Shared Identities: Our Interweaving Threads

Portable applications: Authenticity on a stick?

Myles Grogan
Momoyama Gakuin Daigaku

Reference data:
Grogan, M. (2009). Portable applications – authenticity on a stick? In A. M. Stoke (Ed.), *JALT2008 conference proceedings*. Tokyo: JALT.

Portable applications are a type of software that can run from external memory, such as a flash key. Such software can be used to create a computer environment in English, regardless of the language of the operating system. This ability affords students a chance to interact with an English environment in order to perform practical tasks. After a brief look at the pedagogical underpinnings of using such software in the classroom, three such tasks are outlined in order of complexity, followed by specific problems experienced with the tasks, and some of the classroom management techniques used to overcome them. The paper aims to show how the use of portable applications helps students acquire concrete and worthwhile skills, and helps them become invested in their class through acquiring skills other than English in an English environment.

ポータブルアプリケーションは、フラッシュメモリなどの外部メモリから利用することの出来るソフトウェアです。このようなソフトウェアを使えば、コンピュータのOSの言語が何であっても、英語の環境を作り上げることができます。このソフトウェアの特性を活かし、学生達に英語の環境の中で実用的な課題をこなす機会を与えることが出来ます。最初に、教室内でこのようなソフトウェアを利用することの教育的な基盤にふれた後、実際におこなった3つのタスクを紹介します。続いて、それぞれのタスクを行った際に起こった問題点と、それらの問題点を克服するためのクラスルームマネジメントの技術を紹介します。ポータブルアプリケーションの利用によって、英語の環境内で英語のみならず、実際的で役に立つ技術や知識を身につける手助けが出来る、ということをこの論文の中で紹介させていただきます。

Introduction: What is a portable application, and why use it in the classroom?

"Portable applications" work in English regardless of the language of the operating system, and add an extra depth to any computing activity in the language classroom. The following article concerns itself with practical ways to use portable applications so that students can interact individually or in teams with an all-English computer environment. Students may use their native language to cooperate and discover meaning in that environment, rather than having to roleplay the English

part in an artificial "English-only" situation with Japanese speakers. After discussing the pedagogical background, some details of the original setting will be given by the writer. The paper then outlines some points to remember before starting, and next the experiences with the three tasks used in actual language classrooms. This will be followed by some general guidelines and practical considerations for the computer classroom that make such activities flow more smoothly.

An application is considered portable if it can run on any computer without being installed on the hard disk. Unlike most modern commercial software, early word-processing and game software used to be "executed" (or run) from a floppy disk. Now, software developers have found that, by revisiting this approach, a user can carry their software with them on a flash key, CD ROM, or other memory device. The result is that a user carries and uses the same software environment, with the same settings, on any computer the user may use. This frees users from needing to install software, a process that may require authorization at a level higher than that usually granted to students or teachers in a typical academic setting.

Such software is easily and freely available in a variety of languages, and can easily be used in English on non-English computer systems. In school settings where computers are a much competed-for resource, this portability guarantees the software is available wherever a class is scheduled. Indeed, it can (and has been) run on laptops in a normal classroom. The variety of software available means that many diverse tasks can be assigned, from editing photos to recording a podcast, all through the medium of English. Because the software is portable, students can recreate the experience outside the classroom, at home or even in an Internet cafe if they so choose.

The software I have used in my own classroom is freely available under some form of public license, typically the GNU General Public License (the GNU Project, 2007). The software is developed to be "free," in the sense that ownership of the software allows one to change and distribute the software without needing to worry about copyright and licensing. As a result, unlike many commercial applications, users are free to pass on the software (for example, teacher to student), or even change the software in some way. While teachers may not have the skills to program such changes themselves, the software is largely community based. If you are involved in the community, you may be able to suggest changes and a volunteer may do it for you. Moodle is licensed under this kind of agreement, and this community-based approach has contributed greatly to its success (Moodle Website, n.d., Paragraph 3).

Setting

My setting is a mid-level university in which English is compulsory. In a classroom survey, when asked in class why they are studying English and how it will serve them in the future, many students have said that it is because it is compulsory, and they need it to graduate. Students generally have two 90-minute classes a week, one with a native English teacher, and one with a Japanese English teacher. Curriculum guidelines exist, but are recommended rather than compulsory, and assessment is far from uniform.

Two computer labs are given over as CALL rooms, but these rooms are not always available. Other computer labs can be used, but differ in the software used to run the "lab" functions (e.g., those allowing teachers to monitor what class members are doing at their individual terminals). Both sets of labs have office suites available, but the operating systems and software in those labs are only available in Japanese. Any tasks done for an English class, even in dedicated CALL classrooms, must be performed in a Japanese environment, and require technical support in Japanese. Thus, in a setting where using English is the objective, the environment is overwhelmingly Japanese. I believe this aspect of the experience to be typical for the IT facilities of a good number of similar institutions.

Pedagogical overview

Computing has an inspirational power with learners. Sugata Mitra, talking about work he did on leaving simple Internet devices for children in India to use, found that children soon taught each other how to use such devices, and that up to 200 computer-based English words entered the children's lexicon (Mitra, 2008). The key seems to be that the children found something interesting and useful. Although the situation does not compare directly with that at my university, I wanted to create an "interesting and useful" situation that would fire the students' enthusiasm in the same way.

The Ministry of Education, Culture, Sports, Science and Technology (MEXT) has a set of guidelines which show the level of computer skills a student should have in order to graduate high school (MEXT, 2003). In general, these guidelines ask students to show a basic knowledge of office software (for example, being able to make a chart or graph), as well as knowledge of the Internet and privacy issues. Much like the level of English required for graduation, however, the guidelines are recommendations. Simple surveys on computer literacy I have conducted in class, in addition to classroom experiences of watching students faced with computer-based tasks, show that these standards are not being reached. Some of my first year students still report to me that my class is the first time they have had to use a computer.

A brief and informal Internet-based survey of my students showed that they are generally active on the Internet, with students listing shopping, online gaming, and social networking as the activities they have performed on the Internet. Other computer-related skills are extremely limited, however. Students also seem to be aware that they should have more knowledge than they do, and (after expressing initial reluctance in some cases) show impressive determination in trying to come to grips with the use of office, audio, or other software they get to deal with in their classes.

Despite an institutional distinction between "native" and "Japanese" English teachers, there is no requirement for native teachers to use only English in their classrooms, nor is there any rule dictating students use only English. Auerbach argues for the ESL environment that the evidence favouring exclusive use of English in the classroom is "neither conclusive nor pedagogically sound" (Auerbach 1993, p. 1). For a monolingual EFL classroom of non-English majors in a compulsory course, this can only be truer. My policy for "English-only" is that this must be an active choice on the

part of the students, who feel they will gain something from their decision.

The purpose of using English software, in preference to the Japanese software available, is to create challenge in the classroom environment that may be of some consequence to their everyday life. In essence, this should allow students to work (with L1 input, if necessary), to facilitate what Auerbach calls "the transition" to English (Auerbach, 1993, p. 5). Many students now seek me out after class to check homework, where before they may have let any questions from class slide. Also, I have found that students see tangible evidence of progress in computing as a result of asking questions. A sense of faith and trust develops between teacher and student. This is important, because some students seem then to transfer this strategy to asking questions about English – an outcome I am particularly pleased with.

In that it "has not been specifically produced for the purposes of language teaching," a computer interface fulfils Nunan's rule of thumb for "authentic" material (Nunan, 1989, p. 54). Although individual tasks may still be interpreted as being for "language teaching" by the students, they still seem to see using portable applications as a genuine challenge. Specifically, many of them feel they learn about computing as well as English, and this appeals to them in terms of job-hunting skills and other external motivators. Finally, in terms of students working together to produce a tangible product in the classroom, we (students and instructor) achieve a degree of authenticity in the social context – people struggling to find their way through an environment to reach a specific goal in English, perhaps with the help of a teammate.

School considerations for school computer environments

Facilities such as computer labs are usually staffed and managed by the Information Technology (IT) department, who have different (and legitimate) priorities to language teachers. In some institutions, the maintenance is outsourced to private companies who may not have a teacher's view of the computer for language learning. This difference can be a source of conflict. Teachers should recall, however, many pieces of installed software can also conflict with each other, requiring differing versions of support software (such as QuickTime, used to play video). It is a challenge for IT administrators to juggle such considerations to try to manage and protect an expensive system. Software that teachers try, and even run smoothly, on a home computer may behave very differently within a school system. Because portable applications do not require installing, however, the IT department may be happy to let teachers use them. At my institution, they even helped with general problems with the software despite this being beyond their remit.

Other alternatives to portable applications are available. Yahoo Groups, YouTube, and Google Documents are all easy to use and apply in the language-learning situation, for example. Authorship, however, requires users to sign a user agreement that involves disclosing some personal information (e.g. name, date of birth, email address). Where a student has a strong objection to accepting such an agreement, either the class cannot proceed or the students' concerns are overruled by the judgement of the teacher. Ethical guidelines on this issue are sparse at present, but sacrificing a student's privacy to make a class go to plan

is an issue that needs careful consideration. In addition to the in-class dynamic, in these times of "cyber-bullying" and other unanticipated consequences of computer use, the concerns of other stakeholders have led to some schools blocking access to such online resources, occasionally without telling teachers who may wish to use these resources. In the past, teachers have only found out that sites are blocked after class has begun. To avoid this frustration, readers may want to ask their school for the student account settings on browsers, as this can help with planning some activities in the computer classroom.

On a related point, the age of majority in Japan is 20, and so asking students younger than this to accept user agreements may have a legal dimension as well as a moral one. In a university environment, the issue of a teacher's responsibility for what students do and are exposed to may be complex. On balance, teachers may want to familiarize themselves with school approaches and any policies relating to computers in the classroom, before deciding a plan of action.

Three tasks from my classroom

New Year's cards: Photo editing software

The simplest and most rewarding project to be described here used the Gnu Image Manipulation Program (GIMP), and took place over two classes. In the first class, students and teacher work through an online tutorial from LearnHawk (see Appendix), cutting out foreground figures and placing them on a blurred background. This creates a dramatic, filtered-looking shot from a simple picture. Getting the effect involves creating a copy of the image so the student was working with two "layers," blurring the back image, and removing the background from the front using the "clear" function.

The idea of layers is complex, and needed some illustration. To help students get the idea, make several copies of a picture, each on different coloured paper, and put them in a clear wallet folder. Then cut out part of the picture on the top layer so students could see the different colour below, illustrating visually the idea of layers.

After working through the tutorial (use a portable English version of the Firefox browser for this), students use their own pictures from the Internet, open them in the GIMP, and do the editing operation themselves. The students have prompts from the English tutorial, and can help each other in a team to repeat the process. Using the tutorial involves skills such as skimming and scanning, in addition to the English skills required to interact with the text.

In the next class, students use the same cutting and layering tools to create New Year's cards. Here the instructor can make examples and source materials for students. During a demonstration, everyone uses the same materials to make similar cards. Find swatches of cloth on the Internet to use as a background (although I told students they can scan cloth, paper, or other materials themselves to make a background for their cards). Use an image of the zodiac animal, for example an ox for 2009. This can be opened as a layer. Then clear the background. Shrink the image to fit on the swatch of cloth. Use the text tool to write a greeting, finishing the card. After completing a common model, students can make their own version and print it.

Presentations introducing teams: A simple presentation software project

Students are divided into groups (I had nine groups in my class), usually with four or five members. As an introductory activity, students are asked to interview each other on a range of topics and to get a photograph of each other. Topics for the interview include those already covered in the previous semester or the topics they will cover before the presentations later in the course. In my case, the topic list included hobbies, favourite music, and movies, and what their group members had done over the summer. The interview topics tasks are relatively simple in order to allow simplicity in the computing task.

Following the interview, students are told to prepare a presentation between two and three minutes long introducing one of their teammates. The information from the interview is formatted to fit onto five slides chosen from a template on Impress, the presentation software from the OpenOffice.org suite. Each of the interview topics is presented in a slide, requiring a title and two bullet points. Students are required to find pictures for each slide. Any pictures from the Internet or from the person they are introducing are acceptable, but a picture of the person being introduced has to be included. If the student does not have a picture they wish to share, I ask them to take pictures with their cell phones. (This in itself was fun for many of the students.)

Assessment is based on whether the slides are completed (required for a C), and the extent to which students simply read their slides or add extra information to the presentation. Students adding two extra spoken points to each written bullet point and keeping eye contact with their audience receive an A. Students evaluate each other, but presentations are recorded on video in case of appeals. A review of a random sample of videos shows that, although students in my class were somewhat generous, their assessments were generally acceptable.

Book reports for extensive reading: Word processing

Book reports were the first practical application of a portable application I used, and they have been the most successful in my classes. It is also the most complex in implementation, requiring students to demonstrate many subskills in formatting and layout. It is therefore the final project introduced.

Students are assigned a minimum number of books and specific number of pages from graded readers to read in a semester. This was varied according to the group, but ranged from as little as 125 pages and four books for low proficiency students, to as many as 350 pages and eight books for more advanced students. Students have to reach both the minimum number of books and the minimum number of pages to be awarded a passing grade.

Students record the titles and the number of pages they read. For each book the students read, they write a 50-75 word summary of the story in Writer, the word-processing arm of the OpenOffice.org suite. This is extensively modelled for the students, so they were able to do it without major complications. The reports are checked by a rubric. If all the reports that a student prepares successfully meet a criterion, they are awarded points for that criterion. For example, putting their name on the left and their student

number on the right hand side of a header might get students two out of the 20 points available for the assignment. Appropriate font and bolding are another two points. I ask students to include a picture from the Internet, either of the book cover or an associated image. To get the points, the picture has to be aligned to the left or the right of the paragraph. This requires students not only copy and paste, but also adjust the layout of the picture.

Peers carry out the final grading process. These peers are responsible for checking that the work is complete, and for signing the work to say they have done so. Because this peer editing is initially difficult, and requires multiple passes, I prepare students by insisting that each report is checked and signed by other peers in the classes leading up to the final assessment. Peers who have missed points can be called on to explain why they missed a point to the person whose grade depends on their editing. After two or three rounds of checking, students get the idea that their accuracy in checking will affect their classmates and take more care.

Some guidelines based on experience

Working in the computer lab involves a number of specific aspects of classroom management worthy of attention and care. Tasks that run well on an office computer occasionally run differently or even fail in the lab. As an example, my image manipulation software (the GIMP) ran itself in English on my computer, but it ran in Japanese on the students' computer. A small change to one of the files in the software resolved the issue (thanks to help from our IT department). In addition, while commonly used functions are easy and familiar on portable applications, less often-used functions (such as printing) may look very different to an unfamiliar user. The demands of the classroom can magnify small problems and deliver unwelcome upsets. If possible, it is worth trying software on a student "test" account.

Rather than asking students to download software from a website, it is preferable for them to copy the software from a central school common folder. On my school system, copying and pasting the OpenOffice.org software (approximately 250 MB) to the desktop takes about 3 minutes and longer to a USB device. Smaller software, such as the GIMP or Audacity, copies relatively quickly, taking about 10 seconds. The downloading is a crucial, and occasionally confusing, part of the class, but choral drilling of instructions makes this part easier to follow for students, and can be integrated into the language learning. This is a form of shadowing, and is a useful practice in the computer lab. If students get stuck or lose track, a neighbour may be able to repeat the instructions.

Save work early and often. Some students will inevitably open the software from the common folder. Usually, the software will open and function normally, even if it is running on two computers at once. However, when students then try to save the work, their software will crash and it will be lost. A simple way to avoid this is to save the documents early on, perhaps asking students to write their name or the title of the project. Hopefully, this will set off the crash early and you can overcome the problem without more discouraging losses.

In the computer lab, groupwork is particularly effective. It is easier to teach 9 groups of 4 students than to teach 36 individual students. In small teams, students generally help

each other. As more and more teams complete tasks and produce a product, those teams that are behind ask for help. If all members of a team are stuck on a technology point, then the issue may need to be addressed as a class.

Another useful practice is to agree on a signal that students have finished a task, for example moving their name cards in an easily viewable place (such as on top of the monitor or CPU). Using name cards helps to address students by name, so a teacher can then check on the student directly, or ask another student or team member to help them.

Where tasks have been graded, achieving the technology goal (such as making presentation slides) is a good basis for a C, and linguistic performance can be used for higher grades. Thus, even where students have discussed the task in their L1, successfully navigating the program shows that students can interact with the English environment. This creates an initial success, which students want to build on in language tasks. Finally, because technology tasks often boil down to simple yes/no statements like "Is the document in 12 point Century Schoolbook font?" the door is also open for peer evaluation.

Conclusions

As a classroom practitioner I have used the traits that Barker (2003) says all teachers fall back on: "instinct, experience, and common sense." The use of portable applications in the classroom has required a considerable amount of each of these facets. Activating a new area of learning, with tangible value for the students' future, has in some ways remade the English classroom. Introducing new technology into the classroom seemed to neutralize some of the affective barriers students bring to class, because it steps outside of regular "English" activities. As a result, students seem to be asking more questions in class, both with regard to technology and language, and seem to be successfully making the "transition" to English.

The greeting cards, presentations, and book report activities described above are more than simply ways for me to engage my students in language tasks. It has been my belief that teaching, as a moral act, requires that we teach something of tangible and concrete value to our students. I believe that raising the level of computer literacy fulfils my personal obligation with regard to providing that value. During my class time with the software, I have seen students cooperate to complete tasks. Teams work together to gain expertise through a maze of English instructions. The team interacts with an English environment and comes away with new confidence, and a small victory in that they did what they set out to do, and perhaps were unable to do before. Whilst portable applications may not be the way every teacher's chooses to approach adding this value to classes, it has worked well in my current environment, and I believe it is a viable approach for the classroom generally.

Myles Grogan teaches at Momoyama Gakuin Daigaku. His research interests centre on the role of technology in the student experience.

References

Auerbach, E. (1993). Re-examining English only in the ESL classroom. *TESOL Quarterly, 27*(1), 9-32.

Barker, D. (2003) Why English teachers in Japanese need to learn Japanese. *The Language Teacher,* February 2003.

The GNU Project. (2007, June 29) GNU General Public License, Version 3. [Online] Available: <www.gnu.org/copyleft/gpl.html>

Mitra, S. (2008) Can kids teach themselves? TED Feb 2007. [Online] Available: <www.ted.com/index.php/talks/sugata_mitra_shows_how_kids_teach_themselves.html>

高等学校学習指導要領（平成11年3月告示、14年5月、15年4月、15年12月一部改正）－第2章：普通教育に関する各教科－第10節：情報. [Online] Available: <www.mext.go.jp/b_menu/shuppan/sonota/990301/03122603/011.htm>

Moodle Website. (n.d.), Retrieved 2008 November 22 <www.moodle.org>

Nunan, D. (1989). *Designing tasks for the communicative classroom*. Cambridge: Cambridge University Press.

Appendix

Useful Web resources

The Gnu Image Manipulation Program (GIMP) portable version. <portableapps.com/apps/graphics_pictures/gimp_portable>

LearnHawk has a useful tutorial for the GIMP <www.learnhawk.com/elearns/startinggimp/level2-1.html>

<OpenOffice.org> (This office suite uses its Web address as its name.)

<portableapps.com> has more details on portable applications.