https://doi.org/10.48009/2_iis_2005_303-309

DYNAMIC ONLINE ASSESSMENT SYSTEM

Reggie Davidrajuh, University of Stavanger, Norway, reggie.davidrajuh@uis.no
Koneswaran Tharmalingam, SARA Systems, Norway, koneth@online.no

ABSTRACT

This paper proposes to focus on the design, implementation and evaluation of a dynamic online assessment (DOA) system. In addition to the facilities offered by the most commonly used static online assessments, DOA is also interactive in real-time. This paper focuses on the technology aspects of design and implementation of a DOA system.

Keywords: Dynamic online assessment, Java, XML, chat server, database server

INTRODUCTION

Online assessment is no longer a new phenomena. Literature review provides many works that explore Web-based test systems for online learning [1, 5, 7, 11, 12]. Generally, online assessments provide the facilities stated below.

For the students:

- Tests can be taken anytime
- Tests can be taken anywhere; Questions can be attempted in a stress-less environment
- Test can be taken using a simple personal computer and the minimal requirement is just a Web browser
- Questions can be viewed with special visual effects such as 3D, and objects in motion can be viewed.

For the faculty:

- Marking the test is done automatically and instantaneously; the faculty is relieved from these time consuming duties,
- Questions can be easily recycled from the question bank (database), easily edited and changed; different versions of the same question can be generated for different students

For the administration:

- The marks are automatically collected, analyzed, and disseminated for purposes like evaluation of teaching and learning

Though online assessment are now being extensively used as a standard examination option in both the online and the traditional classroom education, however, it is important to note that these assessments are *static assessment*. Static assessment means, the exam questions are pre-prepared, fixed, and stored in a database, though questions can be chosen randomly during online assessment. In other words, online assessments are basically automated equivalents of written examination with pre-prepared exam question.

This paper deals with *dynamic online assessment* (DOA) system – the automated equivalent of the spoken examination. While the DOA provides all the advantages that a static online

assessment provide, except that the examination can not be taken anytime, as the examiners or the panel that administer the examination must be online during the examination. The questions are no longer static as they can be changed or tailored by the examiners during the course of the assessment. Figure-1 shows the interactions between the actors involved during the static and dynamic settings.

Advantages of DOA. Contrast to its static version, DOA offers a progressive assessment in which the line of questioning can be varied during the course of the examination to benefit the candidate; live interactions between the examiners and the candidate can be used as a motivational tool to encourage the candidate to reveal his full potentials. Most importantly, DOA allows the candidate to initiate and take some control of the interaction, which is completely missing in the static online assessments.

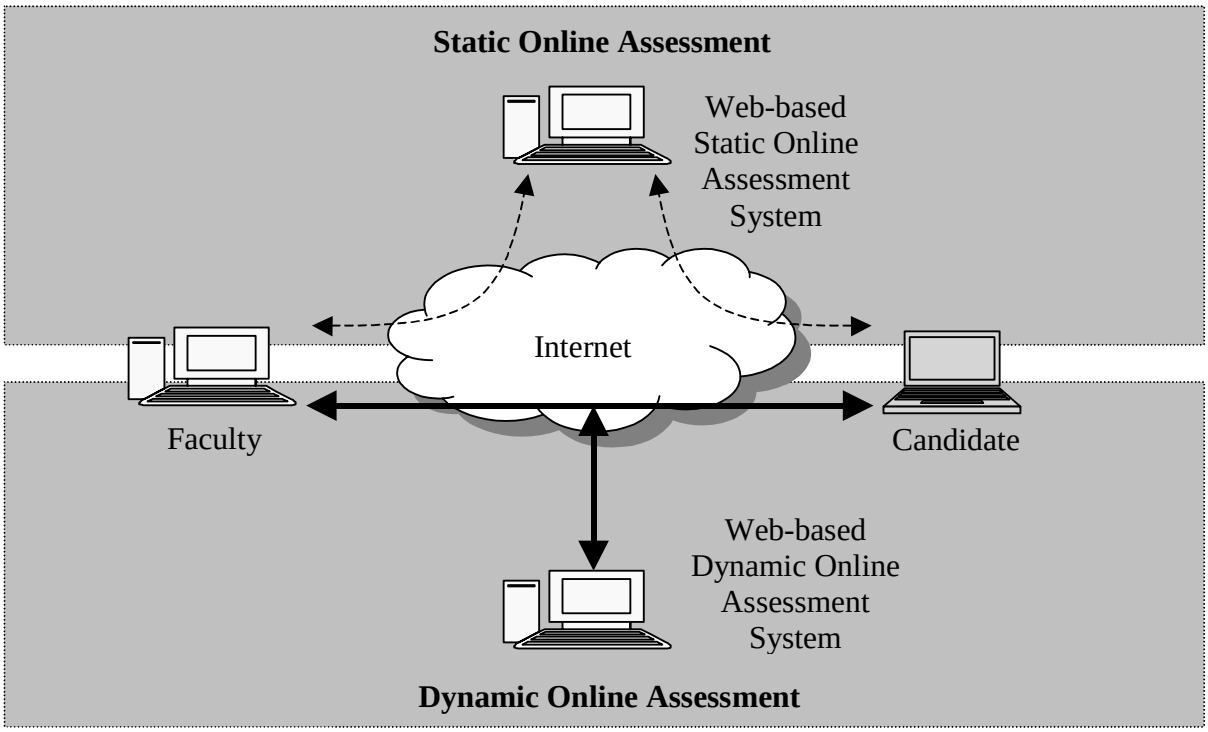

Figure-1: Static versus dynamic online assessment systems

The aim and scope of this paper. The aim of this paper is to show that with a methodical approach of utilizing enabling technologies like Java and XML, design and implementation of a DOA system can be easily built. Thus scope of this paper is limited to technological aspects. The pedagogical, psychological, and social factors in dynamic versus static online assessment are out-of-scope of this paper; interested readers are referred to [8].

Organization of this paper. Section 2 presents the system analysis and design of a DOA system. Section 3 presents implementation details of a testing prototype.

DEVELOPING DYNAMIC ONLINE ASSESSMENT SYSTEM

A DOA system is basically a distributed information system. The problem of developing distributed information systems should be addressed from two points of view: First, the requirements must be clearly identified, and then the best architecture and technologies to satisfy the requirements must be identified [3].

The Requirements

The DOA system should pave nearly all the facilities that a static assessment system offer, to student, faculty, and to the administration. That means, students should be able to take the assessment anywhere, but not anytime as the examiners must be present during the examination. The examiners must be able to prepare questions before hand and store it in a database, but they must administer the exam (deliver questions) real-time. After the exam, the results are shown instantaneously to the student; in addition, the results are stored in the Web server and processed automatically for administrative purposes. This means, there exist three phases: a *pre-examination phase* for preparations of the tests, an *examination phase* during the course of the examination, and a *post- examination phase* during which the system automatically does the administrative processing.

The Architecture

From the requirements stated above, architecture for the system can be devised without much difficulty. The proposed architecture is shown in Figure 2.

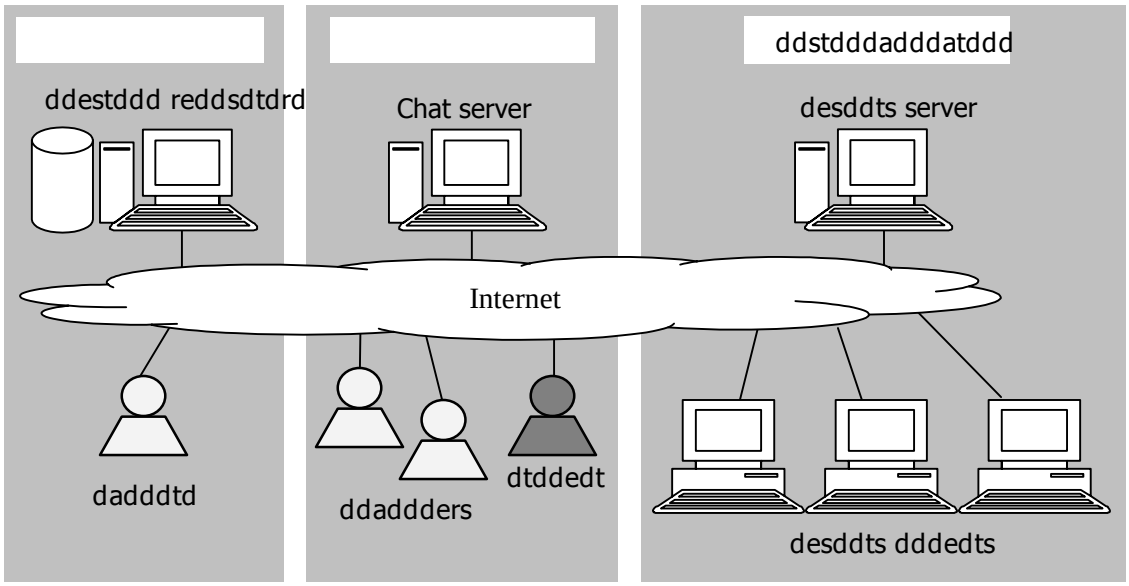

Figure-2: The three phases of the dynamic online assessment system

Pre-examination phase. This is an offline mode in which the faculty prepare their exam questions, save, and edit. The faculty uses a question repository and some editing tools for making and storing the questions.

Examination phase. This is an online mode of the assessment system. During online testing, a student is under assessment by one or more examiners. All the participants (student and examiners) are connected by a chat system. At the end of the test, the chat system stores all the conversations, and the results in a commonly accepted format.

Post-examination phase. After the assessment, the results are stored and processed. A Results server functions as a service provider for those who want to retrieve the results. For example, a staff (a Results client) may seek result a student obtained on a test for a particular course by sending <student ID, course ID> message to the Results Server; a lecturer (another Results client) may want to extract the results of all the students who took his course, by sending <course ID> message.

The Enabling Technologies

Java technology and XML standard make a great combination for implementing distributed information systems. XML enables disparate software systems to exchange messages. Java enables implementation of distributed information systems in a methodical way. Basic development software for both Java and XML are available free of charge, hence upfront investments can hardly be an argument against starting a project with XML and Java.

As Java together with XML provides several technologies for integration, designers face the challenge of combining the technologies in the most effective and flexible manner possible to create an integration architecture that is opt for future extensions and changes. The best place to start the design process is the design templates.

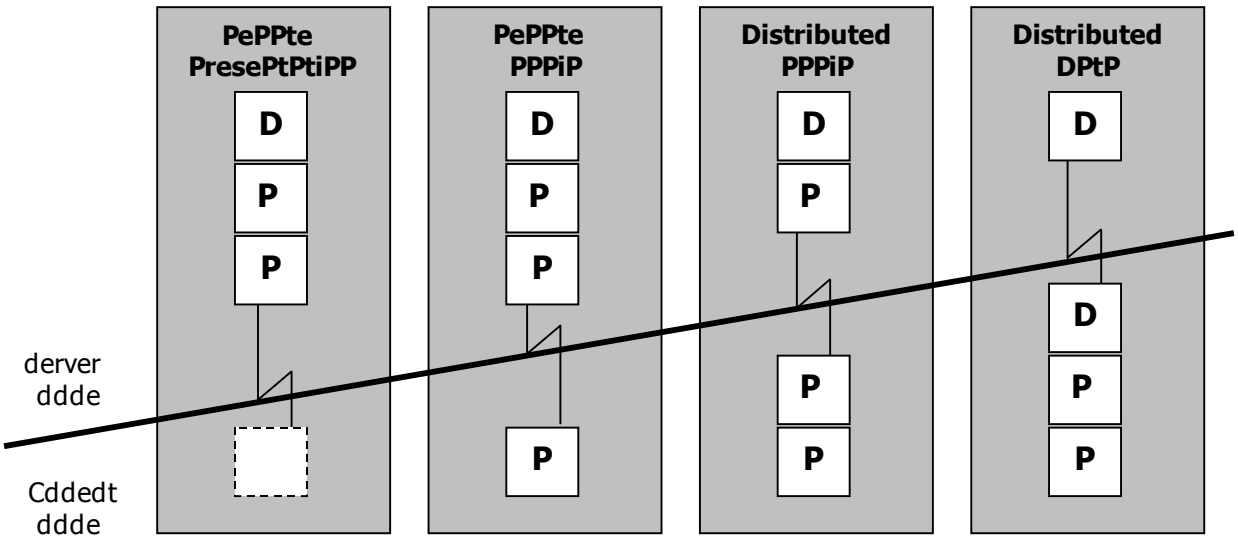

Figure-3: Design templates for building distributed information system

The Design Templates

Figure 3 shows the basic design templates (models) from where the implementation of the three phases of the DOA system can start [1, 4]. A very brief interpretation of the models shown in Figure3 follows:

If the client is a thin client, meaning it uses only the Web browser to interact with the server, then the server must do all the work inclusive the presentation for the client (therefore 'remote logic'). Since the client is armed only with Web browser, client – sever interactions are based on lightweight HTTP (Hyper Text Transfer Protocol) protocol, and on the server-side, the server normally uses a servlet or JavaServer Page to create the view logic for the client [2, 9]. If the client is running an application (not just a Web browser), then the presentation for the client can be done by the client application itself and the server performs the business logic only (remote logic). In this case, the client can use heavyweight protocol like RMI-IIOP (Remote Method Invocation – Internet Inter-Object Request Broker Protocol) to interact with the server and the server may employ an Enterprise Java Bean as the server-side component to work together with the client [4].

In 'distributed logic', the client is a thick client, meaning both the client and the server do the business logic computations. For client – server interaction, asynchronous messaging, for example, could be used, as messaging is an excellent way of loosely coupling a client and a server. If both the client and the server are Java applications, then Java Messaging Services (JMS) API could be used; otherwise, the server could employ (assuming that the server is a Java application) Message-Driven Bean to work with the enterprise messages that the client is sending. Finally, in 'distributed data' model, both the client and the server are independent business applications. The best way for integration is the Web services; Web service is the most loosely coupled way of integrating a client and a server where (XML based) SOAP messages are exchanged [6].

The client-server models described above will be considered during the implementation process.

BUILDING A DOA SYSTEM

This section looks into the most suitable Java-XML based solution for developing a distributed information system to support DOA, as shown in figure-2. The suggestion given below is a simplified version of a testing prototype under construction.

Pre-examination Phase

Ignoring elaborate editing facilities, the faculty uses Web browser to make, edit, store, and retrieve text-based exam questions in a database (remote view). Hence, a servlet on the server-side is all that is needed to implement this phase [2, 4].

Examination Phase

An Internet-based chat system is needed to enable geographically distributed individuals (the candidate and members of the exam panel). The chat system primarily deals with exchange of text messages (data migration) between a limited numbers of participants; hence, simple TCP (Terminal Control Protocol) socket based multithreaded chat server program and client programs

will be satisfactory [4]. This means, all the individuals involved must download the chat client program into their computers first, and use this program during the examination. At the end of the test, the chat server store all text-messages exchanged as an XML document (Figure 4).

```
<?xml version="1.0"?>
<xs:schema id="NewDataSet" targetNamespace="http://tempuri.org/exam.xsd"
      xmlns:mstns="http://tempuri.org/exam.xsd"
      xmlns="http://tempuri.org/exam.xsd"
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
      attributeFormDefault="qualified" elementFormDefault="qualified">

  <xs:element name="transcript">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="courseID" type="xs:string" minOccurs="1" />
        <xs:element name="studentID" type="xs:string" minOccurs="1" />
        <xs:element name="start-time" type="xs:dateTime" minOccurs="1" />
        <xs:element name="stop-time" type="xs:dateTime" minOccurs="1" />
        <xs:element name="result" type="xs:string" minOccurs="1" />
        <xs:element name="dialogue" minOccurs="1">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="examiner" minOccurs="1"
                                          maxOccurs="unbounded">
                <xs:complexType>
                  <xs:simpleContent msdata:ColumnName="examiner_Text"
                                                msdata:Ordinal="0">
                    <xs:extension base="xs:string">
                    </xs:extension>
                  </xs:simpleContent>
                </xs:complexType>
              </xs:element>
              <xs:element name="student" minOccurs="1"
                                          maxOccurs="unbounded">
                <xs:complexType>
                  <xs:simpleContent msdata:ColumnName="student_Text"
                                                msdata:Ordinal="0">
                    <xs:extension base="xs:string">
                    </xs:extension>
                  </xs:simpleContent>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="NewDataSet" msdata:IsDataSet="true"
              msdata:Locale="nb-NO" msdata:EnforceConstraints="False">
    <xs:complexType>
      <xs:choice maxOccurs="unbounded">
        <xs:element ref="transcript" />
      </xs:choice>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Figure-4: XML Schema for test transcripts

Post-examination Phase

The post-examination phase can be realized as a simple database application. First, the test transcripts produced after the examination are parsed using JAXP API and the details are stored in a relational database. Alternatively, it is also possible to store and retrieve XML documents as a whole using third-party software; but third-party software (database and drivers) comes with a price tag. Implementing the results server as a simple database application enables use of servlets (together with entity beans, if needed) to serve thin clients (clients using just the Web browser).

MANAGERIAL IMPLICATIONS

This paper shows that software for online education can be easily built using Java technology and XML standard; all we need is a methodical approach. The most widely used online education software in Norway called "Its-learning" (http://www.itsolutions.no) was actually developed by a group of students as their project work. Nearly all the universities in Norway now use Its-learning as their Web-based online education software.

REFERENCES

1. Brown, D. (1997). Writing Web-based questions with Mallard. *Frontiers in Education Conference, 27th Annual Conference on Teaching and Learning in an Era of Change, 3,* Salt Lake City, UT.
2. Davidrajuh, R. (2004). A Survey on the Java Technologies for Developing Distributed Information Systems, *IBIM 2004 Conference*, Amman – Jordan, July 2004.
3. Fox, G. (March-April 2004). Software Development Around a Millisecond. *Computing in Science & Engineering,* IEEE CS and the AIP.
4. Hughes, M., Shoffner, M., & Hammer, D. (1999). *Java Network Programming*, 2ed, Manning, Greenwich
5. Ko, C. & Cheng, C. (2004). Secure Internet examination system based on video monitoring. *Internet Research, 14*(1), 48-61.
6. Manes, A. (2003). *Web Services: A Manager's Guide*, Addison-Wesley
7. McGough, J., Mortensen, J., Johnson, J., & Fadali, S. (2001). A Web-based testing system with dynamic question generation. *ASEE/IEEE Frontiers in Education Conf, 3*, S3C-23-8.
8. Oviatt, S., & Cohen, P. (1989). *The Effects of Interaction on a Spoken Discourse*. SRI International, CA.
9. Reilly, D. & Reilly, M. (2002). *Java Network Programming and Distributed Computing*, Addison-Wesley
10. Steflik, D. & Sridharan, P. (2000). *Advanced Java Networking*, 2ed., Prentice Hall PTR
11. Tartaglia, A. & Tresso, E. (2002). An automatic evaluation system for technical education at the university level. *IEEE Transactions on Education, 45*(3), 268-75.
12. Thelwall, M. (2000). Computer-based assessment: a versatile educational tool. *Computers & Education, 34*, 37-49.