PROVIDING EFFECTIVE FEEDBACK ON WHOLE-PHRASE INPUT IN COMPUTER-ASSISTED LANGUAGE LEARNING

Alison M. L. Fowler

Providing effective feedback on whole-phrase input in computer-assisted language learning

Alison M. L. Fowler
Computing Department, University of Kent, UK
A.M.L.Fowler@kent.ac.uk

Abstract

CALL systems which allow whole-phrase input are still in the minority and those which do rely either on parsing for feedback provision (a solution which is ineffective when input is poor) or simply process input token by token (an unsatisfactory method if input is incorrectly ordered). Since poor input and incorrect word-order occur frequently in CALL responses a different approach may be beneficial.

The LISC system, developed at the University of Kent to present phrase-translation exercises, uses an error-detection and feedback mechanism based on fine-granularity sequence comparison. It compares input phrases to acceptable (correct) answers, but unlike traditional error-checking routines it does not fail on encountering unordered or ungrammatical input. An added advantage of its error detection method is that the system is language independent.

Introduction

Despite the wealth of software available for computer-assisted language learning, surprisingly little effort has been directed towards defining patterns of best-practice in feedback (Bangs, 2003). Many types of CALL exercises exist, but by far the most common (still) are closed activities built around a single predictable answer (Desmet, 2005). Multiple choice and fill-the-gap tasks abound, as do matching and drag-and-drop style exercises – none of which require much text-based input. There is a real need to develop feedback mechanisms which will allow authors of CALL material to design exercises not limited to these basic answer-formats (L'Haire & Vandeventer Faltin, 2003). CALL applications, as any other form of CAA, need to be able to assess what the student knows rather than what is easy to evaluate (Ashton & Thomas, 2006).

Whole-phrase input has an important role to play in second language acquisition (SLA). The ability to form correct sentences in a new language comes not only from being able to translate and inflect words on an individual basis, but also group words according to strict positioning rules. However the number of errors that can occur with this type of input is so great that effective feedback is very difficult to produce.

This paper describes the web-based LISC (Language Independent Sequence Comparison) CALL system and its use on an ab-initio Spanish course at the University of Kent. The online exercises used on the course are based on sentence translation and thus require whole-phrase input. The error detection mechanism uses fine-granularity sequence comparison to locate errors in input and mark them up in a natural and easily accessible way.

The importance of feedback

The latter decades of the 20th century were characterised by an approach to language learning dominated by CLT (Communicative Language Teaching) based on a belief that focus on grammatical form and consequent error correction was ineffective and possibly even had a negative effect on uptake of a second language (Truscott, 1996). However in the late 1990s researchers began to find shortcomings with the CLT approach – in part because its lack of specific grammatical tuition left learners having to generalise grammar rules from communicative expression (Celce-Murcia, 1997).

More recent studies have shown that corrective feedback in SLA is considered beneficial to the learning process by both students and teachers. James (1999) stressed the importance in SLA of raising learners' consciousness of the discrepancies between their current state of knowledge and their goal state. Schulz (2001) found that students felt cheated if their written work was not corrected on submission and in a study by Hyland (2003) students firmly believed that repeated error correction and feedback would be beneficial to them and that without it they would not improve. Dodigovic (2005) concluded that making learners aware of their errors helps them to notice important linguistic features in the language they are trying to acquire.

Exercises which incorporate whole-phrase input can provide an authentic test of language skills and an effective learning experience, as long as they feature effective feedback mechanisms (if not they become intensely frustrating). Some practitioners have said that it is important not to overwhelm users with feedback. Schwind (1990) went as far as to say that exercises which allow multiple errors to occur should be avoided. One way round this problem is to address errors one at a time, preferably presenting them in order of system-determined priority (Heift, 2003). It is certainly true that learners are more likely to attend to feedback if it is concise and precise (Van der Linden, 1993), but this does not mean multiple errors cannot be addressed successfully simultaneously.

Why is feedback on whole-phrase input problematic?

Although many robust computational grammars exist and are utilised successfully in applications such as machine translation, these are not suitable for error-detection because they assume the language they will be processing is error free (Tschichold, 2003). Parser-based CALL systems can deliver appropriate context-sensitive feedback automatically, but to do so they require over-generating rule systems which have to allow for all possible

erroneous phenomena (Menzel & Schröder, 1999) and this is exceedingly hard to achieve. Such systems need limited linguistic domains to succeed (Schulze, 1999) and can be very inefficient due to the sheer number of rules required. This in turn leads to slow answer processing which is frustrating for users (Hémard, 2003).

In some systems all feedback is manually encoded – this is very costly for any reasonable-sized program (Bangs, 2003). If input errors are unanticipated, parser-based systems may be unable to give appropriate feedback (Delmonte, 2003). They may also not recognise – or mark up inappropriately - answers which are correct but phrased in an unusual way. Since users are accustomed to accepting error mark-up as accurate, this can lead them to try to amend answers which are already error-free (Karlström et al, 2007).

Gauging answer-appropriateness (from a semantic point of view) is another tough issue, which often requires a separate stage of processing. Systems which use techniques such as Latent Semantic Analysis can be fooled by using the expected keywords in any order (Guest & Brown, 2007) and pattern-matching systems will simply fail if the input is not structured as anticipated. These methods are not suitable for automatically marked coursework, unless the exercises are designed to control learner input so as to exclude the possibility of unexpected correct forms (Bailey & Meurers, 2007).

Non-parser based systems tend to process user answers word by word (or even character by character), comparing input with expected solutions. Established sequence comparison algorithms such as Levenshtein distance (Levenshtein, 1966) and its various derivatives perform well in areas where input strings are not re-ordered (speech and hand writing recognition, molecular homology testing, gas chromatogram comparison etc.). Levenshtein distance finds the smallest number of insertions, deletions and substitutions needed to turn one string of tokens into another; however one of its basic principles is the preservation of the existing order of elements in the sequences to be compared. Lowrance and Wagner (1975) introduced an extension which coped with transpositions - but their algorithm only handles the interchange of adjacent elements within a sequence and thus does not go far enough for whole-phrase input CALL exercises (it should be noted that these algorithms were never meant to handle this sort of problem).

Why is sequence comparison with element re-ordering particularly demanding?

Two main issues arise with the use of sequence comparison for error-detection in CALL. The first is the *computational complexity* involved (i.e. the amount of resources required for the execution of the sequence comparison algorithm). This is exceedingly problematic when re-ordering is permitted. The second is that with any reasonably complex example there are often many different possible alignments (ways of pairing up the various elements in the two sequences), some of which may have the same edit distance (cost). Comparing the string ‘A B C B C’ to a second string ‘A B C’ will

produce several different mark-ups - each with the same cost - for turning the second string into the first (in all three cases insertion of two items):

```
A B C _ _    (insert 'B C')
A B _ _ C    (insert 'C B')
A _ _ B C    (insert 'B C')
```

Where the cost is equal, there is no way to determine which is the best mark-up, and in a natural language context this can lead to an algorithm choosing as optimal a mark-up which seems unnatural to a human user (Nesbit & Nakayama, 1990).

By way of example, if asked to translate the source phrase:

```
He smiled when the girl opened the door
```

into German (as shown):

```
Er lächelte, als das Mädchen die Tür öffnete
```

an English speaker might, understandably, assume that the word for girl (`Mädchen`) required a feminine article and the word for door (`Tür`) required a neuter article (whereas the opposite is true) and produce:

```
Er lächelte, als die Mädchen das Tür öffnete
```

Despite the fact that swapping the two articles will produce a correct answer, the mark-up in this case should indicate that there are two *incorrect* items, *not* two positioning errors. If using sequence comparison to detect and mark up errors, it is necessary to ensure that the mark-up reflects as accurately as possible the type of each error. With reference to linguistic information this would be easy, but using sequence comparison alone there are no such clues upon which to base the decision.

The above is a rather simple example. Where a user is asked to translate a phrase into a language which features unfamiliar inflection, differing word order and lexical constructs which require different numbers of words than their source language equivalents, the margin for error is vast. In such cases answer attempts which feature a mix of incorrectly positioned words (or sub-phrases), incorrectly spelt/conjugated words, incorrectly translated words, missing words and redundant words are much more likely to occur.

Consider an inexperienced student attempting to translate the following English phrase into German:

```
My youngest brother wanted to ski for the whole day
every Sunday
```

A correct answer is:

Jeden Sonntag wollte mein jüngster Bruder den ganzen Tag schifahren

But a typical erroneous attempt might be:

Mein jungste Bruder wollte der Tag skien an jeden Sonntag

Where a system cannot cope with re-ordering of elements in a user's answer, the best it is likely to be able to do in terms of feedback is this (missing words are shown in square brackets, redundant words in angle brackets):

[Jeden] [Sonntag] [wollte] **M**ein j**u**ngste_ Bruder <wollte> de**r** [ganzen] Tag **skien** <an> <jeden> <Sonntag>

- 4 missing words: Jeden, Sonntag, wollte, ganzen
- 4 redundant words: wollte, an, jeden, Sonntag
- 1 completely incorrect word: skien (for schifahren)
- 3 words with orthographic errors: **M**ein, j**u**ngste_, de**r**

This mark-up is not natural and features 9 'full' errors and 3 'partial' (orthographic) errors.

If partially correct words cannot be matched - as is the case in many systems - the resulting mark-up will have not 1 but 4 incorrect words: Mein, jungste_, der, skien, and thus 12 full errors in total.

However if the re-ordering of the phrase is taken into account, there are only 5 full and 2 partial errors:

- 2 phrases in the wrong position: "wollte", "jeden Sonntag"
- 1 missing word: ganzen
- 1 redundant word: an
- 1 completely incorrect word: skien
- 2 words with orthographic errors: j**u**ngste_, de**r**

Note that the incorrect capitalisation of the otherwise correct "**M**ein" is only due to its being at the start of the sentence because of the misplacement of other items.

Feedback in the LISC system

LISC consists of three parts – a question generator, an error-detection/feedback module and a course-management module. Only the second is of interest here. LISC's error feedback is generic and identifies five types of error, marking up each one using a combination of font-colour and

font-type or extra mark-up symbols. (Note that in a black and white article the mark-up will be far less clear).

Table 1. LISC Error Mark-up

Error type	Mark-up	Example
Completely incorrect word	**Bold red font**	He got out **from** bed too early
Partially incorrect word	Incorrect letters: **bold pink font**; missing letters: _	We saw a large ele**f**_ant
Incorrectly positioned word (or sub-phrase)	*Blue italic font*	Eat *in the morning* a good breakfast
Missing word	Brown square brackets: **[___]**	They went **[___]** the seaside
Redundant word	Green angle brackets: <word>	Most bread is made from <the> wheat

LISC's error-detection routines are implemented in Prolog, utilising its in-built backtracking to handle the high level of non-determinism inherent in the type of search problem presented by sequence comparison with element re-ordering. Computational complexity and resultant combinatorial explosion is a serious problem, but it is possible to prune search paths as soon as they appear non-optimal and this creates very real improvements in performance. The issue of multiple alignments with identical edit distances arises less frequently the more complex the phrases being considered, but judiciously-chosen heuristics help in such cases.

With the previous example LISC produces a mark-up with only 7 errors (5 full and 2 partial), compared to the 12 which traditional sequence comparison systems would generate.

> Mein j**u**ngste_ Bruder *wollte* de**r** [___] Tag **skien** <an> *jeden Sonntag*

This concise and easily learned mark-up is much less off-putting to students than the equivalent textual descriptions of each generic error – meaning that giving feedback on multiple errors does not have to involve displaying discouraging amounts of explanatory text.

With this approach, there is no need to predict errors, nor to establish answer-appropriacy as a separate stage of processing. Different versions of the system are not required for new language pairs, and because LISC does not use parsing, it will never fail to give feedback - no matter how poor or

confused the input. It is unlikely that a parser-based system *or* a token-by-token sequence comparator would cope with input such as this:

```
Correct:  I wandered lonely as a cloud
User:     Like alone a clod me wnader
```

Whereas LISC can handle it and produces the following mark-up:

```
Mark-up:  <Like> alone__ [___] a clo_d me wnader__
```

Note the bold *and* italic font for the word 'me', denoting that it is both incorrect and in the wrong place in the sentence – this type of match is very difficult to achieve.

Is the feedback effective?

The LISC system has been used for several years on an ab-initio Spanish module at the University of Kent, and more recently at a number of Grammar Schools for both French and Spanish. The system has been exploited in various ways in the different establishments (for in-class tests, in-class compulsory exercises, compulsory exercises to be completed in students' own time and voluntary practice exercises). This paper looks at its use at the University for compulsory exercises completed in the students' own time. Over the last four years LISC has processed over 56,000 student answers in this mode and the marks returned feed directly into the University exam boards.

Given the absence of specific grammatical detail in LISC's answer analysis, it has been necessary to establish whether or not the type of feedback it provides is effective. Three questions were posed:

- Can users identify (and therefore correct) their errors?
- Do users improve as they progress through an exercise (i.e. do fewer errors occur in their answers)?
- Do students find the system beneficial?

Users have two answer attempts in LISC exercises (unless correct on the first occasion). A question's score consists of the scores for both attempts, with the first attempt being weighted higher than the second. This means that students are inclined to think carefully about their first answer, but can improve their mark by paying attention to their errors. This is motivating and encourages an all-important "focus on form" (Long, 1991).

Figures 1 and 2 show the average first and second attempt scores for all 12 exercises used in the academic years 2006-07 (24 students) and 2005-06 (33 students). The same 12 exercises were presented in each case. The exercises varied considerably in terms of difficulty (with exercise 11 virtually no candidates ever got any correct answers on their first attempt), but it is

obvious that in all cases students were able to improve their answers in response to the feedback received.

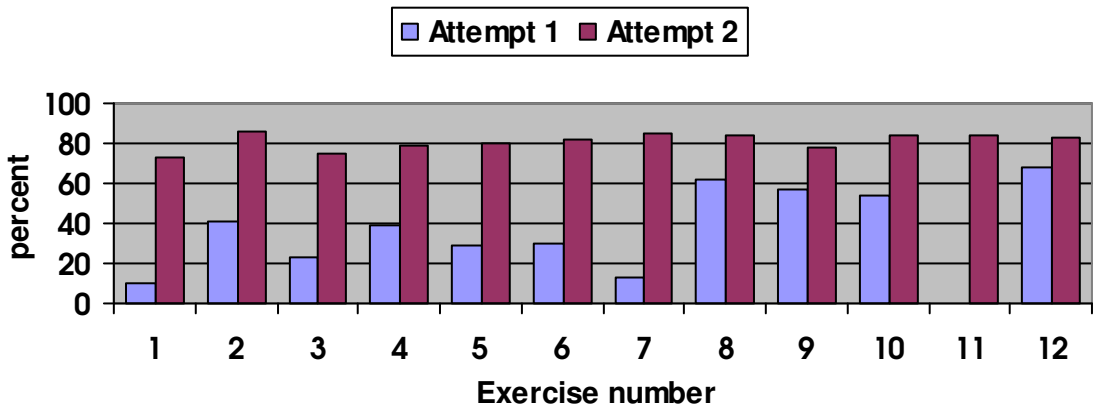

Figure 1: Average scores (by exercise) for 1st and 2nd attempts at questions *[2006-07, 24 students]*

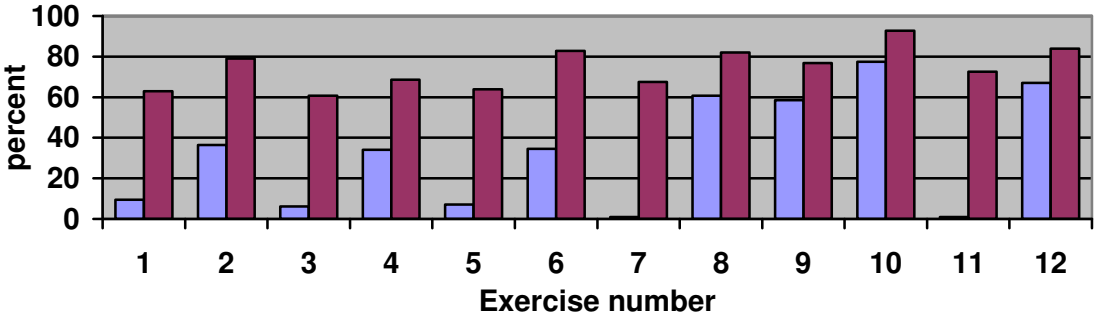

Figure 2: Average scores (by exercise) for 1st and 2nd attempts at questions *[2005-06, 33 students]*

Although involving two different groups of students, it can also be seen that the results from the two years are very similar. This is more apparent in figure 3, which shows the average improvement in score between attempt 1 and attempt 2 for the same 12 exercises in 2006-07 and 2005-06.

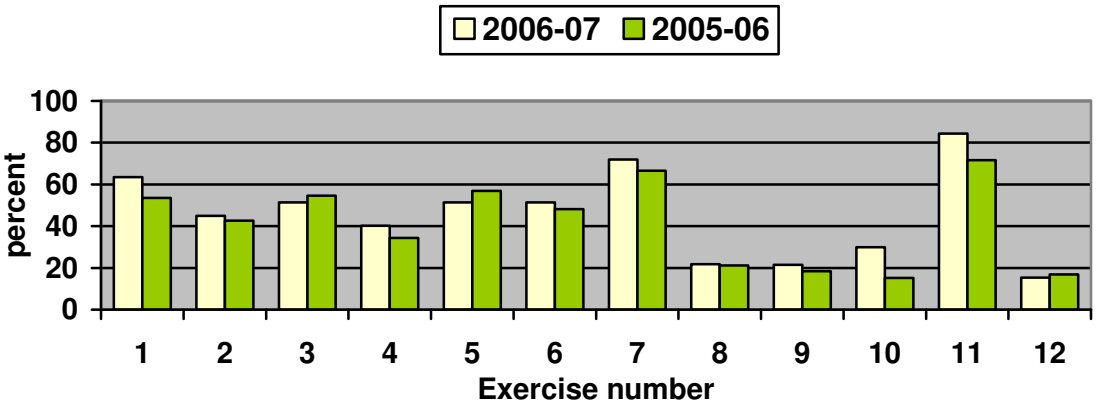

Figure 3: Improvements between attempt 1 and attempt 2 for 2006-07 and 2005-06

If all 12 exercises from 2006-07 are considered as a whole, the average improvement between first and second attempts is 45.39% - in other words the feedback, despite its generic nature, is helping users to identify and correct their mistakes.

To answer the second question (do students improve as they progress through an exercise) every results file for every student for every exercise was split into three equally-sized sections. The first sections of every data-file were merged, and likewise the second and third sections. These amalgamated sections were then analysed to see if there was any difference in the way students performed at the beginning, middle and end of each exercise. In the figures that follow the bars represent the amalgamated first, second and third parts of all results files for the 2006-07 cohorts.

Figure 4 shows the average scores for the *first* answer-attempts occurring in the three separate sections of the data files over the two years under consideration. By taking into account only the scores for first answer attempts the results are not skewed by the error-feedback (i.e. these are scores students received for each question prior to being given any hints). Although as a whole the students from 2005-06 did not perform quite as well on the exercises as those from 2006-07, the average improvements between the first, second and third data sections are consistent between the two cohorts.

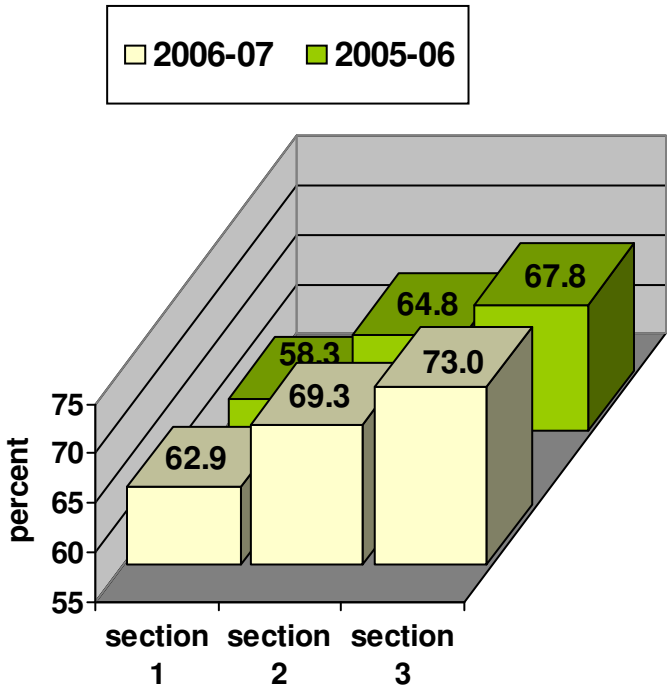

Figure 4: Averaged first-attempt scores

Figure 5 shows the average percentage of questions which students answered correctly on their first attempt. Despite the actual difference in marks for the two years, the improvements between data sections one, two and three are again consistent.

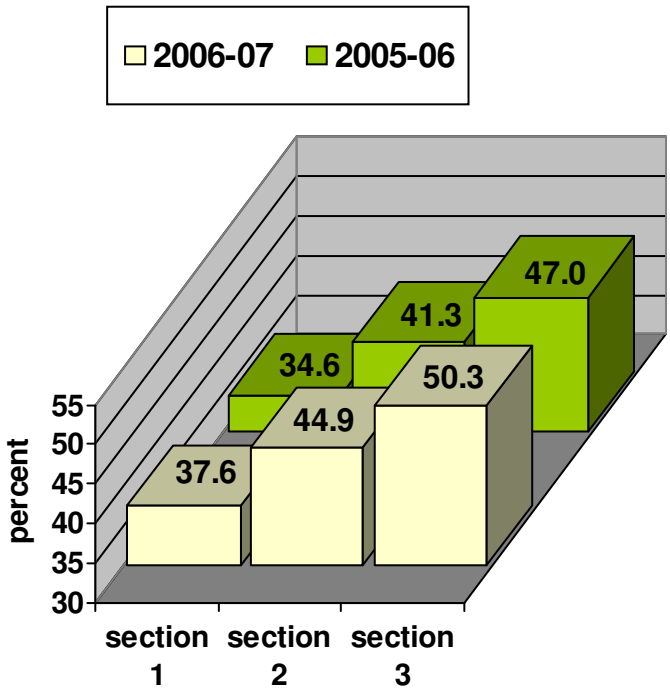

Figure 5: Percentage of correct first attempts

From both these figures it can be clearly seen that as students progress there is a marked improvement in the accuracy of their answers. It should be stressed that this is not simply an effect of short term memory because students do not have to complete an exercise in a single session and may return to it on as many occasions as they require. Some students complete an exercise in a single session; others take several sessions for one exercise (eight sessions seems to be the maximum for an exercise with an upper limit of 40 questions). Furthermore the order of presentation of questions within an exercise is entirely random, so question order has no influence on scores.

Figure 6 shows the average thinking time for first attempts at questions for each of the three sections of the data. The bars for each section show the percentage of the total thinking time for all first attempts at questions in the relevant year. As with figures 4 and 5 only first attempts at questions are considered, so the results are not skewed by the time taken for students to consider feedback. The results from 2006-07 very closely mirror those from 2005-06 and it is evident that as students progress through an exercise their thinking time decreases markedly.

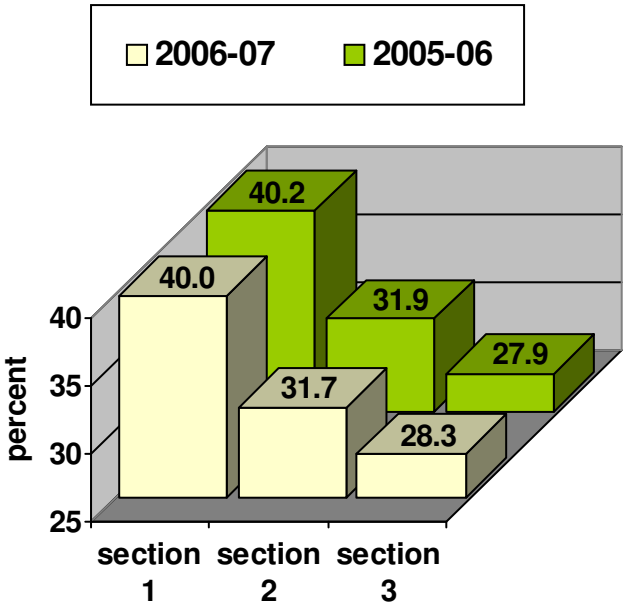

Figure 6: Percentage thinking time for first attempts

With reference to the third question (do students find the system beneficial) it is useful to look at the pattern of use. Table 2 shows the usage figures for the LISC system at the University of Kent over the past four years.

Table 2. Usage of the LISC system at the University of Kent

LISC USAGE	03-04	04-05	05-06	06-07
Average users per lesson *(dependent on cohort size)*	66	21	33	24
Lessons available	16	12	12	12
Total questions attempted	29,797	6,902	11,494	8,339
Average min/max qu's required per lesson	19-38	19-38	19-38	19-38
Average qu's per lesson per user	27	26	29	28

Each exercise has its own minimum and maximum number of questions to complete. There is no obligation to complete more than the minimum number of questions per exercise and full marks can be achieved in this way. However students are permitted to continue beyond the minimum number of questions, and for the most part attempt considerably more questions than

they are obliged to (as can be seen from the final two rows of the table). Taking into account just the exercises set in 2006-07, the total minimum number of questions to be answered (per student) was 230 - yet the average number answered was 340 (48% more than required). This pattern has been consistent throughout the system's use at the University and backs up anecdotal and survey evidence that students do indeed find it beneficial.

Conclusion

The analyses show that as students progress through exercises they become more accurate in their answering and at the same time are able to formulate those answers more rapidly. This sort of improvement would be unlikely to occur were the feedback not useful. Data from the thousands of responses processed by LISC indicates that its algorithms mark up errors in a way that students find natural and easy to comprehend. Feedback on the system has been very positive. A survey of users in 2006 revealed that they found the system easy to use, strongly favoured the immediacy of its feedback and liked being able to review their submitted work at any stage. For tutors LISC has massively reduced marking loads - last year alone staff at the University were spared the marking of over eight thousand questions.

This system can be used to deliver any type of task where there is a definitive set of answers – sentence-translation, dictation, rewording (e.g. active to passive conversion), reading comprehension, vocabulary testing etc. and because the system is language-independent it can be used for exercises in many languages with no modification to the error-detection routines. Stockwell (2007) notes that CALL is a field which would benefit greatly from collaboration amongst researchers and content providers - both at the application and content levels. The fact that LISC can incorporate exercises of different types and in many languages means that language teachers can develop content to be used with their own courses or shared with the wider community. This is already occurring with French and Spanish exercises developed at the University being used by two local grammar schools, and a third (non-local) grammar school developing a pool of French exercises for use both on its own courses and for sharing with other interested secondary level institutions.

References

Ashton, H., and Thomas, R. (2006). Bridging the gap between assessment, learning and teaching. *Proceedings of the 10th International CAA Conference*, 25-36

Bailey, S., and Meurers, D. (2007). On automatically evaluating answers to reading comprehension questions. *CALICO 2007*, San Marco, Texas

Bangs, P. (2003). Engaging the learner – how to author for best feedback. In U. Felix (Ed.), *Language learning online – Towards best practice* (pp. 81-96). Lisse: Swets & Zeitlinger.

Celce-Murcia, M. (1997). Direct approaches in L2 instruction: A turning point in communicative language teaching. *Tesol Quarterly, 31(1),* 141-152.

Delmonte, R. (2003). Linguistic knowledge and reasoning for diagnosis and feedback. *CALICO Journal, 20(3)*, 513-532.

Desmet, P. (2005). Les enjeux de la création d'un environnement d'apprentissage électronique axé sur la compréhension orale à l'aide du système auteur IDIOMA-TIC. *ALSIC (Apprentissage des Langues et Systèmes d'Information et de Communication), 8,* 281-303

Dodigovic, M. (2005). Artificial intelligence in second language learning: raising error awareness. Clevedon: Multilingual Matters Ltd.

Guest, E., and Brown, S. (2007). A new method for parsing student text to support computer-assisted assessment of free text answers. *Proceedings of the 11th International CAA Conference*, 223-238

L'Haire, S., & Vandeventer Faltin, A. (2003). Error diagnosis in the FreeText project. *CALICO Journal*, *20(3),* 481-495.

Heift, T. (2003). Multiple learner errors and meaningful feedback, *CALICO 20(3)*, 533-548.

Hémard, D. (2003). Language learning online: designing towards user acceptability. In U. Felix (Ed.), Language learning online – Towards best practice (pp. 21-42). Lisse: Swets & Zeitlinger.

Hyland, F. (2003). Focusing on form: Student engagement with teacher feedback. *System, 31(2),* 217-230.

James, C. (1999). Language Awareness: Implications for the Language Curriculum. *Language, Culture and Curriculum, 12(1)*, 96-116.

Karlström, P., Cerratto-Pargman, T., Lindström, H. and Knutsson, O. (2007). Tool mediation in focus on form activities: case studies in a grammr-exploring environment. *ReCALL, 19 (1),* 39-56.

Levenshtein, V.I. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics-Doklady, 10 (8)*, 707-710.

Long, M.H. (1991). Focus on form: a design feature in language teaching methodology. In K. de Bot, R. Ginsberg, & C. Kramsch (Eds.), *foreign language research in cross-cultural perspective* (pp. 39-52). Amsterdam: John Benjamins.

Lowrance, R., and Wagner, R.A. (1975). An extension of the string-to-string correction problem. *Journal of the Association for Computing Machinery*, *22(2)*, 177-183.

Menzel, W., & Schröder, I. (1999). Error diagnosis for language learning systems. Language Processing in CALL, *ReCALL (special edition, May 1999)*, 20-30.

Nesbit, J.C., & Nakayama, K.N. (1990). Sequence comparison applied to correction and markup of multi-word responses, *CALICO 7 (4)*, 28-35.

Schwind, C.B. (1990). An intelligent language tutoring system. *International Journal of Man-Machine Studies*, *33*, 557-579

Schulz, R.A. (2001). Cultural differences in student and teacher perceptions concerning the role of grammar instruction and corrective feedback. *The Modern Language Journal, 85 (2),* 244-258.

Schulze, M. (1999). From the developer to the learner: describing grammar – learning grammar. *ReCALL, 11 (1),* 117-124.

Stockwell, G. (2007). A review of technology choice for teaching language skills and areas in the CALL literature. *ReCALL, 19 (2),* 105-120.

Tschichold, C. (2003). Lexically driven error detection and correction. *CALICO Journal*, *20(3)*, 549-559.

Truscott, J. (1996). The case against grammar correction in L2 writing classes. *Language Learning, 46 (2),* 327-369.

Van der Linden, E. (1993). Does feedback enhance computer-assisted language learning*. Computers & Education*, *21(1-2)*, 61-65.