CHAPTER 18 COMPUTER-BASED TESTS

INTRODUCTION

In PISA 2015, for the first time, the primary mode of assessment of student skills was computer based. While paper and pencil was an option, the majority of countries chose to implement the entire survey with the computer. All domains were delivered via the computer, including the optional domain of Financial Literacy. For the countries utilizing computer based assessment, there was a total of 90 language versions.

This chapter focuses on the functionality and technical implementation of the computer based assessments. It also discusses the PISA Student Delivery System (SDS), which integrated the assessments with the computer based questionnaires for delivery of the PISA survey in schools. Finally, we conclude with a discussion of the Open-Ended Coding System (OECS), used for coding of open responses.

ITEM RENDERING

The items for PISA 2015 were implemented using web based technologies: HTML, CSS and JavaScript. Modern web browsers (we used Firefox v22) provide a rich set of functionalities for attractive presentations and engaging interactivity. At the beginning of the development work, an overall user interface was designed, with common elements such as navigation, help and progress indicators. The items were built in such a way that these common elements were shared, so that the same version was used in all items in each language version.

PISA items are grouped into units consisting ofa set of items with a common stimulus. Each unit was constructed independently, with the questions and stimulus components developed first in English, then translated into French to create the two source language versions. Each unit could be viewed on its own, or grouped into test forms for delivery to students as part of the assessments.

In some cases, such as Collaborative Problem Solving (CPS) and the interactive Scientific Literacy units, common functionalities were split out into shared programming libraries that could be reused in multiple units. For example, in the CPS units, the interactive chat functionality was built as a shared library. For each unit, an XML representation of the chat structure was read in. The library then displayed the chat entries to the student, presented response options, and managed the interactive displays that were unique to the units (shown on the right side of the screen). The library also managed the recording of data and scoring of the student's performance based on unit specific criteria.

As well as the visual aspects of the PISA items, the automated coding of student responses was also implemented using JavaScript. Shared libraries were created to implement this coding in a common way. The libraries targeted the various response modes used within PISA:

- Form: for all responses using common web form elements such as radio buttons, checkboxes, dropdown menus and textboxes.
- Drag and Drop: for items using drag and drop as the response mode
- Selection: for items where the response is given by clicking on an object or region of the screen. This can be, for instance, clicking on part of an image, a cell in a table or a segment of text.
- Ad hoc: A general catch all that uses custom JavaScript code to implement the coding. This was used for unique situations, such as coding of the CPS and interactive Scientific Literacy items.

In all cases except the ad hoc coding, the coding for a specific item was specified using rules composed of conditional expressions and Boolean operators. Each library implemented appropriate conditional expressions (e.g., a CONTAINS operator in the Drag and Drop library to test if a drop target held a particular drag element).

TRANSLATION AND ONLINE ITEM REVIEW

Given the need to support up to 90 versions of each unit, one for each national language, automated support for integration of national translations and adaptations was critical. Supporting this process started when the units were initially developed. The HTML files that implement the display of the unit contain only HTML markup and the text to be shown on the screen. Layout and formatting specifications are stored separately in CSS stylesheets. The text of the units is then extracted from these HTML files and saved to XLIFF (http://docs.oasis-open.org/xliff/v1.2/os/xliff-core.html) files. XLIFF is a standard file format used for computer supported translation. Once a translation is completed, the XLIFF file is injected into the original source version of the item, resulting in HTML files with the translated text of the unit.

Experience has shown that the quality of a translation is improved if the translators can view their translation in the context of the items they are translating. In an ideal world, translators would work in a completely WYSIWYG (what you see is what you get) mode, so that they are translating directly in the items. This is not technically feasible, and also may tempt translators to focus more on the visual aspects of the items, which are tightly controlled for comparability, rather than the translated text. A good compromise is to provide an easy to use preview capability, giving users a fast way to view their translations as functioning items. For PISA 2015, this capability was provided through an online portal. Users were able to upload an XLIFF file, either partially or completely translated, and in a matter of seconds they received a preview of the given unit in exactly the same view as a student would receive. From this preview, they could interact with the units in the same way as a student responding to the unit, an important factor for the complex units of Collaborative Problem Solving and Interactive Science. This preview also allowed countries to test and identify potential problems with their translated units before receiving the final software to be used in schools. Problems were fixed as early in the schedule as possible.

SCHOOL COMPUTER REQUIREMENTS

The goal for PISA 2015 was to use the computers available in the sampled schools with no modifications. The PISA system supported Windows based computers in the Field Trial and both Windows and Macintosh computers for the Main Study. The following minimum technical requirements were established for both Field Trial and Main Study.

- CPU Speed: At least 1500 MHz
- Operating Systems: Windows XP or later, MacOS X 10.7 or later
- Installed Memory: 512MB for Windows XP, 1024MB for newer Windows versions, 2048MB for Macintosh
- Free Memory: 384MB for Windows XP, 717MB for newer Windows versions and Macintosh
- Screen Resolution: 1024 x 768 pixels or greater

These were the minimum requirements. Computers with higher capabilities would obviously perform better (e.g., respond faster) when delivering the survey, but these were the minimum settings that would provide adequate performance.

SYSTEM DIAGNOSTIC

In order to verify that the available school computers met these minimum requirements, a System Diagnostics application was provided to countries. This application simulated the test delivery system, but rather than showing the PISA tests or questionnaires, it ran a program to check the computer's hardware and software setup and report this back to the user, typically the test administrator or technical support staff in the school. The System Diagnostics was provided to countries approximately six months prior to the start of the Field Trial and Main Study. This allowed countries to provide the software to sampled schools to determine if their computers could run the PISA Survey. Additionally, it was recommended that test administrators run the System Diagnostics on the day of the test.

For cases where schools did not have adequate quality or quantity of computers, test administrators brought laptops into the school to augment the available infrastructure. In a few cases, countries chose to administer the PISA tests in all sampled schools on laptops brought into the schools. This avoided "surprises" on the day of the test, where computers were not available or not functioning properly.

TEST DELIVERY SYSTEM

The PISA 2015 test delivery system, called the Student Delivery System or SDS, integrated the PISA CBA tests and questionnaires for a country, along with of a number of components packaged together to run as a standalone application on a USB drive. The SDS did not require network connectivity or external resources to operate. All software and data was on the USB drive, and results were saved back to the USB drive. The SDS could also be run from the computer's hard drive if desired. The components which made up the SDS included:

- Apache web server (https://httpd.apache.org/)
- MySQL database engine (https://www.mysql.com/)
- PHP interpreter and libraries (http://php.net/)
- Firefox Portable web browser (http://portableapps.com/apps/internet/firefox_portable)

Together with these open source applications, the actual test and questionnaire content were included. To display this content to the students and collect the results, the PISA Test Delivery System was implemented. Using portions of the open source TAO test delivery system (http://www.taotesting.com/) as a basis, the system was custom built for the needs of PISA. This included implementation of the test flow, which assigns the designated test form and questionnaires to a student, then sequences through the test units and questionnaires in the appropriate order. It also includes the functionality for collecting the survey results and exporting them when the tests are completed. The PISA Test Delivery System was built primarily using PHP, with JavaScript used for interactive displays and communicating between the web browser and web server.[1]

[1] The software that implemented the PISA tests in 2015 will be released as open source. Details on availability are not finalized at the time of writing.

The system was launched by running a custom executable program (PISAMenu) written for controlling the PISA tests. Separate programs were written for Windows and Macintosh. From this program, a test administrator could launch the PISA tests, launch the System Diagnostics, or manage exported data files. These exported files are described below. Launching either the PISA tests or System Diagnostics would start the web and database servers, then launch the Firefox web browser to begin the process. When the PISAMenu program was shut down, the various components of the SDS were also terminated.

The Firefox browser used for the PISA tests was configured to run in "kiosk mode", so that it filled the full screen of the computer, making it difficult for users to access external applications when running the PISA test mode. A keyboard filter was also installed so that students could not easily leave or terminate the browser window, e.g., by pressing Alt-Tab, and switch to another program during the test. The keyboard filter did not completely block such attempts, though. It was not possible to block the Ctrl-Alt-Delete sequence under Windows, as this required installation of a custom software driver at the system level. Our goal was not to install any software on the school computers, so this driver was not used. It was expected that the Test Administrator would monitor the students during the test and watch for cases of students trying to break out of the system.

The first screen a student would see after the test was started was the option to select one of three sessions: PISA Tests, PISA Questionnaires and Financial Matters. The latter was for the Financial Literacy tests, and was only shown in countries participating in this international option. After selecting the appropriate session (which usually was done by the test administrator before the students arrived), the student was prompted for a login ID and password. The login ID was the 15 digit student ID assigned by KeyQuest as part of the sampling process. The password was also assigned by KeyQuest and was an 11 digit number. The first five digits comprised a checksum of the student ID, guarding against input errors. The next three digits encoded the test form which should be used for the student. The last three digits were a checksum of the three digit test form number.

After logging in, the student could optionally be shown a screen asking to select a language for this session. While the SDS was built with all the national languages available for a given country, it could be configured to support only one language. This was the recommended method of operation, where the test administrator chose the language configuration when starting the SDS, based on the school where the testing occurred. However, in some countries, it was necessary to allow the students to choose the language of assessment. The typical reason for allowing student choice for the language was for countries and schools with mixed language environments. In these cases, students decided at the time they started the survey session which language to use. The test administrator would guide students through the login and language selection process.

An important facet of the USB setup was protecting the test content on the USB drives. The PISA tests contain secure test materials, and people who obtain a USB drive should not have access to the test items except during the administration of the survey. To accomplish this, the files for rendering all test materials were stored in the MySQL database on each USB drive. The files were stored in an encrypted format, and access to these was controlled via the web server. When a testing session was first started, the PISAMenu program would prompt for the password used to encrypt the files. Each country was assigned a unique password. This password was validated against known encrypted content in the database and then saved for the duration of the testing session. When a request was made to the web

server for some part of the test content (e.g., one of the web pages or graphic images), the web server retrieved the content from the database and decrypted it on the fly.

One advantage of the SDS architecture used in 2015 was that it could be run without administrator rights to the local computer. This was a big improvement over past PISA cycles, reducing greatly the amount of technical support needed within the schools.

DATA CAPTURE AND SCORING STUDENT RESPONSES

Results from the PISA tests and questionnaires were stored on the USB drives. Data was saved as the students answered each question, then exported at key intervals during the survey. At the end of a session, the results from that session were exported in a single password protected ZIP file. For the PISA tests in Session 1 and 3 (the standard PISA domains and the optional Financial Literacy domain, respectively), the ZIP files contained XML formatted data including logs of the students' actions going through the tests and files with the "variables" exported from the test. The following set of variables were exported for each item in the tests:

- Response: A string representing the raw student response.
- Scored Response: The code assigned to the response when the item was coded automatically.
- Number of Actions: The number of actions taken by the student during the course of interacting with the item. Actions counted were clicks, double-clicks, key presses and drag/drop events.
- Total Time: The total time spent on the item by the student.
- Time to First Action: The time between the first showing of the item and the first action recorded by the system for the item.

Besides these five standard variables, some more complex items, such as the Science simulations and Collaborative Problem Solving, had custom variables that were of interest to the test development and psychometric teams. For instance, for the Science simulations, the system exported counts of the number of experiments performed and the final set of results from each of these experiments.

An important task in PISA is coding of student responses. For computer delivered tests, many of the item responses can be coded automatically. In PISA 2015, this included multiple choice items, drag and drop items, numeric response items, and complex responses to Science simulations. Additionally, in Collaborative Problem Solving, all coding was done automatically, based on the path taken by the student in the chat, as well as other inputs depending on the scenario.

For standard response modes, such as multiple choice or numeric entry, automated coding was done using a rule based system. The correct answer (or partially correct answers in the case of partial credit items) were defined based on Boolean rules defined in a custom syntax. Simple conditionals were possible, e.g., to support different combinations of checkboxes in a multiple selection item where two out of three correct options should be selected. For numeric response items, the rules could check for string matches, which required an exact match against a known correct answer, or numeric matches, which used numeric equivalence to check an answer. For numeric equivalence, for instance, 34.0 would match 34, but they would not match when using string matching.

A challenging part of evaluating numeric responses in an international context like PISA is how to parse the string of characters typed by the student and interpret it as a number. There are differences in decimal and thousands separators that must be taken into account, based on conventions used within

countries and local usage. Use of these separators is not always consistent within a country, especially with increased migration and the pervasiveness of the Internet. For PISA 2015, the coding rules tried multiple interpretations of the student response to see if one of them could be coded as correct. The numbers were parsed in different ways, changing the decimal and thousands separators, testing each option to see if a correct response could be granted full or partial credit. Only if no alternate interpretation of the response resulted in a correct answer would the answer be coded as incorrect.

OPEN ENDED CODING SYSTEM

While automatically coded items formed a significant portion of the tests for PISA 2015, approximately 30% of the items resulted in an open ended response that needed to be coded by a human scorer or coder. On paper, this would be done directly in the test booklets. On the computer, a procedure was necessary to extract the responses provided by the students and present them to human coders. It was important to present these responses in a way that that reflected the students' intent. This task is complicated by the fact that these responses could be more than just text. For some items, a student would select an option from a multiple choice part, then type in an explanation for why they chose that option. Additionally, in Mathematics, students could use an equation editor to insert complex mathematics notation into their response.

For PISA 2015, the coding of these responses was done using the Open Ended Coding System (OECS). The OECS took the open ended response data generated from the SDS, and, following a coding design specified by the psychometricians, assigned these responses to coders. Each coder received responses organized by item, so that coders focused on one item at a time. These responses were formatted and saved in PDF files. The PDFs used PDF Form technology (https://en.wikipedia.org/wiki/Portable_Document_Format#AcroForms), allowing a coder to record his/her judgement of the student performance. The coded PDF files were then imported back into the OECS and saved in a database. From there, reports were generated with statistics evaluating the reliability of the coding, as well as the completion status for each coder and item. Finally, the coded results were exported and then imported into the Data Management Expert to be integrated with the other PISA data.

The use of PDF files for presentation of response data and collection of the resulting codes had advantages and disadvantages. One advantage was that coders were able to work offline, without need for an Internet connection. But disadvantages included slow PDF generation times when creating the files for the coders, and challenges with managing a large number of PDF files. For some coding designs, there could be hundreds of PDF files for each domain processed during a coding session. Proper managing and accounting of the many files presented a challenge. The OECS organized the files such that a folder of files could be copied for each coder, but coders' computer skills varied and sometimes coded files could be overwritten with files for other items. In the future, if PDF files continue to be used, a user friendly application could be built to help with the management of the files. Alternatively, an online, web-based delivery of the student responses could be developed, obviating the need for files.