Chapter 5
Platforms for Delivery of Collaborative Tasks

Nafisa Awwal, Patrick Griffin, and Sam Scalise

Abstract Prior to the inception of the ATC21S^{TM} project, no single platform existed for the implementation of human-to-human internet interactive tasks. (The acronym ATC21S^{TM} has been globally trademarked. For purposes of simplicity the acronym is presented throughout the chapter as ATC21S.) Given the global goals of the project, the assessment tasks and the professional development materials for administration and teaching interventions were constructed on several platforms. The characteristics and capacities, and the advantages and disadvantages, of these platforms are described in this chapter. The mathematics-based collaborative tasks were developed on a Hong Kong-based platform and migrated to the online testing framework of the Assessment Research Centre (ARCOTS) at the University of Melbourne. The reasoning-based tasks were developed directly on this platform. Another platform (FADS) was used for the development of the Learning in Digital Networks or ICT (LDN-ICT) tasks. An attempt was made to modify a serious game product but the platform requirements of the developer (Pixelearning) led to this attempt being abandoned. Issues such as automating assignment of login codes, report generation, and real-time synchrony are discussed in detail in this chapter. The chapter highlights factors that discriminate between the maintenance and management of a local system of assessment and teaching materials and the issues that need to be considered when maintaining and managing such a system on a global scale.

Overview and Design Rationale

The advancement of technology, the growing use of the Internet and the accessibility of the World Wide Web have had a major impact on education. The use of computers and other electronic devices now plays an important role in designing, delivering and

N. Awwal (✉) • P. Griffin
Assessment Research Centre, Melbourne Graduate School of Education,
University of Melbourne, Parkville, VIC, Australia
e-mail: n.awwal@unimelb.edu.au

S. Scalise
University of California, Berkeley, CA, USA

P. Griffin, E. Care (eds.), *Assessment and Teaching of 21st Century Skills*, Educational Assessment in an Information Age,
DOI 10.1007/978-94-017-9395-7_5

authoring educational assessments. The uses of web-based and closed computer-based platforms for delivering assessments has become common practice. Many of these systems are still limited in supporting delivery of traditional assessment methods such as multiple-choice questions, ‘fill in the blank’, and so on. The advantages of using real-time delivery media include remote accessibility (web-based), automated scoring, instant feedback, convenient data storage and so forth. Although these systems have accelerated the process of marking and analysing students’ responses, they have not yet provided any psychometric advantages over traditional paper-and-pencil-based assessments because they have not been used to capture data that link assessment to the students’ cognitive processes or decision making behaviour.

Although still in their relative infancy, however, computer-based assessments can be used to capture information-rich student performances. Unlike traditional test items, computer-based assessments can provide the means for capturing, recording, storing, processing and scoring data that reveal the processes by which students reach their answers. The process data derived through such assessments are considered richer than traditional data as they can describe the type, order and quantity of interactions with a task (Bennett et al. 2003; Greiff et al. 2012). This form of data is essentially collected through the capture of discrete mouse and keyboard events in a process discussed in more detail by Adams et al. (2015; Chap. 6). The technological requirements of such assessments are those for the capture, storage and processing of time-stamped click events. Various forms of client and server configurations can be used for this purpose. The key technologies used to develop the collaborative problem solving tasks in ATC21S are described in the following section. The technologies for the delivery of LDN-ICT tasks are described in the subsequent section.

Delivery of CPS Tasks

The current suite of collaborative problem solving (CPS) tasks for ATC21S is based on an established set of computer-based problem solving assessment tasks developed by Zoanetti (2010) for single-user administration. Their conversion into tasks for two users with differentiated screen views and real-time collaboration tools facilitated the testing of their capacity to elicit collaborative problem solving skills (refer to Care et al. 2015; Chap. 4 for more details).

The CPS assessment tasks developed were delivered through the online assessment system ARCOTS (Assessment Research Centre Online Testing System) of the Assessment Research Centre, University of Melbourne. The authentication process for students controlled access to the tasks. Since the delivery was online, students were able to access the tasks at their convenience via any client browser with a relatively up-to-date Flash Player plug-in.

The graphical components of the reasoning-based CPS tasks were designed using Adobe CS5 and programmed with ActionScript 3. The use of Flash limited the range of devices, such as tablets with iOS, that could be used for the assessment tasks. SmartFoxServer 2X[1] was chosen as the socket server technology enabling clients to share communication and object-manipulation data. SmartFoxServer 2X was chosen over other alternatives because of its availability in a free community version designed to support developers. Thus, all packages for libraries were incorporated into the development environment for use throughout the development of these tasks. To support the design of some of the animations in the tasks, motion tweening utilities were also used. In accordance with the requirements of the ATC21S project, the tasks were initially delivered in four different languages: English, Spanish, Finnish, and Dutch; the XML Strings Panel provided in the Flash IDE was used to embed the appropriate language, as determined by the client operating system, in each task. The tasks automatically detect the language settings of the user's web browser and IP location, and if the language is supported by the system, then all of the assessment tasks are shown in that respective language; else everything defaults to a common language which had been set to as 'English'.

The server components included the LAMP stack (Linux, Apache HTTP Server, MySQL and Perl/PHP/Python) as part of the multi-user architecture, implemented on a remotely hosted server. SmartFoxServer 2X was mounted to provide an open connection between the clients via the server. Use of such a socket application became essential because although single-user architecture might be able to support turn-based games, it would constrain design of tasks which rely on real-time (or minimal lag) updates for both users following activity by one or the other. A number of custom Java packages were developed and incorporated into the platform to supply some of the assessment task logic and to handle the flow of both-way data communication between clients and the MySQL database. The database – designed as a relational structure – and the application packages were configured to support the various target languages and output the resultant task view at the client's end.

Each of the curriculum-based CPS task was implemented as an Adobe SWF Flash 10 object embedded in a customised PHP page and delivered from the same Linux based web hosting server. The swf in the client's browsers was connected to a Flash Media Server for communications and synchronisation of shared objects between the collaborating users. The web server also provided an AMF gateway (which is a PHP based, open source package for handling server-side calls) to allow clients to log into collaborative session events. The task elements written in ActionScript 3 require remote shared objects on Flash Media Interactive Server[2] (FMIS). FMIS is necessary to support the communications and synchronisation of shared objects between the collaborating users and is considered a standard choice for streaming media and shared object synchronisation for swf-based clients. The language detection in this case relies on the language information sent by the browser to the server in the request header.

[1] SmartFoxServer is a middleware for creating multi-user applications, communities and games with Adobe Flash/Flex/Air, HTML5, Android, Unity3D, Apple iOS, Java etc. 2X is its community edition.

[2] Flash Media Interactive Server is from the Adobe Media Server family and supports multi-user media and games delivery on various devices with a wide range of deployment options.

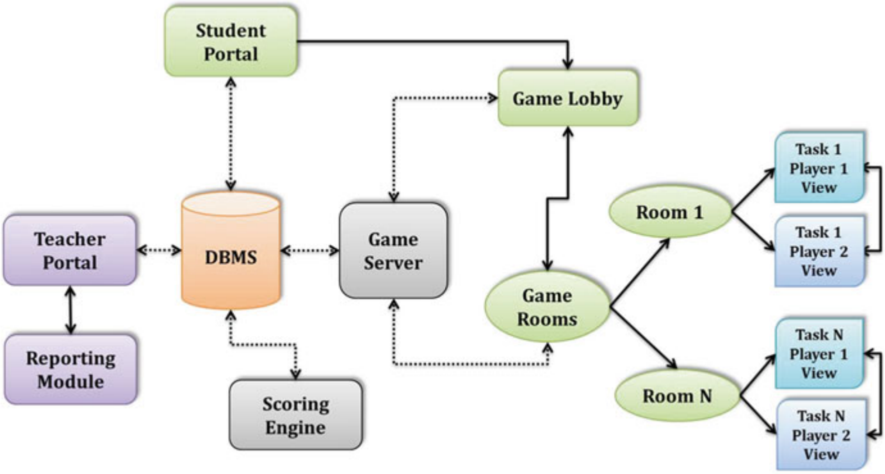

Fig. 5.1 Schematic diagram of the platform

The initial system was eventually expanded to incorporate Flash assessment tasks designed by third parties who used the FMIS as the socket server technology. Hence, an instance of FMIS was also connected on the Linux operating system as a service. The system was built to integrate with the established ARCOTS platform so that the authentication, scoring and reporting modules were common across all assessment environments. While it is not yet IMS QTI[3] standards compliant, and therefore not optimally interoperable, some modularity was built into the platform to help ensure that other forms of assessment tasks could be accommodated. Appropriate fonts for each of the target languages were also embedded in the chat messaging interface.

The assessment tasks were created to be executed in a similar fashion to that of a game within an online multi-user gaming architecture (Fig. 5.1). The collaborative environment followed some basic steps to give users access to the tasks. As mentioned earlier, access to tasks is controlled by an authentication process and limited to participating countries. Hence, students can log into the system using valid credentials (unique student identifier and a shared team code to pair collaborators in a session). On successful login, a student enters a virtual room, referred to as '*Lobby*' in the gaming environment and is presented with a set of CPS tasks for selection. Once students select a task to attempt, they are further provided with options to select a role for different views of the task (unknown to them) as presented for interaction. As soon as the students click on a user icon (avatar) to select their role, a virtual room is created dynamically on the basis of predefined rules (a combination of unique task identifier, shared key and other variations). Upon

[3] The IMS Question and Test Interoperability (QTI) specification defines a standard format for the representation of assessment content and results, supporting the exchange of this material between authoring and delivery systems, repositories and other learning management systems. It allows assessment materials to be authored and delivered on multiple systems interchangeably. It is, therefore, designed to facilitate interoperability between systems.

admission, the student is informed that the room is waiting for their pre-assigned partner to join them. If the student is the second of an assigned pair to click on a user icon, that student will enter the virtual room created by the first student, and the task contents will automatically be presented to both students with differing views and resources. While paired under a session, students can collaborate or operate among themselves and within the problem space. At the completion of a task, students are redirected back to the task-list dashboard or onto a self- and/or peer-assessment.

The whole process steps users through a shared session while collecting their responses for collaborative behaviour. The database logs detail students' responses including actions by students that trigger a change to the task on screen (buttons, text inputs etc.) and also events that don't trigger any change (for example, clicking on buttons which are not enabled). In short, any clicks or activities on screen by students are captured regardless of their effectiveness, because invalid, ineffective and tentative actions may prove to be more informative in later analysis. Classes were defined (PHP and Java were used) to record such responses and to keep track of how users are interacting with the system.

The system is also designed to deal with unexpected interruptions due to internet connection failures or other technical issues: in such events it allows students to continue access from their previous responses. During a collaborative session, if one student of the pair encounters a technical problem and closes their window (or hits refresh/back button on their browser), the system will allow both users to re-enter the task at the initial state of the last common page they shared. On return to the same task, the first returning user sees the role selection page with their previous role already selected and both users are automatically returned to the page on which they were last collaborating. If, at any point, one partner's session stops, the other partner should not be able to proceed to a new page: this can happen only when both partners confirm the next page request within a task. In addition, the other partner is provided with a system notification of partner loss. If a partner closes or refreshes their browser window, the system will detect this activity and – as long as the network connection remains active – inform the other partner of it. The tasks being collaborative, no user is allowed to move forward without corresponding partner.

Given the intended proliferation of tasks in the ATC21S project, programming re-usable client-side and server-side classes and/or packages with ActionScript 3 and Java became an important objective because it allows for efficient up-scaling. These classes managed the majority of processes common across the set of tasks available, including student login verification, real-time chat messages, data storage to the database, real-time sharing of information about task objects between students, and other aspects of game logic.

Delivery of LDN-ICT Tasks

A set of LDN-ICT tasks – different in theoretical context to the CPS tasks – was both developed and delivered through the Formative Assessment Delivery System (FADS) of the Bear Center, University of California, Berkeley. Like the CPS tasks, these were delivered online, but through FADS, using appropriate authentication protocol for participating countries in the ATC21S project. FADS had adhered to a model-based assessment system with reasonable Application Program Interfaces (APIs) and has some compliance with SCORM[4] and IMS QTI. SCORM is a collection of standards and specifications for web-based e-learning. It defines communications between client-side content and a host system called the run-time environment, which is commonly supported by a learning management system. SCORM also defines how content may be packaged into a transferable ZIP file called a "Package Interchange Format".

The LDN-ICT tasks were designed and developed with Adobe Flash in ActionScript 3. All of these Flash tasks created within FADS communicate with an intermediate Flash object, "integrator", which forwards the required requests to the backend database architecture. The integrator is defined by a flash parameter with the name "integratorUrl". The value contains the URL to the swf file that implements the integrator (e.g. http://berkeley.edu/somedirectory/integrator.swf).

All the LDN-ICT tasks were deployed as Adobe Flash swf object and contained a configuration XML that is used for rendering and scoring purposes. This XML needs to be stored by the backend and is delivered to the Flash task through the integrator. In the case of the FADS integrator, this communication is encrypted to avoid disclosure of correct and incorrect answers, but this is not an absolute requirement.

The set of LDN-ICT tasks was delivered only in English and Spanish, with slightly different content design for the respective countries using those languages. The language of the content was not reliant on browser language detection but merely on the task selection that is embedded with users' preferred language. The chat messaging interface was absent from these form of assessment tasks but FADS has used other forms of standard web applications, such as Google Docs, Webspiration Classroom, Kodu GameLab and other similar programs, that facilitate communication and sharing among collaborating groups. In addition, the tasks necessitate the use of external sites and are embedded with required resources. Unlike the CPS tasks, these assessments were designed to be played among two to four collaborating partners.

Most of these tasks are designed to be revisited by the student so that the student can review and alter their previous responses. When each task loads, it assembles an XML request to retrieve any prior responses and uses the integrator's defined function (e.g. getMyResponses) to retrieve past responses and submit the request to the hosting environment. When the responses are received, the integrator issues an event such as 'responsesReceived', to which the task responds by calling the integrator's function (e.g. getMyPreviousResponsesXML), which delivers the response XML. Some of these tasks are designed to incorporate a student's responses to other tasks. In those cases, the task calls another integrator function (e.g. getPreviousResponse) to retrieve relevant responses for the student. This function also handles unexpected interruptions to collaborative sessions caused by internet connection failures or other technical issues.

[4] Sharable Content Object Reference Model (SCORM) is a specification of the Advanced Distributed Learning (ADL) Initiative, which comes out of the Office of the United States Secretary of Defense.

A number of these LDN-ICT tasks rely on student-specific information stored in the hosting environment, such as the student's age, team assignment, or login credentials, to alter the content that is displayed and the availability of access to external websites. Tasks assemble a request for such information as a piece of XML and use the integrator's function (e.g. getDemographics) to retrieve the student's profile and relevant demographic information and submit that request to the hosting environment. When the information is received, the integrator issues an event like 'demographicsReceived', to which the task responds by calling the integrator's function, such as getMyDemographicXML, which delivers a similar XML fragment containing the requested information.

To gain access to the LDN-ICT tasks in a collaborative environment, the assessment system follows a similar multi-user architecture to that of the CPS tasks. Again, as in the case of the CPS tasks, access is restricted through authentication protocol. Collaborating students within the same team are presented with parallel views and resources in their respective browsers, but are allowed to progress into the task space at their own pace, without restrictions on collaboration with prospective partners. Students are allowed to complete the task in their own time and are not required to wait for collaborating partners. Students are made aware of their partners only through the use of the collaborative spaces they share. The tasks were designed to capture student responses or actions, both shared and unshared, within the task environment and external resources. The responses and/or actions mainly consist of various forms of user inputs (textual, graphical, multimedia, etc.) and retrieval of information through the range of resources provided both internally and externally. Any such activity within the task space is thus captured and fed into the backend of the FADS database in the appropriate format as defined by the content delivery mechanism. Again, due to the use of Flash, the range of useable devices was limited (e.g. PCs but not Apple iPads), and the reliance of these tasks on external resources and applications imposed further restrictions on the availability of those resources.

Lessons for an Integrated Collaborative Assessment Platform

This section will focus on some of the issues identified with the earlier assessment platforms and provide some suggestions for design choices on technologies as guidance for future implementation. As discussed in earlier sections, multiple platforms co-existed for delivering the range of assessments of the ATC21ST project. Having multiple platforms with differing technological requirements made it difficult to deliver examples of the ATC21S assessments (both CPS and LDN-ICT tasks) to a wide range of students. It was realised that existing know-how needs to be integrated with emerging technologies to introduce and consolidate multiple portals, automated scoring algorithm of student responses and the feedback mechanism to teachers; many of these were missed in the earlier version of the delivery platforms. Key considerations for such an integrated system should include access for large numbers of simultaneous users, the dispersed locations of collaborating users and devices in use, the multi-lingual capability content management system and results of user attempts provide feedback in real-time. The aim of such a feedback mechanism, similar to the one developed as part of the Assessment and Learning Partnerships project (Griffin 2000), is to allow teachers to monitor student progress over time and link that progress to successful teaching strategies.

Cloud based technologies for assessments are relatively new, but other e-business' (e.g. EBay, Amazon etc.) have commercialised their development efforts into services. This can be used as an alternative medium for interactive task delivery to recuperate some of the issues that earlier Flash tasks or technology may have imposed. Open standards, specifically the W3C standards, in such technologies can be adhered easily, making it a cost effective solution for such deployment. In addition, such technologies offer a variety of patterns and practices that can allow the design of any system to ensure it can be scaled to support a large number of users. Such technology can generate 10–40 times more transaction data compared to traditional online test items due to the nature of the interaction from these collaborative assessment tasks.

As discussed, a key criterion for this project was to develop interactive assessment tasks that use synchronous communications to provide support for a collaborative process. HTML5 has been gaining momentum over Flash in creating such applications. Unlike Flash, HTML5 is supported by all devices (such as Apple iPads) and is not constrained by licensing limits for the use of additional server based software. It can accommodate synchronous communication among thousands of users and thus allow for scalability on the level required by most projects. This option is likely to make financial support more viable for deployment in schools later.

A system based on these latest technologies will provide a consistent experience for users across all browsers and platforms. A disadvantage of these design choices is that HTML5 standards are not adequately supported by current browser versions that are more than two years old. However, while some schools may still be using older version browsers, the freely accessible latest versions do help to alleviate this problem and it is expected that all schools will upgrade to current HTML5 browsers over the coming years.

Implications and Future

Development of complex interactive assessment tasks poses logistical and pedagogical challenges, not only for the developers but also for potential users of the technology. Schools – the target locations of such assessment use – are usually restricted in their access to state of the art technology, including software, hardware, internet access and bandwidth. This gap between the available technology and student access to it is more profound in rural areas and other locations that are not technology-rich. As the need for teaching 21st century skills is recognised, there will be an increasing demand for such assessments, and schools are likely to become better equipped to meet their students' needs for the future.

To design challenging and rational tasks that requires concrete collaboration while preventing all single-player solutions is not easy. However, as new technologies and standards evolve, it is reasonable to expect that new collaborative tasks that make use of the functions described here will be conceived and created. Future plan need to include development of a range of new collaborative tools that can accommodate more features for collecting salient information on students' collaborative problem solving processes. As new schools and new countries take up the ATC21S system, there may be a need for the integration of more functionality and some scenarios may entail new development. To date, the development has been a learning process for the teams involved, and many ideas and concepts have been continually refined. With research needs and opportunities rapidly evolving, an even greater need to explore and extend the system is yet to unfold.

References

Adams, R., Vista, A., Scoular, C., Awwal, N., Griffin, P., & Care, E. (2015). Automatic coding procedures for collaborative problem solving. In P. Griffin & E. Care (Eds.), *Assessment and teaching of 21st century skills: Methods and approach* (pp. 115–132). Dordrecht: Springer.

Bennett, R. E., Jenkins, F., Persky, H., & Weiss, A. (2003). Assessing complex problem solving performances. *Assessment in Education, 10*(3), 347–359.

Care, E., Griffin, P., Scoular, C., Awwal, N., & Zoanetti, N. (2015). Collaborative problem solving tasks. In P. Griffin & E. Care (Eds.), *Assessment and teaching of 21st century skills: Methods and approach* (pp. 85–104). Dordrecht: Springer.

Greiff, S., Wüstenberg, S., & Funke, J. (2012). Dynamic problem solving: A new assessment perspective. *Applied Psychological Measurement, 36*(3), 189–213.

Griffin, P. (2000). *The profiles in practice: School reporting software*. Portsmouth: Heinemann.

Zoanetti, N. (2010). Interactive computer based assessment tasks: How problem-solving process data can inform instruction. *Australasian Journal of Educational Technology 2010, 26*(5), 585–606.