offered for English had to be changed for the Spanish version due to cultural or linguistic differences. For example, the choice of units for distance was changed from miles to kilometers (as in Japanese), and the voice used in generation was changed from passive to active. (Unlike in Japanese, however, the Spanish version does allow the generation of passive-voice items).

Ensuring the Quality and Validity of Multilingual Generation

In implementing any automatic item creation system, it is essential to have some way of ensuring the quality of the items produced. This in turn requires attention to the use of the system, especially whether it is being used to produce items for a specific language or whether the system is being used to produce items simultaneously in various languages. The first use is not different from item modeling. In this case, the goal is for all items produced by an item model to be not only acceptable in terms of content and psychometric characteristics but also to function equivalently as part of some assessment (see e.g., Bejar et al., 2002). That is, we need to be sure that the language produced for each item is appropriate and that the validity of the measurement construct is not compromised in any of the item instances produced. When the system is used to produce items in a single language other than English, a similar process would be required.

However, the use of the system to produce items simultaneously in various languages requires a far more extensive process. Typical applications of multilingual test development such as international surveys of adult literacy (e.g., Murray, Kirsh, & Jenkins, 1998) or international surveys of educational achievement (e.g., Porter & Gamoran, 2002) aim to develop assessments that yield scores with an equivalent interpretation regardless of the language in which they are administered. This introduces constraints on our items in addition to the requirement of validity for the target language group. Since we are producing items in multiple languages from essentially the same logical representations, potentially with the aim of producing the same items on different language forms of an assessment, we must in addition ensure that the linguistic expression of an item does not vary too widely or inappropriately across languages. Otherwise, the item's difficulty could vary across languages in ways unrelated to the construct.

At present, we address this issue by having a broad sample of the program's output examined by native speakers of the target languages (Japanese or Spanish) and by monitoring the grammatical expressions used for each construction, in order to ensure that they do not differ from the English version of each construction in grammatical complexity. Of course, more thorough evaluation of the validity of our items is possible. We could, for instance, study the cognitive processing of students from different language groups who are presented with automatically generated parallel items from different languages. At present, we have no plans to pursue this, but the option may become more attractive once ModelCreator begins to be implemented for practical item creation tasks.

Ultimately, the proper balance between the ideal expression of an item in a single language and the fulfillment of cross-linguistic constraints on item difficulty will depend on the type of assessment under construction. ModelCreator should facilitate the process but in no way solves the significant challenges of test adaptation. However, to the extent that test adaptation principles are captured in the item models and the linguistic constraints within ModelCreator, it should be an asset over time for the design of assessments in several languages.

Conclusion

The ModelCreator item generation system is a flexible, easily extensible program, which currently can create quantitative comparison math problems in English, Spanish, and Japanese. The multilingual adaptation of the system has been considerably aided by the general architecture of the program, in which generation is broken down into an initial step of creating a logical representation of the item, followed by a language-specific generation step in which this logic is fleshed out.

The system is still under development, and the range of item types covered in all three languages is slated to expand. In future work, we hope to add additional languages to ModelCreator's repertoire and to simplify the task of adding content types to the system. Finally, the challenge of evaluation lies just over the horizon for ModelCreator. Once the system begins to be applied to actual item creation tasks, we will be better able to assess to what extent it serves as an aid to test developers and how successful it is in generating usable items.

References

Baker, C. F., Fillmore, C. J., & Lowe, J. B. (1998). The Berkeley FrameNet Project. In *Proceedings of the 17th International Conference on Computational Linguistics:* Vol 1 (pp. 86-90.). Morristown, NJ: Association for Computational Linguistics.

Bejar, I. I., Fairon, C., & Williamson, D. M. (2002). Multilingual item modeling as a mechanism for test adaptation: Applications to open ended and discrete items. In D. Bartram (Chair), *Item & test generation.* Symposium conducted at the International Conference on Computer-Based Testing and the Internet: Building Guidelines for Best Practice (ITC Conference 2002), Winchester, England.

Bejar, I. I., Lawless, R., Morley, M., Wagner, M., Bennett, R., & Revuelta, J. (2002). *A feasibility study of on-the-fly item generation in adaptive testing* (ETS RR-02-23). Princeton, NJ: ETS.

Bennett, R. E. (in press). *Automatic item generation: An overview*. Princeton, NJ: ETS.

Chierchia, G. (1984). Anaphoric properties of infinitives and gerunds. In M. Cobler, S. MacKaye, & M. Wescoat (Eds.), *Proceedings of the Third West Coast Conference on Formal Linguistics* (pp. 28–39). Stanford, CA: Stanford Linguistics Association.

Chomsky, N. (1981). *Lectures on government and binding*. Dordrecht, the Netherlands: Foris.

Comrie, B. (1987). *The world's major languages*. New York: Oxford University Press.

Deane, P., & Sheehan, K. (2003). *Automatic item generation via frame semantics: Natural language generation of math word problems.* Retrieved February 14, 2005, from the ETS Web site: http://www.ets.org/research/dload/ncme03-deane.pdf

Fairon, C., & Williamson, D. (2002). Automatic item text generation in educational assessment. In *Proceedings of TALN 2002* (pp. 395–401).

Fillmore, C. J. (1968). The case for case. In E. Bach and R. T. Harms (Eds.), *Universals in linguistic theory* (pp. 1–88). New York: Holt, Rinehart and Winston.

Fillmore, C. J. (1985). Frames and the semantics of understanding. *Quaderni di Semantica*, *6*(2), 222–254.

Fillmore, C. K., & Baker, C. F. (2001, June). Frame semantics for text understanding. In *Proceedings of the WordNet and Other Lexical Resources Workshop*, *Pittsburgh, PA.*

Greenberg, J. H. (1966). Some universals of grammar with particular reference to the order of meaningful elements. In J. H. Greenberg (Ed.) *Universals of language* (pp.73–113). Cambridge, MA: MIT Press.

Hambleton, R. K., & Patsula, L. (1999). Increasing the validity of adapted tests: Myths to be avoided and guidelines for improving test adaptation practices. *Journal of Applied Testing Technology*, *1*, 1–12.

Hambleton, R. K., & Kanjee, A. (1995). Increasing the validity of cross-cultural assessments: Use of improved methods for test adaptations. *European Journal of Psychological Assessment*, *11*, 147–157.

Irvine, S. H., & Kyllonen, P. (Eds.). (2002). *Item generation for test development*. Mahwah, NJ: Lawrence Earlbaum Associates, Inc.

LaDuca, A., Staples, W. I., Templeton, B., & Holzman, G. B. (1986). Item modeling procedure for constructing content-equivalent multiple-choice questions. *Medical Education*, *20*(1), 53–56.

Matsumoto, Y., Kitauchi, A., Yamashita, T., Hirano, Y., Matsuda, H., & Asahara, M. (1999). *Japanese morphological analysis system ChaSen version 2.0 manual* (2nd ed.; Technical Rep. No. NAIST-IS-TR99009)., Kansai Science City, Japan: Nara Institute of Science and Technology.

McCawley, J. (1998). *The syntactic phenomena of English* (2nd ed.). Chicago: University of Chicago Press.

Murray, T. S., Kirsh, I. S., & Jenkins, L. B. (1998). *Adult literacy in OECD countries: Technical report on the First International Adult Literacy Survey* (NCES 98-053). Washington DC: U.S. Department of Education. National Center for Education Statistics.

Porter, A. C., & Gamoran, A. (Eds.). (2002). Methodological advances in cross-national surveys of educational achievement. Washington, DC: National Academy Press.

Rosenbaum, P. S. (1967). *The grammar of English predicate complement constructions*. MIT Press, Cambridge, MA.

Singley, M. K., & Bennett, R. E. (2002). Item generation and beyond: Applications of schema theory to mathematics assessment. In S. Irvine & P. Kyllonen (Eds.), *Item generation for test development*. Mahwah, NJ: Lawrence Erlbaum Associates.

Sireci, S. G., & Berberoğlu, G. (2000). Using bilingual respondents to evaluate translated-adapted items. *Applied Measurement in Education*, *13*(3), 229–248.

Stansfield, C. W. (2003). Test translation and adaptation in public education in the USA. *Language Testing, 20*(2), 189–207.

Notes

[1] The pronunciation of the *Kanji* employed in Japan is not quite this straightforward. The system represented in (2a) works mostly for *On-yomi*, a pronunciation of a *Kanji* that came into Japanese when a word was borrowed from Chinese and then modified to fit the Japanese phonological/phonetic system. There may be more than one *On-yomi* associated with one *Kanji*, due to various reasons (different loan words from different dialects of Chinese; different loan words at different time periods; different loan words in different parts of Japan, etc.). There is another kind of pronunciation of *Kanji*, called *Kun-yomi*. This came about when a borrowed *Kanji* was matched with existing Japanese words based on (sometimes rough) meaning correspondence. There may be may be more than one *Kun-yomi* associated with one *Kanji*. Combined, some *Kanji* have three or more pronunciations associated with them.

[2] The exact details of when *–wa* marks subjects as opposed to *–ga* are focus of much debate and are also beyond the scope of this report.

[3] The following is the Web site for *ChaSen*: http://chasen.aist-nara.ac.jp/hiki/ChaSen/

[4] *Kakasi* was obtained from: http://kakasi.namazu.org/

[5] Japanese does have some nouns corresponding to this type of event, such as *hikou* ("flight") or *koukai* ("sea voyage"); there are also English loan words such as *drive* or *cycling*. However, their meaning is often more specialized; for example, *cycling* is not for general bicycling activity but refers to more sports- or leisure-oriented type of bicycling.

[6] Given that this new mechanism for case marking had to be introduced for Japanese, one might question how language-independent the underlying linguistic generation system truly is. We concede that there are attested linguistic phenomena that the current system is not set up to handle (say, a Russian-style case marking system in which the case suffix of a noun may depend on whether it refers to an animate entity). However, the range of constructions that the current generation system *can* implement with relatively little language-specific adaptation is sufficiently large that it constitutes a great benefit in doing multilingual item generation. Moreover, as we increase the set of languages in ModelCreator's repertoire, so also will its coverage of linguistic phenomena increase.