ReCALL **16** (1): 41–50. © 2004 Cambridge University Press
DOI: 10.1017/S0958344004000412 Printed in the United Kingdom

Increasing accessibility by pooling digital resources

STEVE CUSHION
Department of Humanities, Arts and Languages, London Metropolitan University, Calcutta House, Old Castle Street, London E1 7NT, UK
(*email: s.cushion@londonmet.ac.uk*)

Abstract

There are now many CALL authoring packages that can create interactive websites and a large number of language teachers are writing materials for the whole range of such packages. Currently, each product stores its data in different formats thus hindering interoperability, pooling of digital resources and moving between software packages based in different technology. The use of Extensible Mark-up Language (XML) for data storage goes a long way to solve this problem and allows for the easy conversion of exercises. Starting from a desire to develop a common format between *Hot Potatoes*, *WELTS* (part of the WELL project) and the *Interactive Language Learning* package from London Metropolitan University, a new version of the *Interactive Language Learning* software, now renamed Guildhall Interactive Software for Multimedia On-line (GISMO), has made such conversion possible. Given the immense resources required to develop the critical mass of material required to make online CALL relevant to an individual's teaching practice, such a common approach is required to facilitate the pooling of resources. Should a bureaucratic or financial decision in an institution result in a change of software, teachers need to be able to easily convert their legacy material. XML technology can facilitate interoperability, thereby increasing potential accessibility by allowing teachers and students to have the use of a greater amount of pedagogical material. It is further proposed, using these developments, to create a large pool of exercises for practice and assessment that is independent of the delivery approach employed. This will obviate the need for teachers to keep reproducing basic language learning material and allow for the expansion of online CALL into more imaginative areas. This possibility introduces the question of standards within XML and whether it is necessary to further specify how the material is stored, perhaps using a standard such as the 'IMS Question & Test Interoperability Specification' or whether XML is a sufficient standard in itself.

1 Pooling CALL resources

It is a daunting task for any teacher to start producing enough CALL material to create that critical mass which is necessary to significantly affect the teaching process. While being unaware of any quantitative research, there is much anecdotal evidence of teachers starting to produce material for a course and then abandoning the project after realising the resource implications. This is particularly demoralising given that much of the

https://doi.org/10.1017/S0958344004000412 Published online by Cambridge University Press

basic work of writing interactive exercises has already been done and there is enough in existence to cover the basic levels of the commonly taught languages. For instance, the London Metropolitan University GISMO project alone has covered the GCSE syllabus for French, Spanish, German and Arabic and made it publicly available (Hémard, 2003).

The problem is compounded by the fact that technology has changed while this writing process has been going on and there is a wealth of material that is pedagogically sound, but which is unusable, being in the wrong format or relying on outdated technology.

There is clearly a need for a pool of interactive exercises from which teachers may draw examples that suit their teaching needs and the progression of their courses. They could then write a small amount of extra material to make any particular points they may wish or to fill perceived gaps. Hopefully they would have the generosity to add this additional material to the common stock. There is no logical reason why such exercises should be tied to one particular authoring package as technological improvements are continually being made and new products appear in the field.

The Subject Centre for Languages, Linguistics and Area Studies, based at Southampton University, has made a valuable contribution to this co-operative spirit by initiating a resources bank which "contains teaching materials in a wide variety of formats which staff have very kindly agreed to disseminate via the Materials Bank so they may be used more widely" (http://www.lang.ltsn.ac.uk/resources/bankcontents.aspx). This valuable source of teaching aids would be even more useful if users could take material that they found pedagogically useful from one format and convert it into their own chosen format and thereby give their work a consistency and seamlessness that would facilitate its use.

2 Converting between authoring packages

CALL authoring packages store their output in files that enable authors to reopen them and correct errors, make changes or add additional examples and then resave. Each authoring package has developed the format for these files on a different basis consistent with the development of the software package itself and very few have given much consideration to questions of interoperability.

One can ask why the question of interoperability is practically important. To give a concrete example, let us look at the different functionality available to authors using either *Hot Potatoes* or the London Metropolitan University software. Both have a "Fill the gap" exercise which, despite different interface considerations, functions in a broadly similar fashion. However, the London Metropolitan University approach allows the easy recording of the spoken voice to accompany the exercise (Hémard & Cushion, 2002). The *Hot Potatoes* approach allows the author to include feedback in the exercise in such a way as to give advice that may explain to the learner why an error was made. Evaluation of CALL material for Arabic has shown that the use of the same material in different contexts and to practise different language skills is beneficial to the language learning process (Cushion & Hémard, 2003). A teacher authoring some exercises to reinforce a particular point could therefore use the London Metropolitan University package to practise skills associated with listening, while using *Hot Potatoes* to examine possible grammatical errors in greater depth. The resulting two exercises can be mixed seamlessly in the same website. If the author can convert between software packages

https://doi.org/10.1017/S0958344004000412 Published online by Cambridge University Press

with ease, it is possible to quickly reuse material in different environments that appear appropriate to the situation without the necessity for a laborious retyping process. Colleagues at Dublin City University and Goldsmiths College, University of London have conducted experiments with just such a mixed use of the two packages.

Several years ago, considerable work was lost by colleagues who had worked diligently to produce CALL material using the *Guide* package and found that their work was lost when the institution was no longer prepared to finance the necessary licence. The fact that painstaking work to author language learning exercises can be lost through technological changes, financial considerations or external decisions, makes work to address this question seem useful.

The development of XML (http://www.w3.org/XML/) as a data storage standard opens the possibility of storing data in a machine readable form that is easily convertible for use on different systems. XML enables the sharing of data along with information about that data in a platform independent manner. Data is stored in text files with the information about that data stored in tags that resemble HTML formatting tags. Authors develop their own specific tags to describe the information contained in their documents. This means that, while developers can store information in a form that is applicable to their own needs, the fact that the data is text based and platform independent enables another potential user of the same data to write a simple program to convert the data to their specific computational needs. This vastly increases the possibility of data sharing and obviates the dilemma of either being tied into old technology or losing previous work. It also mitigates the problems caused when bureaucratic decisions in educational institutions change platforms or fail to fund the renewal of licences.

The use of such a platform independent approach to data storage has the advantage of separating the pedagogical from the technical. While there is obviously a dialectical relationship between the two, each has its own pace of development and this decoupling confirms the ability to assert the primacy of educational requirements over the technology. In asserting this relationship it is necessary to take issue with Armitage and Bowerman (2002) when they say in opposition to Nicholson and Mulhern (2001):

> "Certainly, we feel that to say 'any ICT application worthy of the name' must be pedagogically 'driven' is too strong a statement."

and further on the same page

> "the pedagogical approach potentially conflicts with another recognised principle, that the application must present ideas in a way that is not possible using more conventional materials"

Such statements conflict with the basic principles of software development that place the requirements of the user above the interests of the developer. In the case of CALL software the users whose interests are paramount are teachers and students and if the software is not driven by pedagogy then it must be driven by the technology. These statements appear to have their origins in a fear of not extending the possibilities opened up by the use of computers in education. In fact, the dialectical mentioned above is a much more appropriate mental model for ensuring that CALL lives up to its

https://doi.org/10.1017/S0958344004000412 Published online by Cambridge University Press

potential. Classic software design theory starts from the position that it is the duty of the software engineer to meet the requirements of the client and the users. In the process of doing this, new possibilities arise and the user requirements change to reflect this, thus precipitating another turn around the development lifecycle. To allow, however, these possibilities to dictate the development direction of the software is to make a classic software development error and leads inevitably to the development process spiralling out of control.

Lewis and Chase (2003: 4) list seven 'quality characteristics' for software development: correctness, reliability, robustness, usability, maintainability, reusability, portability and efficiency. They define correctness as "the degree to which software adheres to its specific requirements". Of the other characteristics to which they refer, we shall see how the use of XML can lead to significant advances in the fields of usability, maintainability, reusability and portability. The idea of the separation of data from the computational process necessary for its delivery is a commonplace in software engineering theory; the practical application of this theory to CALL software practice is however much rarer.

The developers of the *Interactive Language Learning* authoring package (Hémard & Cushion, 2002) from London Metropolitan University decided to convert their data storage to the use of XML. This was done following discussions with the producers of the *Hot Potatoes* authoring package (http://web.uvic.ca/hrd/halfbaked/) as, despite using different technology and having a different interface, there was much common ground with their underlying approach. As stated earlier, colleagues at Dublin City University and Goldsmiths College had also done some work producing language learning websites that combined material produced with both packages. This seemed the obvious place to start, therefore, in an investigation of practical interoperability. As stated in the introduction, it had also been the intention to work with the WELTS project (WELL Language Testing System) from Liverpool John Moores University. Unfortunately, they have suffered a major systems problem and collaboration had to be suspended while it is hopefully resolved.

3 File format

The first step was an investigation of the form the *Hot Potatoes* mark-up tags take. *Hot Potatoes* has six basic exercise types of which five function on similar basic lines to comparable exercises in the London Metropolitan University authoring package. It was decided to use the same format for these exercises in order to profit from their previous development experience and to make conversion as easy as possible. The *Hot Potatoes* package has prioritised ease of understanding in their tag names to make the XML files understandable and user friendly themselves. While the overwhelming majority of users will never open an XML file, the fact that this can be done using a web browser makes it sensible that anyone with cause to look at the encoded data should be able to easily read it. Thus a simple gap fill exercise with the title "Fill the gap with the correct part of the verb in brackets" and the first gap to be filled as "[llamarse] Alicia Fuentes" and expecting the answer "Me llamo" would be stored by the following code fragment, which is taken directly from the XML file in which *Hot Potatoes* stored this example.

https://doi.org/10.1017/S0958344004000412 Published online by Cambridge University Press

```
<title>Fill the gap with the correct part of the verb in brackets</title>
<gap-fill>
      <question-record>
      <question />
      <answer>
             <text>Me llamo</text>
             <correct>1</correct>
             <feedback />
      </answer>
      <clue>llamarse</clue>
      </question-record>
      Alicia Fuentes.
</gap-fill>
```

Hopefully this is intuitive and thereby renders the encoded data intelligible and even hand editable. The only major initial change that was made was that *Hot Potatoes* data files have different file extensions depending on the exercise type and use the extensions mainly for file associations in Windows, so that one can click on the file to open it in the application automatically. We preferred to have a generic file extension '.xml' so that the files can be easily opened in a web browser. This preference comes from the different software architecture of the two packages, *Hot Potatoes* has a separate, effectively stand-alone, application to author each type of exercise, with a separate interface application capable of launching any particular type of exercise generator, whereas the London Metropolitan University package operates from within one unified application. To get round the need to identify the exercise type, we added another data element: `<exercise-type>`. It is one of the considerable advantages of XML that more data elements can be added without making the file illegible to another application reading the data that has no need or expectation of finding that data there. This seems logically obvious, but would not be true of many proprietary file storage formats that would be thrown wildly off course by finding unexpected and unneeded information.

4 Implementation

It has been mentioned that the two packages under consideration use fundamentally different technology to produce their ends. *Hot Potatoes* uses JavaScript while the London Metropolitan University package is written in Java. The similarities of name should not be allowed to give the impression that there is the slightest relationship between the two programming languages. The beauty of XML is that it stores the data as a text file and it is up to the reading application to interpret that data.

The Java programming language has embraced XML from the beginning and an extensive range of XML parsers has been developed for use with Java, a parser being a program that enables the reading, writing and validation of XML data (Whitehead, Friedman-Hill & Veer, 2002). It is therefore a relatively simple programming task to incorporate such a parser in a Java authoring package and to thereby read and write fully validated XML data storage files. The London Metropolitan University authoring package had previously stored its data as text files that merely stored the text as seen on the authoring screen. Thus the simple gap fill exercise discussed above would have been stored as:

https://doi.org/10.1017/S0958344004000412 Published online by Cambridge University Press

[llamarse\Me llamo] Alicia Fuentes.

There was no possibility of storing the instruction, the title or anything else about the exercise. Thus when one reopened an exercise to correct an error or to add extra examples, all the extras such as the instructions had to be retyped. Using the new XML file format, one merely identifies the file containing the exercise and all the necessary additional information is input directly and the authoring package puts itself into the state necessary to save the exercise in the correct form once the user has made the required changes.

The development of the new approach to data storage occurred during the work of creating version 5 of the application which entailed converting to Java 1.3 (the previous version ran on version 1.1) and responding to changes required by the evaluation process (Hémard & Cushion, 2003). The evaluation process had identified the teacher requirement for the inclusion of images in all exercises and of reading texts in multiple choice exercises as well as the student requirement for access to appropriate grammatical information while they are in the process of working on the exercise. Another student requirement was to have a caption under every image to identify it. The software has been amended to permit such requirements while the new file format enabled the storing of the necessary information by the addition of new data elements.

To give an example of these changes, here is the start of the XML file that stores a reading comprehension exercise with an illustration.

```
<data>
  <exercise-type>multiple choice</exercise-type>
  <language>spanish</language>
  <title>Transport in Spain</title>
  <image>ave_sala.gif</image>
  <image-caption>Las Líneas de Alta Velocidad AVE</image-caption>
  <notes>1</notes>
  <reading>
        <include-reading>1</include-reading>
        <reading-text>RENFE cuenta con un amplio servicio nacional e
internacional. En muchas partes del país el servicio es lento y poco eficiente en general. Pero las ciudades principales están conectadas por un
servicio de trenes rápido y moderno. El Talgo, por ejemplo es un tren muy
rápido que alcanza una velocidad de doscientos kilómetros por hora. Tiene
aire acondicionado, pero también es más caro y un billete para el Talgo
cuesta el doble de un billete para los otros trenes. El tren más rápido es
el AVE (Alta Velocidad Española) que alcanza hasta trescientos kilómetros
por hora. Es un tren muy cómodo con cafetería y el precio del billete es
ligeramente superior al del Talgo.
        </reading-text>
  </reading>
  <questions>
        <question-record>
                <question>¿Cómo están conectadas las principales ciudades
                 españolas?</question>
                <must-select-all>1</must-select-all>
                <answer>
                        <text>El tren más rápido es el AVE.</text>
```

https://doi.org/10.1017/S0958344004000412 Published online by Cambridge University Press

```
            <correct>0</correct>
            <feedback />
        </answer>
        <answer>
            <text>Las ciudades españolas están conectadas por
            un amplio servicio de trenes rápido y moderno.</text>
              <correct>1</correct>
              <feedback />
        </answer>
    </question-record>
  </questions>
</data>
```

For data elements representing Boolean values, 1 indicates *true* and 0 indicates *false*. XML stores characters of Unicode 256 and above, by converting to HTML decimal escapes (ampersand + hash + unicode number + semicolon). This permits the storage of work associated with non Latin based languages (Cushion & Hémard, 2003).

To briefly return to the issue raised by Armitage and Bowerman (Armitage & Bowerman, 2002) discussed above, it may appear at first sight that this new functionality was somehow technologically driven, whereas the need had been identified for some time from user walkthrough evaluation (Hémard & Cushion, 2001) and technological change had merely permitted its implementation.

5 File conversion

When it comes to exporting an exercise generated in the London Metropolitan University software to a *Hot Potatoes* exercise, the application has to add a different root element to identify the exercise type, for example `<hotpot-jcloze-file>` for a gap filling exercise and a few additional elements that *Hot Potatoes* expects to see such as `<version>`. The file is then saved as another XML file with the appropriate file extension for the exercise type such as '`.jcl`' for a gap fill. When this is opened in *Hot Potatoes*, all the basic information is there to recreate the exercise as a *Hot Potatoes* exercise. When this is saved from inside that application, additional information particular to *Hot Potatoes* is added without removing the new elements inserted by the London Metropolitan University package. Importing an exercise into the London Metropolitan University software from *Hot Potatoes* is merely the reverse of this process.

Having set out deliberately to create an XML data storage approach based on many similar elements to that used by *Hot Potatoes*, it is hardly surprising that exporting to their software and importing work that had been completed using their software was extremely easy. The only problem encountered was when we started to restrict the functionality of our software to ensure that it could all be exported to *Hot Potatoes*. *Hot Potatoes* has a crossword style of exercise with no equivalent in the London Metropolitan University package which in turn has a "multiple-choice gap fill" exercise that cannot be easily reproduced in *Hot Potatoes*. There was a valuable lesson learnt from the attempt to force incompatible exercise types into unsuitable software. Data storage is a useful and necessary device, but it is secondary to the main functionality of the application. Any attempt to restrict potential functionality because it cannot be shoehorned into a standard storage format is to misunderstand the role of standardisation.

https://doi.org/10.1017/S0958344004000412 Published online by Cambridge University Press

Import and export to other, less similar authoring software that uses XML is, in theory, just as easy, it only being necessary to determine the conventions used by that package to store data and programming the Java parser to rewrite the elements accordingly. Some simple experiments with WebCT have shown this to be true.

This opens up the possibility of transferring exercise material between software packages and vastly increasing data reusability. Once data is stored in XML in any format, it can easily be converted to another format, provided the actual data itself is compatible. This opens up the welcome possibility of automatic upgrade or system transfer. The London Metropolitan University CALL development team had produced a very large amount of interactive exercise material for French, Spanish, German and Arabic. Using the previous storage mechanism described above, each upgrade required an immense amount of manual work as every exercise had to be opened, the exercise type identified and saved again with the instruction typed or pasted in. As part of this process, numerous files and directories had to be deleted, while ensuring that the sound files for the audio enhanced exercises were retained. This laborious procedure is being repeated now for the last time as, when the next upgrade comes, it will be possible to write a routine that will read each XML file and complete the course of action just described automatically, thereby vastly simplifying the maintenance programme.

6 Standards

The work described leads logically to the question of standards. If there was one standard XML format for storing exercises, there would be no need to write file conversion routines, no matter how easy that is. There is indeed a standard, the IMS Question & Test Interoperability Specification (http://www.imsproject.org/), normally referred to as QTI. This is the only serious contender to become an international standard, the problem is that it is extremely complicated and it is not intuitively easy to read the elements directly. This is not an insurmountable problem, although one wonders why counter-intuitive jargon-ridden elements have been chosen when plain ordinary words could have been used, for instance the nature of the data that the element `<render_fib>` describes is not immediately obvious, indeed the author is still unable to determine what it stands for despite reading the documentation. In all matters of computer programming, the use of intuitive, self evident descriptors aids clarity and makes reusability and maintenance much easier.

Given the ease with which data can be converted from one XML specification to another, it is surely better to produce a data storage format that suits the needs of the program concerned and then see if it is possible to convert to the QTI standard. While sensible standards are essential, there is always the danger that an over prescriptive standard is likely to restrict innovation.

> "There is a clear risk that the needs of corporate training departments, defence institutions, and the commercial providers of e-learning tools (all of whom are wealthy in comparison with the state education sector) will tend to dominate the design and implementation of the specifications, and therefore ultimately the nature of the materials created within them." (Arneil & Holmes 2003)

https://doi.org/10.1017/S0958344004000412 Published online by Cambridge University Press

Given the ease with which conversion can be done between data storage formats using XML, the XML specification itself is probably a sufficient level of standardisation for the purpose of converting language learning exercises between different technology and software development approaches. There may be wider issues that require the development of an overarching standard and this may be a greater imperative than the simple exchange of data discussed here. Nevertheless, for the purposes of transferring material from one software package to another, QTI does not yet seem to have proved itself indispensable or, indeed, even very useful. Should such an imperative appear in time, it will be relatively easy to convert the material stored in the XML format proposed above to the QTI or any other standard that appears. It must not be forgotten that the QTI standard is far from universally accepted and is not yet complete and stable. There is clearly a need for wider consultation and considerable research before any such standard can be universally accepted in the way that XML has become.

7 Resources bank

The real significance of this for language teachers is the possibilities it opens up for collaboration and pooling of resources. The ability to migrate material from one software package to another extends data life enormously and frees the practitioner to choose a different approach without extensive rewriting.

Further, to return to the problem cited at the start of this article, that of producing enough of a critical mass to make a serious impact on teaching and learning practice, the ability to take exercises written using different authoring packages greatly increases the potential pool of basic material from which a course of study may be drawn. To start this process, the London Metropolitan University authoring package and all the interactive exercises so far produced for French, Spanish, German and Arabic are being made freely available for download from the web (http://languages.londonmet.ac.uk/CALL/). Teachers may of course decide to use the material as it is, following the same grammar progression as it is currently organised. The intention is, however, that the ability to open an exercise and save it elsewhere will allow teachers to rearrange the exercises to fit in with the progression that they are currently using and to add more where they perceive a gap. This could prove particularly useful if the course follows a book and the teacher wishes to supply some supplementary exercises. It is to be hoped that any additional exercises produced by anyone who uses the software would be added to the common resource bank. Users are not restricted to the London Metropolitan University authoring package. If they were to wish to use *Hot Potatoes* instead they can use the authoring package to export to it. Equally work done in *Hot Potatoes* can be just as easily imported

We have already had some experience of the process of such reorganisation. London Metropolitan University was created by the merger between London Guildhall University and the University of North London. Both universities taught French and Spanish. The material produced by London Guildhall University had to be reorganised to suit the progression previously used at UNL. Without the changes outlined above, this would have been a daunting task, whereas, in the event, the first semester French for Beginners was rearranged in two days.

https://doi.org/10.1017/S0958344004000412 Published online by Cambridge University Press

8 Conclusion

The use of XML has led to significant advances in the fields of usability, maintainability, reusability and portability.

By making the reopening, amending and resaving so much easier, the teacher can edit an on-line course with ease, shaping it to changing educational requirements. By the same token, data maintenance is made easier and therefore more likely to be carried out. Previously, if a mistake was discovered, it often went uncorrected because of the nature of the repair process, but there is no longer any excuse for not correcting errors. As for the software maintenance process, the fact that all the necessary data for an exercise, its type, language, instructions and any supplementary details such as illustrations or grammar feedback are all contained in one file makes the process of upgrade potentially automatic. Data reuse is facilitated and there is already portability between two packages with the potentiality to convert any exercise to any other application that uses XML as a storage medium and which supports that exercise type.

Most importantly for teachers and students, however, is the fact that this technology greatly facilitates the sharing of teaching materials and gives the possibility of extending CALL usage dramatically.

References

Armitage, N. and Bowerman, C. (2002) Knowledge pooling in CALL: Programming an Online Language Learning System for Reusability, Maintainability and Extensibility. *Computer Assisted Language Learning* **15**(1): 27–53.

Arneil, S. and Holmes, M. (2003) Servers, clients, testing and teaching. In: U. Felix (ed.), *Language Learning Online*. Lisse, Netherlands: Swets & Zeitlinger.

Cushion, S. and Hémard, D. (2003) Designing a CALL package for Arabic while learning the language *ab initio*. *Computer Assisted Language Learning* **16**(2–3): 259–266.

Hémard, D. and Cushion, S. (2001) Evaluation of a Web-based Language Learning Environment: the importance of a user-centred design approach for CALL. *ReCALL* **13**(1): 129–142.

Hémard, D. and Cushion, S. (2002) Sound Authoring on the Web: meeting the users' needs. *Computer Assisted Language Learning* **15**(3): 281–295.

Hémard, D. and Cushion, S. (2003) Design and Evaluation of an Online Test: assessment conceived as a complementary CALL tool. *Computer Assisted Language Learning* **16**(2–3): 119–139.

Hémard, D. (2003) Language Learning Online: designing towards user acceptability. In: Felix, U. (ed.), *Language Learning Online*. Lisse, Netherlands: Swets & Zeitlinger.

Lewis, J. and Chase, J. (2003) *Java Software Structures*, Boston: Addison-Wesley.

Nicholson, J. and Mulhern, G. (2001) Integration of ICT into mathematics teaching, *Mathematics Today* **37**: 148–152.

Whitehead, P., Friedman-Hill, E. and Veer, E. V. (2002) *Java and XML*, Toronto: Visual.

Websites

http://languages.londonmet.ac.uk/CALL/
http://web.uvic.ca/hrd/halfbaked/
http://www.lang.ltsn.ac.uk/resources/bankcontents.aspx
http://www.w3.org/XML/
http://www.imsproject.org/

https://doi.org/10.1017/S0958344004000412 Published online by Cambridge University Press