Henk Louw

Design considerations for CALL based upon evaluation criteria for CALL

A B S T R A C T This article presents a rubric for the evaluation of Computer-Assisted Language Learning (CALL) software based on international recommendations for effective CALL. After a brief overview of the pedagogical and implementation fundamentals of CALL, and a discussion of what should be included in a needs analysis for CALL evaluation, the rubric is presented. The author then illustrates how the evaluation criteria in the rubric can be used in the design of a new CALL system.

Keywords: Software evaluation, CALL, language laboratory, MarkWrite, writing across the curriculum, software development

1. Introduction

Computer-assisted language learning (CALL) came onto the scene of language pedagogy almost at the same time as the advent of the personal computer but, much as in the case of automatic translation, has not made as much headway as was once enthusiastically expected (Hémard, 2006). A disappointingly small number of lecturers espouse the use of CALL. Just as new language learning books are continually published, new computer programs for learning and coaching languages are also produced annually. While the criteria for the creation of a text book are relatively fixed, the criteria for evaluating and designing CALL are not as set in stone, largely because of the immense possibilities of the medium and the number of variables to take into account. Just as a book is written with the reader in mind, and evaluated accordingly, this article argues that a CALL system should be evaluated as well as designed with a set of detailed considerations in mind.

While many articles have been written on the evaluation of software (many of them are given in the bibliography), none of them are complete enough to use in practice to make an informed decision on the best CALL package to purchase. Even worse, none of them makes explicit the link between what is evaluated and what is designed. To put it more bluntly: a

complete evaluation grid could not be found for CALL software packages. Secondly, it seems as if designers and evaluators work from two different rule books – designers design what they think is needed, whereas evaluators evaluate according to their needs. It is easy to identify a few reasons for this. For example, the complex nature of language pedagogy and the individual needs dictated by different contexts may be crucial to why software is evaluated differently in specific situations.

The immense cost of both purchasing software (when compared to books) and developing software (also compared to books) simply increases the urgency of establishing a detailed set of considerations for evaluating and designing software.

The purpose of the article is twofold – to establish which considerations should be used to evaluate a CALL system and to illustrate how these considerations could be used in the design of a CALL system. The design of the new software program MarkWrite is used as an example to illustrate how these considerations may be used.

Although there are many similarities between selecting a CALL system and selecting a new course book for a module, selecting a CALL system is more difficult. One obvious difference is that it is easier to quickly page through a book to get an overview of its contents, while it is a considerably bigger and costlier task to buy, install and evaluate a piece of software. The more interactive nature of software also provides many variables which are not part of the evaluation of a textbook, such as more graphics, sounds and audio, navigation, system requirements, and many other computer-specific considerations. The long-term implications of the choice of software may be more severe, since it is often an institution which has to buy software, whereas it is the students who buy new textbooks annually for themselves.

This article first provides general considerations for the use of CALL from national and international literature, after which a set of evaluation criteria for CALL is proposed. The evaluation criteria are then used as design criteria, using the MarkWrite software as an example.

2. Basic understandings and terminology of CALL from the literature

Computer-assisted language learning, as most things in the computer world, is a fast-developing discipline. Some terms and agreements have come into effect regarding the design and evaluation of CALL systems. Among these are eight fundamentals, falling into roughly pedagogical and implementation categories, each of which is elaborated on:

Pedagogical fundamentals

1. CALL- systems need to have a solid educational base and be integrated into the whole teaching curriculum. The effectiveness of CALL depends on an effective pedagogical base and not on the computer. Good pedagogical practice is just as necessary for CALL as it is in traditional teaching.
2. CALL should be seen as a teaching tool. A computer is not a human. Each may be better at their respective tasks, but CALL is and remains a teaching tool and cannot be a total solution by itself, even if the system is marketed as a stand-alone application.
3. There is a difference between a coach and an independent teaching tool. Although all CALL-systems are teaching tools at the disposal of humans, some CALL programs are

designed to be utilised *only* as tools while others are stand-alone applications designed to teach *independently* of human teachers.

4. CALL systems are specialised teaching tools. There is no *single* computer program available to do comprehensive CALL. For completely computerised language learning to take place, it would be necessary to invest in more than one program, each focusing on a different aspect of teaching.

Implementation of CALL

1. Effective CALL is dependent on an effective policy. A policy must be created for the use of CALL in a language laboratory, classroom or module. Such a policy would have to indicate how the other considerations are to be handled.
2. CALL is dependent on a CALL environment. The whole context of the teaching situation influences the effectiveness of the CALL system. A CALL environment is not dependent simply on the quality of the computers or the software – the organisation of the physical classroom and the implementation of the programs are also important. This is just as relevant if CALL does not take place in a language laboratory, but is instead used as self-study tool.
3. All staff members need to be trained in the use of the program, but a dedicated CALL technician or CALL manager has to be appointed.
4. Software has to be evaluated annually to establish if it is still relevant to the specific educational situation. It is no use sticking to a program that does not perform or is not suited to the specific environment or application, or which does not deliver as promised.

Each of the above aspects is now dealt with in more detail.

3. CALL systems need to have a solid educational base and be integrated into the whole teaching curriculum

Teachers should assess software just as carefully as they scrutinise textbooks. Electronic resources can by their nature provide access to authentic language samples, but it is up to the teacher to structure activities and projects that promote meaningful interaction with these materials (Bradin, 1999:175; Buell, 1999:217).

Barr and Gillespie (2003:69) explain that "a computer-based environment needs to be carefully constructed in order to ensure that all the other components of learning are effectively integrated into it. It is important to ensure that the uses of computer technology in this type of environment are not seen as separate, but rather that they are integrated, working together to enhance the process of teaching and learning … CALL packages must not be seen as stand-alone creations." The technology must be used to integrate the learning and teaching methods with the resources available. Technology need not be used at all costs. "If other more conventional teaching and learning methods work well, then there is little point in using computer technology."

Technology should be support for a total **environment for learning**, instead of a stand-alone tool or source of information only. Technology can change how, what and whom is taught, but it is more important to understand good pedagogy than to understand the technology (Egbert

et al., 1999:ix-1; Murray and Barnes, 1998). The pedagogical goals must be clear and the use of technology must further these goals (Bradin, 1999:160), while all CALL should be grounded in sound teaching methods (Levy, 1997).

Egbert *et al*. (1999:2-3) make it clear that despite the large number of language acquisition and language learning theories, researchers and teachers generally accept that:

- language acquisition is the result of an interplay between some kind of cognitive mechanism and environmental factors;
- not all language learners learn in the same way, at the same rate or for the same purposes;
- interaction between learners and other speakers is very important.

Very broadly, the following assumptions about learning are evident in the pedagogical theories currently in use:

- "Language learners must be involved not only in social interaction but in *purposeful* interaction, which includes a real audience that is actively involved with the learners" (Egbert et al., 1999:4).
- Learners should have an authentic goal for their work. Authentic tasks have the same type of cognitive challenges as complicated real-world tasks. "It is important to design tasks so that students can use their current proficiency level to function in authentic communications" (Egbert *et al*., 1999:5). The Internet and e-mail are useful tools for real, live communicative tasks.
- Learners should be exposed to a variety of sources of input (Egbert *et al*., 1999:5).
- Educators must assist the learners by creating an environment with an optimal stress level. This is done by creating a learner-centred classroom in which learners have some control over their learning. The educators' expectations must be reasonable and the goals attainable (Egbert *et al*., 1999:6).
- A learner-centred classroom is necessary. Such a classroom is one that develops learners' confidence and skills to learn autonomously. Learners should also be able to design and coordinate tasks in a variety of contexts (Egbert *et al*., 1999:6).
- Peyton (1999:17) writes that all learners move in their speaking and thinking towards a stage where they can function alone.
- Written communication is important for learning. Peyton (1999:17) notes, "Most work on the dynamics of interaction and their effect on learning has focused on oral interaction. However, research on written interaction in dialogue journals with teachers and in letters exchanged with older students has shown that these interactions can also develop language, thought, and reading and writing abilities".
- Co-operative learning is important. Staton points out, "To be able to think in new situations – which is the real goal of all education – [learners] need a lot of experience in thinking with someone who is good at it. Just as we learn language by talking with someone who is good at it in specific situations concerning tangible, shared experiences, so we learn to think by thinking with someone to solve a joint task or problem" (quoted by Peyton, 1999:18).

Many of the above criteria can be met by the judicious use of computers in the classroom. Egbert *et al*. (1999:8) write that "… just as there seems to be no one right way to teach or learn language, there is most likely no one best way to use computers for language learning." Their

argument encompasses five critical questions to ask about the computer-assisted classroom:

1. How effective is group work as an aid to L2 learning?
2. Should students drill and practice new structures?
3. What can be done to encourage participation among students who seldom ask questions or initiate interaction?
4. To what extent does the correction of errors assist in L2 learning?
5. Which technologies are best for supporting the best methods of teaching and learning?

All five of these questions are just as applicable to pedagogy in normal computerless classrooms.

4. Teaching tool

A computer cannot replace a human teacher. Bradin (1999:159) explains that any appraisal of a software package should be based on knowledge of what computers are capable of and what the inherent drawbacks of computers are. Teachers and computers may each be better at certain specific tasks (Egbert et al., 1999:9). Ma and Kelly (2006: 21) trace CALL efficiency back to information available about the learner, which should influence *theory, computer technology* and *user actions*. Computer-assisted pedagogy is therefore much the same as normal human interaction pedagogy, but a CALL classroom (teaching situation) cannot be directly compared to a normal classroom as the technology introduces many new variables. **Course content** should be emphasised as the focus of instruction – not the computer. The computer is a tool and not a human teacher (Sivert & Egbert, 1999:46), but since a computer communicates with learners, they can in fact learn communicative competence by simply using the computer (Chapelle, 2003:11). For example, the computer asks questions of the user such as: "Do you want to save the changes made to document X?" However, learners should already have basic computer literacy to enable them to use CALL software.

As a tool, computers have advantages. Peyton (1999:17) describes the following advantages of computers:

- It is a medium of communication that creates new opportunities for writing and learning.
- Computers provide synchronous (real-time) and asynchronous (time-delayed) interaction.
- One-on-one interaction between students and teachers or among students within classrooms is possible.
- Wider communication with individuals and groups around the world is possible as well.
- Text and talk are available in the classroom and in a rapidly expanding world.
- Resources are not bound by physical space.

Despite the above qualities, computers cannot be compared directly to a human teacher, but should be seen as very effective tools which can be utilised by a human teacher. Computers do not replace teachers, but simply change the nature of their work. Chapelle (2003:xiii) states that although the use of technology is regarded as the obvious (unmarked, normal and natural) way to go, the case to be made is that CALL should not be compared with classroom language learning. Rather, "the challenge is to provide evidence for the most effective ways to design software for CALL, to use the software effectively in tasks, and to help learners to take advantage of the electronic resources available to them." In other words, use the computer as a tool and not as a teacher replacement.

To elaborate on the difference between human teachers and computer tools, consider the common complaints against CALL systems. A 2003 survey of CALL systems done by the author identified common weaknesses in almost all the CALL software evaluated. For obvious reasons the programs surveyed cannot be mentioned here. These weaknesses are:

- Students can click through most of the screens without filling in something. Common pedagogical principles require that students should take some action. While many human teachers also allow students to be passive, this is not good practice.
- If the program requires something to be filled in, the user can simply type gibberish and the computer will accept it. Great advances have been made to counter the gibberish effect, but in some cases the software is still not able to detect it.
- The interfaces are in most cases very plain (which in itself is not a problem), but there is a lot of space that could have been used for additional information or supporting graphics.
- Most of the programs do have supporting graphics, but these vary in their effectiveness. The programs that the author found to be the most interesting and effective had very good supporting graphics.
- Most of the programs do not provide adequate feedback. Especially in a computer environment where one can basically "click until you get it right," feedback is needed on correct **and** wrong answers.

Students may also be sceptical of the computer's ability to judge their work effectively (Spencer & Louw, 2009). It should, however, be kept in mind that sometimes students are also sceptical of their human teachers' ability to judge their work effectively.

A final consideration to keep in mind is the distinction between software for language improvement (used by students who already know the language to an extent) and software aimed at teaching an unknown language from scratch. This article deals only with software for language improvement.

5. Coaches versus tools

When considering CALL for a university or school set-up, there are essentially two types of systems available in two different mediums: coaches and tools are available *online* (local area network or the Internet) and for *local* (stand-alone PC) units. A discussion of the differences and advantages/disadvantages of coaches and tools merits a whole separate article. Davies and Hewer (2009) distinguish between six different types of software and more than 20 different applications of Information and Communication Technology (ICT). The simplified distinction will be presented here:

1. **Coaches** are stand-alone units designed to be accessed by the students and worked through from start to finish. The programs are pedagogical (computerised "teachers"), providing lessons, examples, prompts and tests. The tests can comprise monkey puzzles and drag-and-drop quizzes. Many of the more modern systems are language coaches simulating the complete teaching experience. Someone can buy such a system from the local computer store or book store and independently start learning a language from scratch. In the South African context, an example would be the language learning software created by the Centre for Text Technology at the North-West University, such as Tsenang (Berg & Pretorius,

2003). More international language learning software, such as the Pimsleur or Rosetta Stone software, are sold over the Internet.

2. **Tools** are aids to teaching. They may also have lessons, examples, prompts and tests, but in addition they can be altered by the teacher to suit his or her own needs. In some cases the tools do not have any lessons or examples, unless the lecturer creates them. The most basic of these tool programs simply provide the students with drill-type exercises and are seldom if ever used independently of a structured curriculum. Standard corpus linguistics software used to analyse student writing may also be considered a tool and its use as such is discussed by Cowan, Choi and Kim (2003) and Granger (2003).

Some programs are composite (both tools and coaches). They aim at developing skills in some or all areas of language and provide tools to lecturers to assist them in customising the software to their specific needs.

Arnett (2009:27) however, warns against an over-dependence on computer tools. In his article he bemoans the poor ability of students to spell, and their blind reliance on their computer grammar and spelling checker. He implicitly warns against turning tools (such as a spelling checker) into a coach.

6. Specialisation

At present, most available computer programs are very specialised in content. For example, some are just aimed at improving writing, whereas others are just for improving vocabulary. To address more than one aspect of learning, more than one computer program would be necessary. One piece of software is never enough (Sivert & Egbert, 1999).

Related to this is the implementation of the CALL. One must establish whether the software will be used as bought for a specific module as a whole, or as support for one specific outcome in a module. One also needs to establish whether the CALL system will be used as a stand-alone module in a CALL laboratory where students have to work through it as part of their class activities, or whether it is simply support for them which they should work through at their own pace at home. As with normal classroom interventions, custom-made teaching materials are better than commercially available, generic material. CALL software which can be adapted to the specific situation is therefore a better option than generic software. See Chapelle (2003); Cowan, Choi and Kim (2003); and Granger (2003) on this topic. Specifically Granger (2003) and Cowan et al. (2003) use the computer to enhance their custom-made exercises.

7. Policy

Aligning module outcomes and their application is difficult whenever more than one person is working on the same module or set of modules. Adding a computerised section or computerised support for the module simply adds another variable and another participant to the mix which necessitates structured collaboration. Consider the view of Barr and Gillespie (2003:78):

> All learning environments are complex, but computer-based language-learning environments are particularly complex, so co-ordination is vital. We have found that the most successful integration of environments has occurred when there has been high-level management support for the development of such environments, not only in providing the finance, but also in shaping the direction and motivation of the system.

Bishop (1999) distinguishes between a mission for the CALL environment, policy for the classroom, and policy for the software. Guidelines for the policy of a CALL laboratory or classroom according to Bishop (1999:272) should state:

- When and how hardware and software are upgraded
- Who is allowed to download and upload files
- Where information on policy is kept and what it is used for

In addition, the mission of a CALL environment should ask the following questions:

- What is the instructional rationale for the use of CALL?
- Are the computer applications appropriate, effective and intelligently applied?
- What student results are expected?
  (Bishop, 1999:272).

The following are Bishop's (1999:275-276) recommendations for a *software policy*:

- Recommendations for new software will have to be made at least annually or two or three times per year. Recommendations must be archived with the reasons for the recommendations.
- The software evaluation form can be developed in-house. The idea is that the form should prevent duplication of recommendations and also provide ideas for the use of the available software.
- A minimum 30-day trial period of any software considered for purchase. Personal experience and observation are very important.
- The linking of software with learning objectives: the software may be excellent, but if it does not fit your curriculum, you are wasting your money.
- No-one but the system administrator should be able to install software on the computers.

The various considerations for CALL software and the evaluation thereof could fill volumes, but the above points highlight some of the core principles. A next step in evaluating CALL software is to do a needs analysis.

8. Environment

Materials are not inherently better just because they are on the computer. CALL should ideally occur in an optimised learning environment. Sivert and Egbert (1999:41) describe the ideal technology-enhanced language learning classroom:

> The word *classroom* implies a place where different kinds of learning can take place and where technology use is subordinate to discovery and understanding. In this setting, learners enter a classroom designed for comfort and collaborative learning. They ideally find a cushioned seat equipped with casters in front of a large desk with a recessed monitor; each desk is part of a group of four desks facing one another. Books, papers, and pens are spread over the quad of desks. As learners begin their work, they move easily among other members of their quad and even among other members of the class and the instructor. Instead of concentrated silence, one hears the lively discussion of learners working together on task-oriented and project-based assignments. The software available assists in driving the assignments, and several other media are used in developing and completing the tasks.

The resources available may not always permit such a classroom, but three types of technology-enhanced language learning classrooms exist:

1. The self-access lab
2. The computerised instructional classroom
3. The language development centre
   (Sivert & Egbert, 1999:42).

Each of these has its own advantages and disadvantages which Sivert and Egbert (1999:42) discuss in more detail. As mentioned earlier, the use of CALL may in fact not be dependent on a specific classroom as students may work independently at home or simply learn from the computer as an additional source of input. The immersion principle suggests that learners are given enough exposure to the target language to develop their ability to comprehend the language. With the massive growth of the Internet, virtual immersion is possible for any student, but instruction would still be necessary to form and shape language acquisition (Chapelle, 2003:36).

A separate consideration under the environment is the level of interactivity which is available and the extent to which this interactivity is desirable. Some CALL systems (especially the online versions) introduce students to other students in other parts of the world as "chatting partners" to enhance their written communication skills in a real-world setting, and it appears to work (Fitze, 2006; Yuan, 2003). However, the poor quality of "chat room English", Mxit language and cellphone abbreviations (SMSs), may be the very reason why most teachers will discourage these practices despite research findings indicating that they do not have such a big adverse effect on students as is proclaimed in staff tea rooms and corridors (Plester, Wood and Bell, 2008).

Whichever model is chosen, it should be kept in mind that CALL is not a stand-alone activity. The specific outcomes of the computerised component should be discussed and settled on by all stakeholders, and the computerised component explicitly integrated into the curriculum. Everybody working with students will know that if something does not count for marks, very few students will work on it independently. This also means that a participation mark is not acceptable, since it degrades the perceived status of the computer component (Spencer & Louw, 2008).

9. Staff

A surprising number of academic staff (from all disciplines) are closet technophobes and are allowed to be so by their institutions. Technophobia may result in initial negative reactions to software (as also noted by Murray & Barnes, 1998:250). It may take some convincing to persuade especially older staff to (a) see the benefits of CALL and (b) once they have realised the benefits, to consent to being trained in its use. A single pundit of CALL at a campus is simply not enough to successfully implement CALL, and this person will soon tire of continually having to motivate or nag his or her colleagues to use the available technology.

With regard to training, Bishop (1999: 278) makes it clear that **all staff members** should be explicitly trained in the use of the program or programs. It is no use having one expert whose absence causes the system to come to a halt. All teachers should know what the software is able

to do to optimise its use. "Too many beautiful CALL centres have good software but only one person who knows about it. Such a centre is not a CALL environment; it is just a room full of computers." Barr and Gillespie (2003:77) have found that untrained, uninvolved or uninformed staff cause students to react negatively to CALL, as well as when there is limited, incomplete or inadequate equipment. It should be clear therefore that untrained staff (and staff who are unwilling to be trained) can scuttle the complete operation, and ignorance should therefore be avoided at all costs. Davies et al. (2009) also caution that staff training is an "ongoing and unending process" for which funding should be made available every year.

10. Relevance

CALL should continually be assessed to ensure its continuous relevance to the situation. It is important to note the Hawthorne effect: "Any group that is being studied while doing a new or different activity usually performs better" (Egbert *et al*. 1999:10). Therefore CALL-systems should be evaluated annually to see if the programs are furthering the specific learning goals (Egbert *et al*., 1999:11-12; Bradin, 1999:160). It is very important to ensure that a program is not just being used because it is available. "Every piece of software in the CALL centre must have a direct, positive impact on and relationship to what is being taught elsewhere in the school" (Bishop, 1999:274). Most programs create their own sets of data which can be analysed, but it is imperative to state in a CALL policy who gets access to this data in order to preserve the students' rights (Bishop, 1999:281-282). The evaluation should also be done by using a definite set of criteria stated in the CALL policy and it should be done *while using the system*. This is very important for research.

In the light of the above, Bishop (1999:273) provides some acceptable and some poor reasons for using CALL:

ACCEPTABLE reasons for using CALL:

1. Classes are too large to monitor individual progress.
2. Students in the same class are of varying levels and need more individual attention.
3. Students need CALL to prepare for their real-life business environments.
4. Students need more one-on-one practice than they can get in the classroom.
5. Students have to do collaborative projects as a means of enhancing communicative skills.
6. CALL provides for enriched, alternative means of communication.
7. CALL provides practice in a skill or an introduction to concepts that cannot be offered otherwise.

While these all seem like legitimate reasons, the tipping-point question is still most probably: "Does it work?" Davies and Hewer (2009) discuss the reasons for using CALL at length, starting with:

> **Concrete evidence** on the effectiveness of CALL is difficult to obtain. There is plenty of **anecdotal evidence** about the positive effects of CALL. Teachers often report on their students being "enthusiastic", "engaged", "motivated" and even "excited" in classes in which CALL is used, but are sceptical about **measuring** its effectiveness.

While anecdotal evidence is not overly convincing, the authors cite many different case studies and other research on the effectiveness of CALL which appear to point in the general direction

of CALL being effective. Ngu and Rethinasamy (2006) however, found traditional teaching to be more effective in their context. Also compare Murray and Barnes' (1998) discussion of how to get past the initial impression of software to evaluate its true pedagogical value. Tsiringa and Virvou (2004) tested their students with a pre-test and post-test, but they were only comparing two different kinds of software. The pre-test and post-test approach might be useful. For this approach to work, however, the software has to be purchased, and the question remains how to efficiently evaluate software without going to the trouble and expense of purchasing and implementing it. The evaluator should be able to determine whether a CALL system is likely to work before implementing and testing it.

Davies and Hewer (2009) also indicate that the immediateness of feedback is considered a big advantage of CALL by students and throughout their lengthy discussion of five different CALL centres in Europe Davies *et al*. (2009) mention many additional advantages of using CALL, some of which contradict Bishop (1999). It seems that the advantages and disadvantages vary depending on the specific types of activity and the structure of the CALL, but in each case there are definite advantages and disadvantages to using CALL in that situation.

POOR reasons for using CALL according to Bishop (1999:273) are:

- CALL is enjoyable (Davies et al. (2009) differ on this, though).
- It is available.
- Everybody is doing it.
- Students want to play on computers (once again, Davies *et al*. (2009) state the opposite – playing on computers may motivate students).
- Computers keep students busy.
- The teacher needs some extra preparation time.

Davies *et al*. (2009) mention another poor reason – saving money. They state quite clearly that "technology is much more expensive than 'chalk and talk'…", but that "unfortunately, administrators in schools and universities are prone to regard the use of technology as a means of cutting down on staff, in the belief that 'throwing hardware at a problem' will save money". They refer specifically to a computerised language centre, but one can safely assume that their opinion also applies to CALL in general.

The issue of relevance has become even more evident with the advent of ICALL systems (*Intelligent* computer-assisted language learning systems) in which the computer learns what the students' specific needs are and adapts accordingly. While ICALL promises to be highly effective in individualising pedagogy, there is still a lot of research to be done before programs are made really intelligent. Most systems are intelligent on only a small part of the curriculum. The system evaluated by Tsiriga and Virvou (2004), for example, only focused on teaching the passive voice.

In short then, evaluating the effectiveness of CALL involves more factors than evaluating the effectiveness of a textbook or module. While the software is often more focused, there are many more variables to take into account.

11. Needs analysis

As with designing any new module, the first step in selecting software is to do a needs analysis with the parties involved. The findings of this needs analysis should be included in the evaluation rubric of the software.

The parties involved in CALL are, however, more diverse than those involved in simply designing a new module. In this case, three different sets of end users should be accommodated. Simultaneously, issues of integration, policy (already discussed) and budget are also "parties" to take into account. The three sets of users (in no particular order) are:

Set 1: the lecturer and the systems administrator

Set 2: the students and the systems administrator

Set 3: the university IT personnel and the systems administrator

The systems administrator is the only role player directly involved with all the other parties. Each of the identified parties is elaborated on below.

12. Lecturers

A needs analysis done at the Potchefstroom Campus of the North-West University found the following with regard to CALL: Lecturers want a program that is user friendly, does not require a lot of training, is intuitive and is adaptable to their needs. The program should decrease their workload and should not necessitate additional administration. The lecturer should also be in a position to indicate the specific outcomes required of the software.

These expectations are, however, overly optimistic of the abilities of the software. Lecturers may, for example, expect the same program to teach and correct grammar as well as teach argumentation and editing skills for various levels of student proficiency. This is obviously impossible for a single system to accomplish. Different linguistic skills are not directly equivalent to different levels of student proficiency and, unlike humans, a computer cannot hazard guesses as to the reasons why students make certain errors. It is important therefore that the lecturer should rank the requirements according to desirability. This ranking may then be cross-correlated with the ability of the software to accomplish the outcome and the value it will add for the lecturer. Something which is easy to teach and mark, but is time-consuming, would therefore be ideal to delegate to a computer, on condition that it is done in a monitored, structured way.

13. Student needs

As indicated earlier, it is bad practice to use a system simply because it is available. The specific needs of the student should be taken into account. While the lecturers are often in a position in which they can indicate these needs, their intuition is not always accurate (Louw, 2007:96). It may be advisable to ask at least a few students to test a piece of software since their opinions are relevant seeing as they are the ones who will use the software (Lasagabaster & Sierra, 2003). It will further enhance the understanding of student needs if they write a standardised test to establish accurately what their exact language needs are.

A second important variable with regard to student needs is the time spent learning to use the software. Tsigira and Virvou (2004) found in their study that one system was more effective than another, but that the more effective one required more time to master by the students. Time is important to all, and students are quick to complain about anything which they deem to "waste" their time. A computerised language learning system, no matter how effective, should

not be an add-on to an already full curriculum, otherwise students will see it as punishment. A cost/benefit analysis might be necessary to establish whether the time spent learning and using a new software package is worth the benefit provided by the package.

14. Systems administrator and IT personnel

It is preferable to have a separate systems administrator and IT support person, but in some instances the budget constraints will not allow for that.

14.1 Systems administrator

In a CALL system, systems administration is not a small job. Any system, no matter how advanced, has to be monitored and administered. This job should not just be dumped on the person with the most IT skill but should be assigned to a person who is aware of the module outcomes and who can make sure that the system is used to meet these outcomes. It is also advisable to establish beforehand the amount of time which could be expected to be spent administering the system. Some considerations here are:

1. Assisting students who are not computer literate enough to use the system
2. Answering e-mail queries about the system (Spencer & Louw, 2008:121)
3. Troubleshooting bugs in the software
4. Keeping the software updated
5. Setting assignments in the software
6. Selecting specific assignments to do
7. Drawing a report from the software
8. Calculating and assigning marks for exercises or assignments done on the software.

The above considerations come from personal experience as a systems administrator as well as interviews with other practitioners and the University IT personnel over a period of five years.

14.2 IT specialist

It should be clear from the above list that the responsibilities of the systems manager and the IT specialist may overlap. This largely depends on whether or not the system used is stand-alone software, a computer laboratory, or an online educational package. The job of the IT specialist will differ according to the same specifications. His job all but disappears if it is stand-alone software which the students take home. If the software is used in a LAN-based environment in a computer laboratory, he then has the most work, since he has to make sure that all the computers are working without a glitch. LAN-based software is sometimes loaded on a central server which may or may not be administered by the University's central IT administration. The IT specialist then has the job to liaise with them. Since these requirements and internal red tape will differ from university to university, a discussion of this is not warranted for the purposes of this article.

15. IT infrastructure and software costs

The university's IT expert should be in the position to advise on what the situation-specific infrastructure needs are and what the existing infrastructure can handle. While most universities nowadays provide computer access for the students, Internet use is still expensive in South Africa, with South Africa ranking 66th of 114 surveyed countries in The Global

Information Technology Report 2008-2009 (Dutta & Mia, 2010:350). Students are quick to complain if the software they are required to use requires them to go onto the Internet. This is especially true if they have to pay for their Internet use, or if they are required to work on their own at home on the software.

While engineering students are often required to purchase software packages, the purchase of electronic support for languages has apparently not yet caught on. Prescribing a set of software in addition to the normal coursework books will no doubt initially upset a number of students.

In South Africa, one can also not expect all students to have their own computers. Requiring an additional few hundred students to use computers for language learning may place an unbudgeted for strain on computer systems at the University. When planning to use CALL, it is therefore advisable to confer with the local IT infrastructure representatives.

Davies *et al*. (2009) present a lengthy discussion on the costs involved in setting up and managing a multimedia language centre, and they also issue a stern warning regarding copyright fines. If a computerised system requires workbooks to accompany it, the copyright laws must be adhered to. The number of variables to take into account when referring to the cost of a language centre are too numerous to discuss within the constraints of this article. Suffice it to say therefore that the cost of the software itself (as distinct from the complete CALL environment) should obviously be within reasonable limits.

Another important aspect to take into account when considering the infrastructure is quite simply where the students are. If the students are all on one campus, the campus layout and availability of computer laboratories may influence networking. One should also consider whether the students will have access to a lecturer or administrator if they need help. If distance students are somewhere in a remote location, completely cut off from human support such as other students or lecturers, they may be dependent on waiting for e-mail replies or telephone call-backs.

In short, any laboratory setting has the following variables regarding the infrastructure:

1. *Physical resources*: classrooms, laboratory space, libraries and academic offices.
2. *Technological resources*: the provision of up-to-date computers.
3. *Communication*: the management of information and its transmission to all involved in the learning process.
4. *Human resources*: staff who are trained to teach students and eager to adopt new methods and technologies. Significant technical support not only for the maintenance of hardware but the development of teaching materials is also required.
5. *Pedagogical strategies*: teaching strategies need to be drawn up and related to the delivery of the curriculum.
6. *Cultural context*: the approach to learning adopted by staff and students.
   (Barr and Gillespie, 2003:69)

16. Budget

CALL is expensive and is not just a way to get by with less staff. Instead, it should be seen as a way to multiply the effectiveness of the available staff, even if that means paying extra initially.

A decision should be taken beforehand on how much to spend although it is important to remain flexible in one's expectations. Different budget considerations are required for different types of software. Students are not likely to buy software costing more than a few hundred rand, while a server-based software package can easily cost hundreds of thousands of rand.

In addition, server-based software and Internet-based software may require the license to be renewed annually. It is necessary to establish how the rate increase will be calculated in order to avoid later problems. For example, it is possible that a good introductory price will later be raised to astronomical proportions (see Spencer & Louw, 2008). CD-based software may also require annual (or monthly) updates which (while often free) may require large downloads. Internet cost should therefore be taken into consideration, especially in Africa.

A needs analysis is not something to be taken lightly. The large number of variables and role players make this important step a time and resource-consuming activity.

17. Evaluating software

Based on the discussion above, and using the information from Barr and Gillespie (2003) and Bradin (1999), as well as discussions with local IT managers and lecturers, the following guidelines are provided to evaluate software. Just as a needs analysis is an extensive process, exploring the software can be just as daunting a task. Bradin (1999) proposed an evaluation system to determine what software to use and how. In his system, exploring the software is a two-step process involving feasibility and quality (Bradin, 1999:162).

Feasibility considerations according to Bradin are:

1. Will the software run on the specific available computer platform?
2. Do teachers and students know how to use the specific platform?
3. Will the software run on your network? If the software crashes on one workstation, can the program be restarted without interfering with the rest of the network?
4. Can the software be made available to many students? Can it be installed on a Web-server or even taken home to be installed on personal computers of the students?
5. Does the software require Internet access? Some programs offer interactive lessons via the Internet. This requires a very fast Internet connection and lots of RAM. (While this consideration may have been relevant in 1999, it is most probably not relevant in 2009.)
6. Can you afford it?

Most of the above feasibility considerations actually form part of the needs analysis. As far as the quality of the program is concerned, Bradin (1999:164-165) mentions three specific areas of consideration:

1. Content
2. Format
3. Operation

17.1 Content

Bradin proposes the following considerations regarding content:

1. What is the goal of the software?
2. Is the level appropriate?

3. Is the content accurate and up to date? Has it been proofread carefully?
4. Is the material culturally appropriate?
5. Does the software accommodate the students' learning styles and preferences?
6. Is the software interesting?
7. How flexible is the software? It could be necessary to use more than one type of software to accommodate different learning styles.

The considerations regarding content are very much applicable to the evaluation of the contents of books, although CALL also has many more variables.

17.2 Format

Books, websites and software have increasingly more impressive and colourful layouts and graphics. However, an evaluator should be wary of smoke and mirrors and pay attention to the following aspects:

1. Is the interface consistent?
2. Is the screen display effective?
    a. Is text size sufficient?
    b. Are colours distracting or do they add to the attractiveness of the screen?
    c. Is the quality of graphics sufficient that they are clear?
    d. Do the pictures and graphics add to the pedagogical effectiveness of the program or are they just gimmicks?
3. Are the motivational devices effective?
    a. Can the sound be disabled?

The format and layout of any software should above all be functional.

17.3 Operation

Evaluating how easy the software is to operate should not just be left to the evaluator. In this case, the instructor, systems manager and students should be given an opportunity to evaluate how easy the software is to operate. Bradin proposes the following:

1. Is the software easy to use?
2. Are the tasks and directions clear?
3. Can the text and graphics be printed?
4. How much control are the learners allowed?
5. How interactive is the software?
6. Are the quality and degree of the feedback adequate?
    a. Is it appropriate to the age of the intended audience?
    b. Is it immediate?
    c. Do correct answers also get feedback?
7. How good is the HELP file?
8. What kinds of records does the software keep?
    a. Can the records be printed?

When software is being tested, some of the students should be on the test panel. It is also possible that software producers or companies will only direct the customer to their model schools (Bradin, 1999:172), which may necessitate an individual investigation to find additional (possibly

negative) information. Tsiriga and Virvou (2004:412) also support the idea that the software should be tested empirically by quantitative and qualitative means, on real students. Regrettably, the time, money and personnel are not always available to adhere to this recommendation.

18. Writing a program evaluation rubric

Based on the information in the above discussion, a rubric was created to evaluate available CALL software packages with the specific aim of assisting students in their writing. The original version of the rubric was created in 2003 to find software for the Potchefstroom Campus of the North-West University.

The rubric, which has been adapted over time, is presented below. A discussion follows on how such a rubric can be used to evaluate how a new CALL tool which was developed at the North-West University measures up to international recommendations for CALL.

The column marked "status" indicates the relevant importance of the feature. A "very important" feature has 10 marks allocated to it and the evaluator must then assign a mark out of 10 to the feature. Less important features are awarded a maximum of 5 marks and the evaluator likewise needs to assign a mark to it. The criteria are also explained in enough detail that different people will know what is expected of the specific variables.

Evaluating software is a tedious and time-consuming job. The evaluation rubric is not exhaustive but it should provide a good starting point for somebody to evaluate CALL software.

SOFTWARE NAME: .....

NETWORK REQUIREMENTS

CRITERIA	DEFINITION AND NOTES	STATUS	MARK
1. Works on my available operating system	Which operating system are you using? Some software will not work on the latest operating system. If you cannot load the software, you cannot continue with the evaluation.	Very important. If the program cannot work on your system, you obviously cannot continue the evaluation.	…/10
2. Web based	Can be loaded on a central server. For our purposes, we needed the software to run from a central server. For other purposes, it may not be important, but it may be more important that the software can be taken home and installed on a personal computer.	Very important	…/10
3. Good support	A good HELP file or good online support is essential. A local (South African) distributor for the software is an added bonus. Online support is less optimal than a built-in HELP file.	Very important	…/10
4. Budget	Cheaper than Rx for individual packages (take-home packages) or cheaper than Rz for server-based or Internet-based software.	Very important	…/10

5. Upgrading	Preferably free (take-home packages) Fixed yearly rate increase (server-based or Internet-based software)	Very important	…/10
6. Autonomous	Can students access and work on the program unsupervised?	Advantage	…/5
Total for **Network Requirements**			

CONTENT (Pedagogy)

CRITERIA	DEFINITION	STATUS	MARK
1. Software outcome	*Note: As mentioned above, a complete discussion of pedagogy would not fit in the scope of this article. The reader is advised to list his or her own pedagogical requirements here, but must be sure to be as clear as possible. Two examples are provided below.*		
	Example 1 – General vocabulary: The software should assist students in acquiring new general vocabulary words. It should indicate usage in everyday life, the applicable register, and should "sound" the pronunciation.	Very important	…/**10**
	Example 2 – Vocabulary exercises: the system should provide diverse types of vocabulary exercises. It should not give the same exercises to all the students.	Advantage	…/**5**
2. Level	Advanced L2	Very important	…/10
3. Accuracy of content	Up to date	Very important	…/10
	Error free	Very important	…/10
4. Culturally appropriate	Many software programs are American in content and the topics and discussions are unknown or strange to South African students.	Advantage	…/5
5. Interesting	It is difficult to establish what students will experience as interesting; however, you are not looking for a textbook on a screen. If the software fails to make use of the available resources provided by a computer, it fails to make use of the pedagogical possibilities of the medium and as such may be considered not as well planned as one would hope.	Very important	…/10
6. Authorable	Depending on the outcomes of the CALL-system, it may be necessary for the lecturer to change some parts of the program content.	Advantage	…/5
7. Graphics	Do they add to the pedagogical effectiveness?	Advantage	…/5

8. Motivational devices present?	Does the program have devices to motivate students to do better? Examples include games, token "trophies" or triumphant sounds.	Advantage	…/5
9. Workshop	Can the program be linked so that learners can use it in a workshop?	Advantage	…/5
Total for **Content**			

NAVIGATION

CRITERIA	DEFINITION	STATUS	MARK
1. Directions clear	Are the instructions given to the students easy to understand?	Very important	…/10
2. Printable	Can a student print out a piece of work?	Advantage	…/5
3. Interactive	Are the students required to physically do something?	Very important	…/10
4. Feedback	Is it appropriate to the age and level?	Very important	…/10
	Immediate?	Advantage	…/5
	Do correct answers also get feedback?	Advantage	…/5
	Do wrong answers also get feedback?	Advantage	…/5
5. HELP file	Available	Very important	…/10
	Help on program issues	Very important	…/10
	Help on content issues	Advantage	…/5
6. Keep records	Store data	Very important	…/10
7. Print of records	Are stored records printable or only available digitally inside the program?	Advantage	…/5
Total for **Navigation**			

FORMAT

CRITERIA	DEFINITION	STATUS	MARK
1. Consistent interface	Do the screens look the same?	Advantage	…/5
2. Screen display	Text size sufficient	Very important	…/10
	Colours: adding value	Advantage	…/5
	Graphics clear	Advantage	…/5
Total for **Format**			

ADMINISTRATION

CRITERIA	DEFINITION	STATUS	MARK
1. Sifting	Grading or evaluating students, so that those who are more capable do not waste their time on easy exercises.	Very important	…/10
2. Pacing	Are students forced to work?	Very important	…/10
3. Active engagement	Students should not be able to simply click through all the screens without actively engaging in the activities.	Very important	…/10
4. Controlled by student	Can the student choose the sequence of the exercises?	Status depends on the outcomes of the software.	…/5
5. Controlled by lecturer	Can the lecturer choose the sequence of exercises?	Status depends on the outcomes of the software and the preferences of the lecturers.	…/5

ADDITIONAL

CRITERIA	DEFINITION	STATUS	MARK
1. Manual for students	Online	Advantage	…/5
	Hard copy / In-program (Printed versions may be more expensive, but some students prefer them, while in-program manuals may be easier to navigate. Consider which is best for your students.)	Very important	…/10
2. Manual for administrator	Online	Advantage	…/5
	Hard copy / In-program	Very important	…/10
3. Manual for teacher	Online	Advantage	…/5
	Hard copy / In-program	Very important	…/10
4. Time spent on the software	A well-informed opinion is not possible if too little time is spent on the software.	Very important	…/10
5. Which activity seemed the most enjoyable?	Questions 34 and 35 are simply intended to test general perceptions of the evaluator. Be vigilant of big differences between lecturer and student perceptions. If the students hate the software, it will not be effective no matter what the lecturer's opinion.		
6. Which activity seemed the most effective?			

19. Can evaluation criteria be used in the creation of CALL?

While the above rubric was initially created with the intention of establishing which is the best software system for a specific purpose, being aware of these requirements make it easier to plan effectively the creation of a new system. Using the evaluation system, it was established that existing software did not fully meet the needs of the NWU students or lecturers, especially with regard to argumentation, and the implementation of the principles of writing across the curriculum, due to the segmented (specialised) nature of many programs. Initial research on feedback (Louw, 2006) proved the viability of using a computerised marking system (MarkWrite, developed by CTexT® at the NWU), but snowballing from the initial research it is now considered appropriate to develop a much larger system than simply a marking system. MarkWrite shows tremendous potential for further development.

MarkWrite currently falls under the category of a tool as it is not intended to be a stand-alone writing coach. Due to space constraints, the whole project cannot be described here, but suffice it to say that MarkWrite is an electronic tool for marking student texts faster and more accurately using standardised feedback. MarkWrite will have two parts in its final form – MarkWrite Marker and MarkWrite Student. In MarkWrite Marker, lecturers mark student texts and send them their feedback as an HTML file. MarkWrite Student will be a teaching tool in which students have to do exercises based on the feedback. MarkWrite Student will also systematically assist students to create their assignments.

In taking the evaluation criteria one by one, the table below illustrates how these design criteria are taken into account during the planning and programming stages of MarkWrite. Scores were not assigned since the purpose of scoring was to get an overall score for different systems to see which answered best to the needs, and MarkWrite is not being compared here to any other specific system as it is custom built.

Also note that not all evaluation criteria are equally applicable to design and a different priority hierarchy will apply to design than to evaluation. For consistency though, the same rubric is presented here as above, illustrating how it can be used as design criteria.

Table 1 Design criteria taken into account during the planning and programming stages of MarkWrite

NETWORK REQUIREMENTS

CRITERIA	DEFINITION AND NOTES	STATUS	APPLICABILITY AND APPLICATION
Works on my available operating system.	Which operating system are you using? Some software will not work on the latest operating system. If you cannot load the software, you cannot continue with the evaluation.	Very important. If the program cannot work on your system, you obviously cannot continue the evaluation.	MarkWrite was designed to work on the latest Windows systems and is therefore not backward compatible.
Web based	As a design criterion, this evaluation criterion differs according to the software purpose.	Very important	The intention with MarkWrite is that it can be used as a stand-alone application, or later integrated into a Web teaching platform such as WebCT or Sakai.
Good support	A good HELP file or good online support is essential. A local (South African) distributor for the software is an added bonus. Online support is less optimal than a built-in help file.	Very important	The developers at CTexT® went to great lengths to ensure that an accurate HELP file is shipped with the product. This includes a video illustrating how the software functions.
Budget	Cheaper than Rx for individual packages (take-home packages) or cheaper than Rz for server-based or Internet-based software.	Very important	Market research needs to be done to determine a fair price for the system. Since it is an own development for use at the NWU, this criterion is less relevant.
Upgrading	Preferably free (take-home packages) Fixed yearly rate increase (server-based or Internet-based software)	Very important	This consideration is dealt with by the marketing team and is not an actual design consideration, but upgrading can be done on a needs-driven basis.
Autonomous	Can students access and work on the program unsupervised?	Advantage	The initial vision of MarkWrite is not supposed to be used by students, but only by markers. This design consideration will be taken into account when the student part is being designed.

CONTENT (Pedagogy)

CRITERIA	DEFINITION (as applicable to MarkWrite)	STATUS	APPLICABILITY AND APPLICATION
Software outcome	The outcomes for a system such as MarkWrite cannot be directly related to the module outcomes of a specific module, since MarkWrite is meant to be used in writing across the curriculum. The outcomes applicable here are instead stipulated in the introduction to this thesis: the feedback provided with the system should work and it should be practically optimised.	Very important	
	Students should have a greater knowledge and awareness of the qualities which make for good paragraphs, introductions and conclusions.	Important	Specific research has been done to test the techniques used in MarkWrite to ensure that students have a greater awareness of the qualities of good paragraphs, introductions, and conclusions. Research has therefore been done to ensure that the design outcomes match the teaching outcomes of the software.
	Students should have a greater awareness of the specific problems and recurring errors in language.		MarkWrite has been designed to count and calculate the number of errors which a single student or a class group make. This helps both the learner and the lecturer to identify recurring errors.
	MarkWrite should adhere to the qualities of good pedagogical feedback.	Very important	Standardised feedback and radio button feedback as utilised in MarkWrite, adheres to good pedagogical practice.

Level	Advanced L2	Important	MarkWrite is adaptable in the sense that the standardised feedback can be tailored to the specific level of the students, but since the intention is to use it in writing across the curriculum, the level needs to be at advanced L2.
Accuracy of content	Up to date	Very important	As no system can be up to date for more than a few days, there had to be commitment to continually develop MarkWrite. This will ensure that it stays up-to-date. During the internal testing phase of the program, user requests were considered and some were built into the system immediately.
	Error free	Very important	Having a program which is error free is virtually impossible, but stringent testing forms part of the design process to ensure as few errors as possible.
Culturally appropriate	Many software programs are American in content and the topics and discussions are unknown or strange to South African students.	Advantage	The standardised feedback in MarkWrite can be adapted to the specific situation, subject or language. The techniques used in MarkWrite have all been tried and tested on students from different cultural backgrounds and of different proficiency levels. The exercises used in MarkWrite will be tested under diverse situations as well.
Interesting	It is difficult to establish what students will experience as interesting; however, you are not looking for a textbook on a screen. If the software fails to make use of the available resources provided by a computer, it fails to make use of the pedagogical possibilities of the medium and as such may be considered not as well planned as one would hope.	Very important	MarkWrite is not a teaching coach and therefore this criterion does not influence the design process.

Authorable	Depending on the outcomes of the CALL system, it may be necessary for the lecturer to change some parts of the program content.	Advantage	As indicated before, some parts of MarkWrite can be adapted to the specific situation. However, much of the testing is done before implementation to reduce the need for subsequent rewriting of pedagogical content.
Graphics	Do they add to the pedagogical effectiveness?	Advantage	MarkWrite does not contain many graphics. It is aimed at being as functional as possible, much like a computer spelling and grammar checker. Much thought has gone into the layout of the page, the positioning of the tools, the text and window size and other functional layout issues.
Motivational devices present?		Advantage	The first version of MarkWrite is simply the teacher version. The student version of MarkWrite will have exercises for the students based on their feedback, and in this case the motivational devices will be designed into the system.
Workshop	Can the program be linked so that learners can use it in a workshop-environment?	Advantage	The initial version of the marker is not intended to be used by learners.

NAVIGATION

CRITERIA	DEFINITION	STATUS	APPLICABILITY AND APPLICATION
Directions clear	Are the instructions given to the students easy to understand?	Very important	The feedback given to the learner is standardised and the intention of the original research was to make it easy to understand (Louw, 2006).
Printable	Can a student print out a piece of work?	Advantage	Learners receive their feedback in HTML format. This is a "website" and the feedback can therefore be printed if necessary.
Interactive	Are the students required to do something physically?	Very important	In the first version of MarkWrite, learners are not compelled to use the feedback. This is due to networking and system constraints. The intention is that a later version (MarkWrite Student) will force students to actively engage with the planning and editing of their texts, as well as with the feedback.
Feedback	Is it appropriate to the age and level?	Very important (The whole MarkWrite system at present is built around the concept of feedback)	This is not applicable to the current version of MarkWrite, but will be applicable to the exercises in MarkWrite Student.
	Immediate?	Advantage	This is not applicable at present.
	Do correct answers also get feedback?	Advantage	This is not applicable at present.
	Do wrong answers also get feedback?	Advantage	This is not applicable at present.
HELP file	Available	Very important	A help file with screen capture videos will be available.
	Help on program issues	Very important	A HELP file with screen capture videos will be available.
	Help on content issues	Advantage	Due to the variety of texts which can be marked with MarkWrite, this is not possible for the system.

Keep records	Store data	Very important	MarkWrite is able to store individual and class records.
Print of records		Advantage	The output file of the records is printable.

FORMAT

CRITERIA	DEFINITION	STATUS	APPLICABILITY AND APPLICATION
Consistent interface	Do the screens look the same?	Advantage	MarkWrite currently has only two screens. The one screen is the marker interface which is always the same and the other is the output HTML file, which is also always the same.
Screen display	Text size sufficient	Very important	The text size can be adjusted at will.
	Colours: adding value	Advantage	Colours are used sparingly in MarkWrite, but when used the colours add value to the feedback. For example, feedback on different categories of errors is shown in different colours.
	Graphics clear	Advantage	Currently, graphics are not used in MarkWrite. It is possible to include graphics later on, but then with a clear pedagogical purpose.

ADMINISTRATION

CRITERIA	DEFINITION	STATUS	APPLICABILITY AND APPLICATION
Sifting	Grading or evaluating students, so that those who are more capable do not waste their time on easy exercises.	Very important	The intention with MarkWrite Student is that learners only receive exercises based on their individual feedback. This will ensure that all exercises are directly applicable to the specific student. This criterion is not applicable to MarkWrite Marker.

Pacing	Are students forced to work?	Very important	This is only applicable to MarkWrite Student and will be taken into account during development.
	Students should not be able to simply click through all the screens without actively engaging in the activities.	Very important	This is only applicable to MarkWrite Student and will be taken into account during development.
Controlled by student	Can the student choose the sequence of the exercises?	Status depends on the outcomes of the software.	This is only applicable to MarkWrite Student and will be taken into account during development.
Controlled by lecturer	Can the lecturer choose the sequence of exercises?	Status depends on the outcomes of the software and the preferences of the lecturer.	This is only applicable to MarkWrite Student and will be taken into account during development.

ADDITIONAL

CRITERIA	DEFINITION	STATUS	APPLICABILITY AND APPLICATION
Manual for students	Online	Advantage	
	Hard copy / In-program	Very important	MarkWrite's manuals are incorporated into the HELP files. At present the help file is only necessary for the lecturers or marking assistants who use it to mark, since the students will simply receive their HTML files via their e-mails as attachments, or via the web-based learning platform.
Manual for administrator	Online	Advantage	Currently MarkWrite is a stand-alone application. Once it is incorporated into a system such as Sakai, an administrator's manual will become necessary.
	Hard copy / In-program	Very important	See above.
Manual for lecturer	Online	Advantage	
	Hard copy / In-program	Very important	This is included in the help file.

Time spent on the software	A well-informed opinion is not possible if too little time is spent on the software.	Very important	This criterion has to do with the evaluation of the software. However, the amount of research time spent in development can apply here. The development of the software is a long process. Since it is not simply a series of drill-type exercises, much time has been spent on research and more is required to ensure a good product. Once the MarkWrite Student part of the software is being developed, research will be necessary to ensure high-quality, level-appropriate exercises adhering to the qualities of effective pedagogy and CALL.
Which activity seemed the most enjoyable?	This question and the one below are simply intended to test the general perceptions of the evaluator. Be vigilant of big differences between lecturer and student perceptions. If the students hate the software, it will not be effective no matter what the lecturer's opinion.		It is uncertain how exercises and feedback can be made enjoyable for students. At present, MarkWrite Marker is simply an advanced marking tool and this criterion applies even more so to MarkWrite Student
Which activity seemed the most effective?			The practice of providing standardised feedback has been proven to be effective. See Louw, 2006.

20. Conclusion

Integrating CALL into a language curriculum is not a decision to be taken lightly. It requires all parties involved to be educated about what is possible with CALL, will be aware that it is not just one person's job, and will take it seriously enough to properly evaluate software to fit the module outcomes. Before the evaluation can be done, certain decisions need to be made, clarified and discussed. Once a suitable software package has been decided upon, the specific modules have to be re-written in order to integrate the software into the everyday teaching. Although this procedure may sound like a very daunting task, computerised language coaches and computerised language tools may in the end save lecturers and students a great deal of time. A cost-benefit analysis may prove that language learning software is still worth the trouble, but it is an absolute necessity to approach the process in a structured, well-thought-out manner.

The author has attempted to show that (as is the case with outcomes-based education) it is advisable to keep the final evaluation criteria and good pedagogical principles in mind when planning and creating a CALL system. It has been illustrated how many different variables and role players have to be taken into account during the evaluation of a CALL software package. It has also been illustrated that the same evaluation rubric can also be used effectively as a guiding principle for the design of a new system.

The rubric is in no sense hierarchical in nature, but by far the most important aspect of evaluation (which is emphasised in many of the consulted sources) is the fact that effective pedagogical practice is of paramount importance. No amount of features will compensate for poor pedagogical practice. Since MarkWrite is first and foremost a tool to provide effective (pedagogical) feedback on student writing, it is therefore vitally important to establish exactly what constitutes the most effective feedback on student writing.

Creating a CALL system or CALL tool is a daunting task requiring many hours of research and a lot of money. It may be more expensive to create a CALL system than to write a book, and in this case one should question if the benefits of using the computer will eventually really outweigh the costs. The author is of the opinion that the MarkWrite project, although it is still in its infancy, will meet the requirements for effective CALL, and that the numerous further development possibilities of the system will prove to outweigh the cost of development by far.

BIBLIOGRAPHY

Arnett, J. 2009. The way we write now. *Adults Learning* (20)5:27

Barr, J.D. & Gillespie, J.H. 2003. Creating a computer-based language learning environment. *ReCALL* 15 (1): 68-78.

Berg, A.S. & Pretorius, R.S. 2003. Tsenang!: An interactive multimedia programme for learning Beginner Setswana. *Journal for Language Teaching*. 37(1): 1-12.

Bishop, A.L. 1999. Setting policy for the evaluation of the CALL environment. (In Egbert, J. & Hanson-Smith, E. eds. CALL environments – research, practice and critical issues. *TESOL*.)

Bradin, C. 1999. CALL issues: instructional aspects of software evaluation. (In Egbert, J. & Hanson-Smith, E. eds. CALL environments – research, practice and critical issues. *TESOL*.)

Buell, J. 1999. Resources for CALL. (In Egbert, J. & Hanson-Smith, E. eds. CALL environments – research, practice and critical issues. *TESOL*.)

Cowan, R., Choi, H.E. & Kim, D.H. Four questions for error diagnosis and correction in CALL. *CALICO Journal* 20(3): 451-463.

Chappelle, C.A. 1999. Theory and research; investigation of "authentic" language learning tasks. (In Egbert, J. & Hanson-Smith, E. eds. CALL environments – research, practice and critical issues. *TESOL*.)

Chapelle, C.A. 2003. English language learning and technology. Amsterdam: John Benjamins Publishing Company.

Chen, J., Belkada, S., & Okamoto, T. 2003. How a web-based course facilitates acquisition of English for academic purposes. *Language learning and technology* (8)2: 33-49.

Davies, G. & Hewer, S. 2009. Introduction to new technologies and how they can contribute to language

learning and teaching. Module 1.1. In Davies, G. ed. *Information and communications technology for language teachers* (ICT4LT), Slough, Thames Valley University [Online]. Available from: http://www.ict4lt.org/en/en_mod1-1.htm [Accessed 16 September 2009].

Davies, G., Hamilton, R., Weideman, B., Gabel, S., Legenhausen, L., Meus, V., & Myers, S. 2009. Managing a multimedia language centre. Module 3.1. In Davies, G. ed. *Information and communications technology for language teachers* (ICT4LT), Slough, Thames Valley University (Online). Available from: http://www.ict4lt.org/en/en_mod3-1.htm Accessed 4 August 2009.

Dutta, S. and Mia, I. (eds.) 2010. The Global Information Technology Report 2008–2009 – Mobility in a Networked World. Geneva: SRO-Kundig.

Egbert, J., Chao, C. & Hanson-Smith, E. 1999. Computer-enhanced language learning environments: an overview. (In Egbert, J. & Hanson-Smith, E. eds. CALL environments – research, practice and critical issues. *TESOL*.)

Fitze, M. 2006. Discourse and participation in ESL face-to-face and written electronic conferences. *Language learning and technology* 10(1). January: 67-86.

Granger, S. 2003. Error-tagged learner corpora and CALL: a promising synergy. *CALICO Journal* 20(3): 465-480.

Hellar, I. 2005. Learner Experiences and CALL-Tool Usability — Evaluating the Chemnitz InternetGrammar. *Computer Assisted Language Learning* 18(1&2): 119-142.

Hémard, D. 2006. Design issues related to the evaluation of learner–computer interaction in a web-based environment: activities v. tasks. *Computer Assisted Language Learning* 19(2&3): 261 – 276.

Lasagabaster, D & Sierra, J.M. 2003. Students' evaluation of CALL software programs. *Educational Media International*. 40:3/4: 293-304.

Levy, M. 1997. Theory-driven CALL and the development process. *Computer Assisted Language Learning* 10(1): 41-56.

Louw, H. 2006. *Standardising written feedback on L2 student writing.* Potchefstroom: North-West University. (MA dissertation).

Louw, H. 2007. Moving to more than editing: standardised feedback in practice. *Ensovoort*, 11(2): 83-104.

Louw, H. 2008. The effectiveness of standardised feedback when L2 students revise writing. *Language Matters*, 39(1): 88-109.

Louw, H. 2009. Moving to more than editing – a checklist for effective feedback. *Journal for Language Teaching* 43(2).

Murray, L. & Barnes, A. 1998. Beyond the "wow" factor: evaluating multimedia language learning software from a pedagogical viewpoint. *System* 26: 249-259.

Ngu, B.H. & Rethinasamy, S. 2006. Evaluating a CALL software on the learning of English prepositions. *Computers & Education* 47: 41–55.

Peyton, J.K. 1999. Theory and research: interaction via computers. (In Egbert, J. & Hanson-Smith, E. eds. CALL environments – research, practice and critical issues. *TESOL*).

Plester, B., Wood, C. & Bell, V. 2008. Literacy, 42 (3). November: 137-144,

Sivert, S. & Egbert, J. 1999. CALL issues: building a computer-enhanced language classroom. (In Egbert, J. & Hanson-Smith, E. eds. CALL environments – research, practice and critical issues. *TESOL*).

Spencer, B. & Louw, H. 2008. Theory versus implementation: First-World technology in a Third-World teaching context. *Language Matters*, 39(1): 110-126.

Tsiriga, V. & Virvou, M. 2004. Evaluating the intelligent features of a web-based intelligent language learning system. In*ternational Journal on Artificial Intelligence Tools*, (13)2: 411-425.

Wachman, R. 1999. Classroom practice: autonomy through authoring software. (In Egbert, J. & Hanson-Smith, E. eds. CALL environments – research, practice and critical issues. *TESOL*).

Wang, X. 2006. Evaluating a Web-based listening programme for Chinese University non-English majors. US China Education Review (3)5: 74-76

Yuan, Y. 2003. The use of chat rooms in an ESL setting. *Computers and composition* (20): 194-206.

ABOUT THE AUTHOR

Henk Louw
North-West University
Potchefstroom Campus
Email: henk.louw@nwu.ac.za
Tel: 018 299 1049