MOLE: An Extension to MLE Moodle

Hemant Sakharkar, Sridhar Iyer and Malathy Baru

Department of Computer Science and Engineering
Indian Institute of Technology Bombay
hemantsakharkar@it.iitb.ac.in
sri@iitb.ac.in
malati@it.iitb.ac.in

***Abstract*— Learning in the class can be augmented by using learning management systems, either on PCs or mobiles. This enables 'any time, any where, any device' access to the Learning content. As mobile devices become increasingly affordable for student population, any tool on this platform will have more reach than a PC. It will also translate to increased number of beneficiaries. In this paper we discuss some existing learning management tools for mobile and the issues that arise in network constrained areas. We specifically consider an existing system MLE Moodle[3], and present a modification to its architecture for resource and network constrained environments.**

I. Introduction

Learning through alternative channels can supplement and augment classroom learning. With the advent of Internet, access to online content is taken for granted. In developing countries, sustained Internet connectivity is still a major challenge for both rural and urban areas. Hence though Internet based learning systems are available, access to these systems across the student population is still limited. Cost is another major factor as many systems need subscriptions to access content. We believe that it is critical to look at alternative channels for disseminating learning material.

As the mobile penetration[11] is way above computer penetration in India, it makes economic sense to explore how mobile devices can be used as the platform for learning systems. Mobile devices are also an attractive alternative channel, as they become increasingly affordable for student population. Any tool on the mobile platform will have more reach than a PC. This does not mean that the desktop learning system becomes redundant. It becomes the backbone and the mobile devices serve as the front end. The major advantage of using the mobile is that it provides the flexibility of accessing the system from anywhere.

However, mobile devices cannot display large volumes of rich multimedia content, due to screen size and storage limitations. So mobile devices are not suitable for a student learning a concept for first time. Nevertheless, we believe that mobiles can be useful for revision of content and practice tests. Once the student has learnt the subject either in a classroom or from a book or using a PC, he/she requires practice quizzes to master the subject and score in the exams. Since the mobile device is ubiquitous, it is desirable to conduct quizzes using mobiles, thereby helping students to practice anywhere.

In this paper we present the pros and cons of some existing learning systems for mobiles and details of a popular one, MLE Moodle[3]. We discuss the alternative approaches for resource constrained environments along with a detailed theoretical analysis. Finally, we propose some modifications to MLE Moodle to adapt it to constrained environments.

Section II presents comparison of some of the existing mobile based learning tools. Section III presents an overview of MLE. Section IV discusses the concerns and Section V presents some alternative. Section VI gives architecture of the proposed implementation.

II. Study of existing learning systems on mobile

M-Learning is the term coined for learning on the mobile and is defined as “Learning that happens across locations, or that takes advantage of learning opportunities offered by portable technologies[6]” There are a number of mobile based learning systems available today. We have studied some tools like Wizdom.in[8], MLE Moodle[3], Mobile ELDIT[9] and evaluated the quiz management component of these tools. Some of the parameters on which the tools were evaluated are: (i) Open source - whether they are open source or not, (ii) Type of questions supported - Multiple choice, Multiple answers, True/False and Short answers, (iii) Automatic grading, (iv) Hints – whether hints and correct answers are displayed, (v) Type of Multimedia – whether images, audio, video, (vi) Chat support, (vii) Forum support, (viii) Wiki, (ix) Support for adaptive learning, (x) Roles supported on mobile, (xi)Mobile phone supported.

The features of various tools as per the above parameters is tabulated in appendix. We omit a detailed discussion due to lack of space. Most of these tools are proprietary, hence the end user pays not only the cost of the tool but also for connectivity on mobile to access information.

Moodle[1] is the most popular open source learning management system. It has both a desktop component and a mobile component. The details of Moodle are discussed below.

III. Moodle & MLE

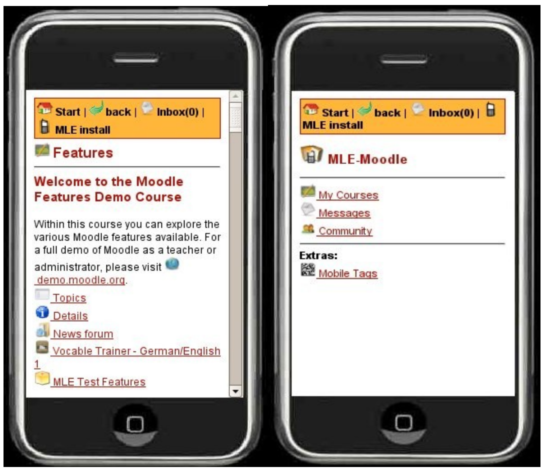
The word Moodle was originally an acronym for Modular Object-Oriented Dynamic Learning Environment. It is based on the social constructionist framework of learning. It is released under the GNU Public License GPL[12]. It is used for producing Internet based courses and websites.

As of January 2009, there are more than 55,000 moodle sites and it continues to have average downloads of more than 70,000 per month. (http://moodle.org/stats/d). It is currently available for many flavors of Linux and also Windows and Mac.

Many institutes and universities use Moodle for their course management. Indian Institute of Technology Bombay[4] has also been using Moodle for many years. Students use this tool for collaborative learning and faculty find it useful as a platform to connect to the students of their respective courses.

Moodle has numerous features[2] like lessons, assignments, forums, quizzes, resources, and Wiki. Moodle has now been extended to the mobile and is called MLE: Mobile learning extension.

MLE[3] is available in two versions. The first version is browser based and can be accessed directly through the mobile browser. In this version the same content available on the Moodle server is presented to the user on a different set of interfaces (which have been built exclusively for mobiles). The interfaces are very intuitive, easy to navigate and the learning curve is minimal. Some screenshots of MLE, accessing the Moodle server using the web-browser on the phone are shown in Fig 1:


Start | back | Inbox(0) |
MLE install
Features
Welcome to the Moodle Features Demo Course
Topics
Details
News forum
Vocable Trainer - German/English 1
MLE Test Features
Start | back | Inbox(0) |
MLE install
MLE-Moodle
My Courses
Messages
Community
Extras:
Mobile Tags

Within this course you can explore the various Moodle features available. For a full demo of Moodle as a teacher or administrator, please visit demo.moodle.org.

Fig 1:MLE on mobile browser screen

The second version of MLE is a J2ME application which has a client component and server component. The application is available for different mobile devices like Nokia, Samsung Motorola, as the device configuration is different for each mobile. The application needs to be installed on the mobile and the client component communicates with the server component for data transfer. Some data is locally stored on the mobile, but as data storage and caching is limited on a mobile, the bulk of the content is transferred back to the server. A Gateway server (placed between the Moodle server and mobile device), optimizes the data from the server for easy display on the mobile screen. Some shots of mobile client component of MLE are shown in Fig 2.

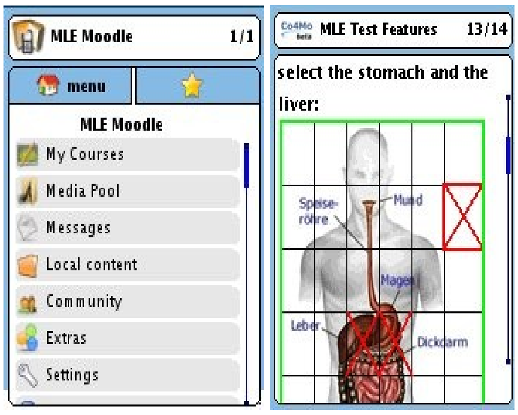


Fig 2: MLE mobile client screen

MLE Moodle has all the features supported by Moodle. We installed and tested the MLE Moodle on some smart phones like N95, E60 and found it to be working satisfactorily. Also the J2ME client works on almost all Java enabled phones. Hence we chose MLE as a tool for our system.

IV. Concerns and issue with MLE

In this paper, we focus on issues and solutions for the Quiz engine of MLE. The Quiz engine performance depends upon number of questions in the quiz and affects these following factors.

1. Time: $time \propto questions$
   The time to display the first question is directly proportional to number of questions in the quiz. As the number of questions increases, the time to download data and its parsing, also increases.
2. Memory: $memory \propto questions$
   The memory requirement is also directly proportional to number of questions in quiz.

The above problems arise because of structure of XML schema used in MLE. The current implementation is such that all questions in quiz are represented using a single XML Structure. The sample XML schema is as shown in Fig 3.

```
<pagesuite title="Question 1">
<page>
A question with a textbox:<br>
<question id="q1" points="10">
How much is 1 plus 2:<textbox id="text1" number="true">
<answer solution="3" answer="1+2 is 3&lt;br&gt;if you have one apple and add another two apples, you get three apples." points="10">
</textbox>
<br>
</question>
</page>

<page>
A single choice question:<br>
<question id="q2" points="12">
What does MLE stand for:<br>
<checkbox id="c1" group="g">
<answer solution="0" answer="My Learning Engine is just an invention." points="12">
</checkbox> My Learning Engine<br>
<checkbox id="c2" group="g">
<answer solution="1" answer="MLE stands for Mobile Learning Engine." points="12">
</checkbox> Mobile Learning Engine<br>
<checkbox id="c3" group="g">
<answer solution="0" answer="Mobile Learning Equipment is just an invention." points="12">
</checkbox> Mobile Learning Equipment<br>
</question>
</page>
</pagesuite>
```

Fig 3:Client-Server Model without prefetching

Note that these questions are all in the same <pagesuite>. Hence they are all downloaded onto mobile and parsed fully, before displaying the first question. This increases the user's waiting time and the storage required on mobile. Moreover, this gets aggravated for networks with bandwidth constraints, as well as devices with resource constraints.

These considerations led us to explore alternatives approaches for environments with network and resource constraints.

V. Alternative approaches

Two obvious approaches for downloading quiz from the server are: (i) *One by One approach*: The questions are downloaded, one at a time, only after individual requests from the user, or (ii) *All in One approach*: the entire quiz is downloaded at the start and then displayed.

In One by One approach, the application downloads the first question and shows it to the user. The next question is downloaded only after the user completes answering the first and requests for the next question. Hence the disadvantages are:

i. After answering a question, the user has to wait (idle), till the next one is downloaded.
ii. If the connectivity is not consistent or there are network errors, the user may have to try multiple times for download of each question.

In the All in One approach, the application downloads the entire quiz on the mobile at the start of quiz. This addresses the disadvantages mentioned above. However, it has its own disadvantages of:

i. If quiz is quite large then storing the entire quiz on the mobile is problem. Some mobiles may not be able to store entire quiz.
ii. The time require to download the entire quiz is time consuming, which increase the initial startup time. There is also the probability of network disconnect during such a long transfer.

Hence it is clear that an intermediate approach of downloading a given number of questions (k) at each request, might be more suitable. Moreover, this could be combined with pre-fetching of the next set of questions, while the user is answering questions already downloaded. However, it is necessary to determine a suitable value for this number (k).

A theoretical analysis of the above techniques is done in section A.

A. Theoretical Analysis

In theoretical analysis we consider the client server model as shown in fig 4:

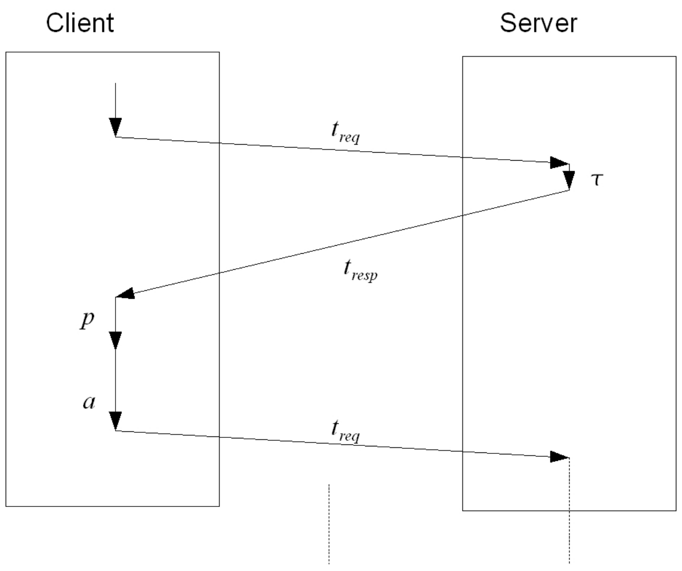


Fig 4:Client-Server Model without prefetching

The parameters considered for analysis and other terms are shown in Table 1 below.

TABLE I
PARAMETER AND DESCRIPTION

Para-meter	**Description**
t_{req}	Time required for client to create a request and transmit over the network to the server. *[Assumed to be constant]*
τ	Time required for server to create the response. This includes: (i) the time required to parse client request [*assumed constant*], and (ii) the time required to assemble the desired number of questions (k) into a response message [*depends on k*].
t_{resp}	Time required to transfer response from server to client [*depends on k*].
p	Time required for the client to parse the response message [*depends on k*].
a	Time required by user for answering questions [*depends on k*].
d	Time taken between display of two consecutive questions [*depends on approach]*
T	Total time required for the user to complete answering all the questions.
n	Total number of questions.
α	Size of a question (KB) [*assumed constant*].
β	Bandwidth of the wireless network (in Mbps) [*assumed constant*].
l_{req}	Size (in KB) of storage required to create the request message [*assumed constant*].
l_{quest}	Storage required (in KB) to store questions on client [*depends on k*].
l_{ans}	Size of storage (in KB) required to create answer message [*assumed constant*].
L	Total size of storage required on client.
w	Window size (for pre-fetching scheme).

Note that:

$$t_{resp} = \frac{k \times 8 \times 1024}{(1024 \times 1024)} \left(\frac{\alpha}{\beta}\right) = \frac{k}{128} \left(\frac{\alpha}{\beta}\right) \quad \text{--- Eq. (1)}$$

where t_{resp} in sec.
When k =1

$$t^1_{resp} = \frac{1}{128} \times \left(\frac{\alpha^1}{\beta}\right) \quad \text{--- Eq. (1.1)}$$

t^1_{resp} is the time taken for one question to transfer from server to client. As α^1 (size of question) is assumed to be constant, t^1_{resp} is constant. Also, τ^1 (time required for server to create the response with a single question), p^1 (time required to parse a single question) and a^1 (time required to record the answer for a single question), are all assumed to be constant.

Now consider k questions being sent for each request. Hence for n questions, the number of requests made by client will be $\left\lceil \frac{n}{k} \right\rceil$. Hence the total time and storage requirements are:

$$T = \frac{n}{k}(t_{req} + k\tau^1 + k t^1_{resp} + k\,p^1) + na^1 \quad \text{-Eq. (2)}$$

$$L = l_{req} + k\,l^1_{quest} + l_{ans} \quad \text{--- Eq. (3)}$$

where l^1_{quest} is storage for single question.

Now we illustrate each approach considering a example. We will assume following values for calculation.

TABLE II
PARAMETER AND VALUES

Parameter	**Values**
t_{req}	20ms
τ^1	5ms
p^1	100ms
a^1	1000ms
n	10
α	100KB
β	2Mbps
l_{req}	5KB

From Eq. (1.1), we get $t_{resp}{}^1$= 390ms.
Now assume $1^1{}_{quest} = l_{ans} = \alpha$

The various times for the different approaches are computed as follows:

1) One by One approach:

In case of one by one, where user has to request explicitly for each question, substituting k=1 in Eq. (2) and (3) we get total time T_o and storage requirement L_o as:

$$T_o = n(t_{req} + \tau^1 + t^1_{resp} + p^1 + a^1) \quad \text{-- Eq. (4)}$$

$$L_o = l_{req} + l^1_{quest} + l_{ans} \quad \text{-- Eq. (5)}$$

$$d_o(i) = t_{req} + \tau^1 + t^1_{resp} + p^1 + a^1 \quad \text{-- Eq. (6)}$$

where i = i^th question.

Note that the storage required on mobile is constant and contains only one question. Also, delay between display of consecutive questions is constant.

Substituting values from Table II in Eq (4), (5), (6) we get:

T_o = 15150 ms
L_o = 205 KB

$d_o(1)$=1515 ms
$d_o(2)$=1515 ms
$d_o(i)$=1515 ms

2) All in One approach:

In case of all in one, all the questions are downloaded on single request before starting the quiz. Therefore substituting k=n in Eq.(2) and (3) we get total time T_a and storage requirement L_a as:

$$T_a = t_{req} + n(\tau^1 + t^1_{resp} + p^1 + a^1) \quad \text{-- Eq. (7)}$$

$$L_a = l_{req} + n\, l^1_{quest} + l_{ans} \quad \text{-- Eq. (8)}$$

The delay between display of consecutive questions is:

$$d_a(i) = t_{req} + n(\tau^1 + t^1_{resp} + p^1) \quad when \quad i = 1$$
$$d_a(i) = a^1 \quad when \quad i > 1 \qquad \text{--- Eq. (9)}$$

where i= ith question.

Eq. (9) shows that the time required to display the first question depends upon the number of questions, whereas when i>1, time required to advance to next question is constant as a^1.

Substituting the values from table II in Eq(7), (8), (9) we get

T_a = 14970ms
L_a= 1105KB
$d_a(1)$=4970ms
$d_a(2)$=1000ms
$d_a(3)$=1000ms

3) Prefetching approach:

Now to analyze the prefetching technique we have to consider the model as shown in Fig 5:

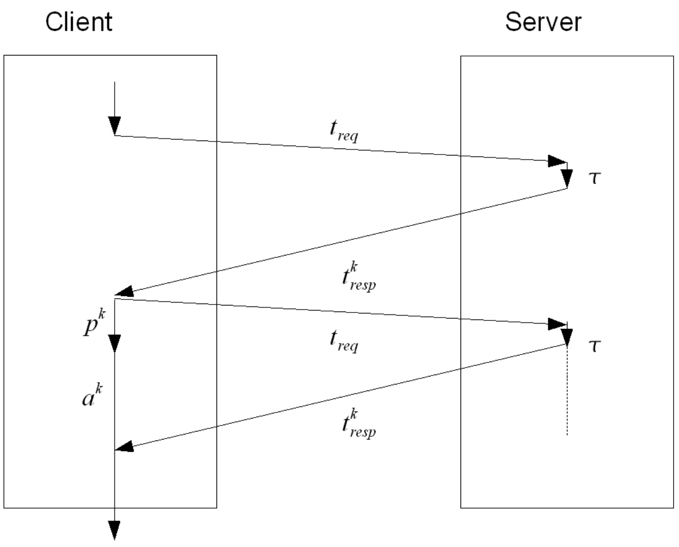


Fig 5:Client-Server Model for prefetching

In prefetching technique when k questions are downloaded, the user starts answering them. During this period, the next set of k questions are concurrently fetched from the server. Since these two activities are concurrent, the total time needs to take into account only the activity which takes longer. In other words, the activity which take less time will be eliminated from the total time calculation. We now modify equations (2) and (3) as shown below.

$$T_{prefetch} = \frac{n}{k}(t_{req} + k\tau^1 + k\, t^1_{resp} + k\, p^1) + \left(\frac{n}{k}\right) k\, a^1 - \left(\frac{n}{k} - 1\right) e \qquad \text{-Eq. (10)}$$

where e= time eliminated due to prefetch.

Now in the case when the time taken by the user to record k answer is more than time taken to transfer the next set of k questions, i.e., when

$$k\, a^1 > (t_{req} + k\tau^1 + k\, t^1_{resp} + k\, p^1)$$

we get e= $t_{req} + k\tau^1 + k\, t^1_{resp} + k\, p^1$

Substituting e in equation (10) we get

$$T_{prefetch} = t_{req} + k\tau^1 + k\, t^1_{resp} + k\, p^1 + n\, a^1 \quad \text{-Eq. (11)}$$

On the other hand, in the case when the time taken to transfer k questions is more than time taken by user to record the answer, i.e., when

$$k\, a^1 < (t_{req} + k\tau^1 + k\, t^1_{resp} + k\, p^1)$$

we get $e = k\, a^1$.

Substituting e in Eq. (10) we get

$$T_{prefetch}=\frac{n}{k}(t_{req}+k\,\tau^1+k\,t^1_{resp}+k\,p^1)+k\,a^1 \quad \text{Eq(12)}$$

The delay between the display of consecutive questions is:

$$d_{prefetch}(i)=t_{req}+k\,\tau^1+k\,t^1_{resp}+k\,p^1 \quad where\, i=1$$
$$d_{prefetch}(i)=a^1$$
$$where\quad i>1\,,\quad k\,a^1>(t_{req}+k\,\tau^1+k\,t^1_{resp}+k\,p^1)$$
$$d_{prefetch}(i)=(t_{req}+k\,\tau^1+k\,t^1_{resp}+k\,p^1)/k$$
$$where\quad i>1\,,\quad k\,a^1<(t_{req}+k\,\tau^1+k\,t^1_{resp}+k\,p^1)$$

-- Eq. (13)

Now in order to analyze storage requirement for the prefetching technique, assume that m prefetches (of k questions each), can occur while the user is answering the first set of k questions. Hence there are m requests during this time. To compute L, the storage required by application, we modify Eq (3) as

$$L=l_{req}+m\,k\,l^1_{quest}+l_{ans} \quad \text{--Eq. (14)}$$

When m=1, no data is prefetched by application and m>1 means data is prefetched and stored by application.

As l_{req} & l_{ans} is negligible in terms of $m\,k\,l^1_{quest}$ so

$$L_{prefetch}\approx m\,k\,l^1_{quest} \quad \text{--Eq. (15)}$$

Thus we get the storage on the mobile (receiver window size), in terms of number of question we have

$$w\approx m\times k \quad \text{--Eq. (16)}$$

Now we consider a special case when k=1 which means on each request only one question will be sent in response. We assume that time required to record the answer (a) is more than transfer the question ($t_{req}+\tau+t_{resp}+p$). Substituting k=1 in Eq (11), we get

$$T^1_{prefetch}=t_{req}+\tau^1+t^1_{resp}+p^1+n\,a^1 \quad \text{-Eq. (17)}$$

The delay between display of two consecutive questions is:

$$d(i)^1_{prefetch}=t_{req}+\tau^1+t^1_{resp}+p^1 \quad where\, i=1$$

$$d(i)^1_{prefetch}=a^1$$
$$where\, i>1\,,\quad a^1>(t_{req}+\tau^1+t^1_{resp}+p^1) \quad \text{-Eq. (18)}$$

Substituting values from table II in Eq. (17) and (18), and assuming w = 5, we get:

$T^1_{prefetch}$ = 20+5+390+100+10*(1000) = 10515ms

$L_{prefetch}$ = 5+ 5* 100+100= 605KB

$d^1_{prefetch}(1)$= 515ms

$d^1_{prefetch}(2)$= 1000ms

$d^1_{prefetch}(3)$=1000ms

Since this delay is equal to the time taken to record the answer, the user does not experience any delay (or wait) between display of questions.

We show an example to compare and summarize the techniques discussed above, in section B.

B. Example for comparison of techniques

We compare the above techniques in a time line, assuming ten questions in the quiz. Time line is as shown in fig 6.

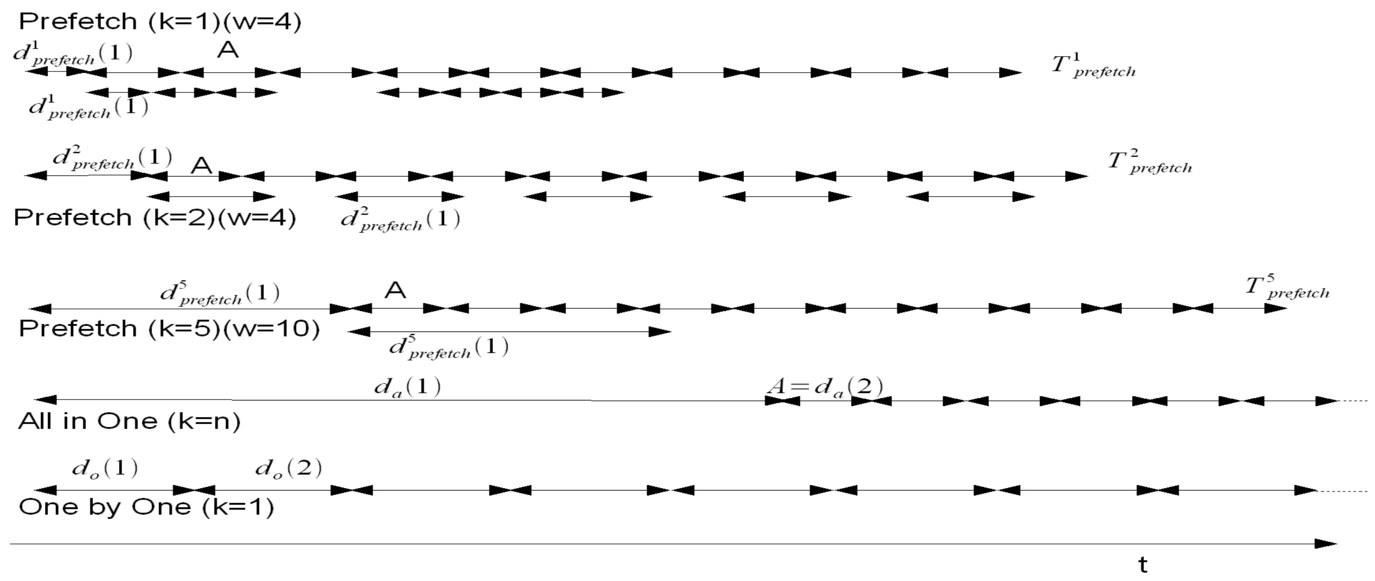


Fig 6:Time line for all techniques

Now for comparison we considered start time, delay between questions, total time & waiting time of user. In latency experience by user which eliminate the time to record the answer from delay equations. Comparisons are tabulated in table III.

we assume $a^1 > (t_{req} + \tau^1 + t^1_{resp} + p^1)$

TABLE III
COMPARISONS OF TECHNIQUES

Technique	**Start time**	**Delay between question**	**Total time**	**Waiting time for user**
One by One	515ms	1515ms	15150ms	515ms
All in One	4950ms	1000ms	14970ms	0ms
Prefetch				
(k=5,w=10)	2495ms	1000ms	12495ms	0ms
(k=2)(w=4)	1010ms	1000ms	11010ms	0ms
(k=1)(w=4)	515ms	1000ms	10515ms	0ms

Now we discuss the storage size required on mobile using above techniques. In one by one technique, from Eq. (6) only one question is stored. In All in one technique, from Eq. (9) which says the size is dependent on number of questions. In prefetching, from Eq. (16) we see that as k increases the storage size increases and start time also increases. In this case it is better to keep k at the minimum value.

Now for w (size of window), observe that low value of w will have low memory requirement but it will require a reliable network. On the other hand, for intermittent connected network, w should be high. Since high w increases storage on mobile, there is trade off between storage size and network reliability.

In our chosen technique for implementing in MLE, k and w are configurable parameters with default values of 1 & 4 respectively. We propose to implement a sliding window protocol [7] to fetch multiple questions.

The State Diagram in Fig 7, shows what can be prefetched and when. When user authenticates, the application is accessed. The user has to select an option from a given list. This time cannot be used for prefetching since the content is dependent on user request. When the user starts the quiz, the number of questions does not depend on user request, so such content can be used for prefetching.

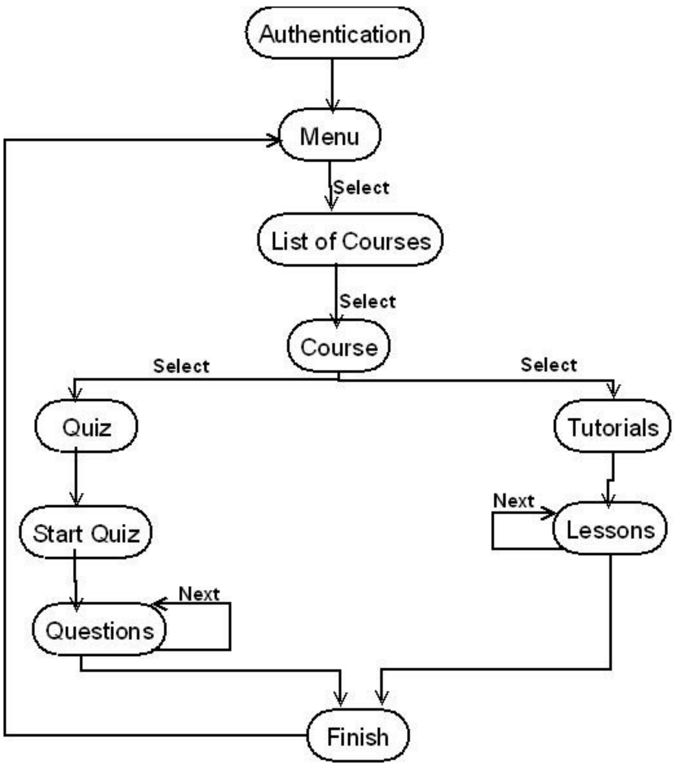


Fig 7: State diagram of mobile client

To implements this technique in existing MLE MOODLE we propose modifications to the existing system in next section.

VI. Implementation of prefeching scheme MOLE

A. System architecture

The three tiers and their components already implemented in MLE are tabulated in table IV. Our proposed architecture for Quiz engine is shown in Fig. 8.

TABLE IV
SYSTEM IMPLEMENTATION

Tier	**Components**	**Implementation Technology**
Front	PC browser, mobile browser, mobile app	XHTML, XHTML MP, XML, J2ME
Middle	Security, access control, application processing	Web server, PHP
Database	Database server	MySQL, SQL

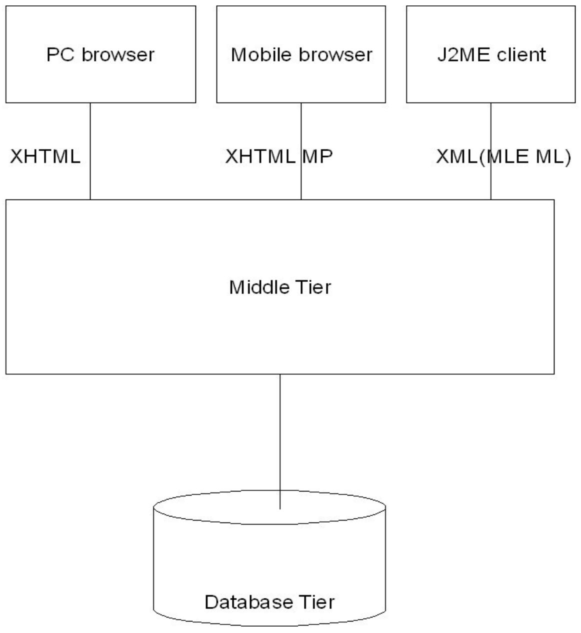


Fig 8: System Architecture

The details of the three tiers in existing application are:

1) Database tier - which provides storage and access of system data.
2) Middle tier - which is web application hosted on web server provides security, application processing logic and uses data from database for processing.
3) Front tier- It is a client which is accessing the Web server. In our application we propose support of three kinds of client.
   - PC browser- which supports web 2.0 applications, the data is transferred in XHTML[13] format.
   - Mobile browser - which supports WAP 2.0 application, the data is transferred in XHTML MP[5] a subset of XHTML
   - Mobile application - which will create using J2ME and data is transferred in XML(MLE ML) structure.

The architecture of the mobile client is discussed in section B.

B. The Architecture of mobile client

We have proposed three kind of access to Quiz engine, viz., PC based, Mobile browser based, and J2ME client application. We propose a modified architecture for mobile client as shown in Fig 8. It has Asynchronous machine, Direct machine and existing MLE J2ME client.

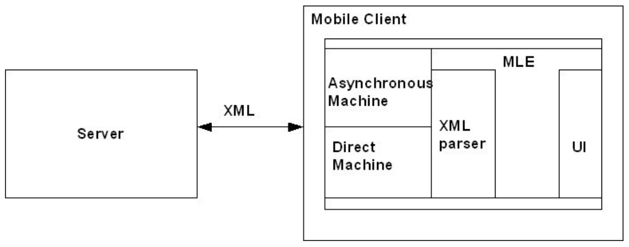


Fig 9: Architecture of mobile client

The details of the various components are given below:

1) Asynchronous machine

The Asynchronous machine is used for prefetching the data. It is a new component to be incorporated into MLE. As discussed above, while the user is answering the quiz, new questions will be prefetched. The sliding window protocol will be implemented when multiple questions are to be prefetched. The number of questions will be determined by size of window, which in turn depends on configuration and storage of mobile.

2) Direct machine

The Direct machine is already present in MLE. It has two functions. One function is fetching data based on user request. When user clicks on the link which requires fetching that data, direct machine will be used. Eg. If a user clicks on any link in list of “Topics” the contents of Topic to be shown depends on the user navigation. The second function of Direct machine is to convert user response to XML for sending back to server.

3) XML parser

The XML parser is already present in MLE. It is used to parse the data from the Asynchronous and Direct Machines before display to the user.

4) Data Transfer

The data transfer between client and server in existing MLE Moodle takes place using XML, known as MLE-ML. The MLE-ML contains only presentation data which is used for presenting data on UI of the device. The structure of MLE-ML is as shown earlier in Fig 3. As mentioned eearlier, this structure where a <pagesuite> tag contains all the <page> tags, results in all questions being downloaded before displaying the first page.

In our implementation, we divide the structure into smaller structures, such that questions can be displayed as they arrive at the client.

```
 <page>
        Question 1
</page>
<page>
       Question 2
</page>
...
```

Now the pages are divided into smaller tags that contain the presentational data. We need to embed some identification in each request made by the asynchronous machine for prefetching. We embed <id> tag to each <page> tag and all <id> tags of <page> are added in <pagesuite>. Now the asynchronous machine fetches question with help of <id>. Also we have to send back the answers to server using asynchronous machine, so we add another tag <Async-response>.

An example of the modified XML schema is shown below.

```
<pagesuite title =”quiz”>
        <data>
                <Async-response>yes</Async-response>
                <packets>
                        <id>1</id>
                        <id>2</id>
                        <id>3</id>
                        <id>4</id>
                </packets>
        </data>
</pagesuite>
<page>
        <data>
                <id>1</id>
        </data>
  Question 1
</page>
<page>
        <data>
                <id>2</id>
        </data>
Question 2
</page>
<page>
        <data>
                <id>3</id>
        <data>
Question 3
</page>
<page>
        <data>
                <id>4</id>
        </data>
</page>
</pagesuite>
```

The development of modules to incorporate into MLE, as described above, is ongoing.

VII. Conclusion and Future work

With rapid strides in technology development, such as wireless communication, mobile devices have become more and more powerful and inexpensive today. They have become a necessity and are handy, portable and easy to learn. Hence we believe that such devices can be fully exploited to build learning systems for anytime, anywhere for learning.

Adaptability to m-learning systems will be successful only if we are able to encourage educators, technologist, learners, administrators to embrace mobile technology for the purpose of learning. A lot of research has begun focusing on using mobile technology; however there remains a vast lag in the adoption of these ideas by companies and adapting these tools in academia.

The Quiz engine of a learning management system is suitable for adapting to mobiles. In this paper, we have proposed and analyzed a simple modification to the MLE-Moodle's Quiz engine that can improve the user experience.

After the completion of our implementation, we plan to do rigorous impact assessments and implement further enhancements. We hope that large number of students having mobile devices will be beneficiaries of our system. We also envisage adapting the system to literacy programs in rural areas.

Acknowledgment

This work is being carried out under TICET project: “Tata Teleservices IIT Bombay Center of Excellence in Telecommuincation”. The authors would like to thank Vijaya Laxshmi & colleagues in the lab for their contributions.

References

[1] Moodle [online] http://moodle.org
[2]Moodle Features Demo [online] http://moodle.org/course/view.php
[3] MLE Moode [online] http://mle.sourceforge.net/ mle.php
[4]Moodle at Indian Institute of Technology Bombay [online] http://moodle.iitb.ac.in/
[5]XHTML MP [online] www.developershome.com/wap/xhtmlmp/xhtml_mp_tutorial.asp?page=introduction
[6]M-Learning Wikipedia [online] http://en.wikipedia.org/wiki/M-learning.
[7] Andrew S. Tannenbaum “*Computer Network”* 3rd Edition
[8] Wizdom [online] http://www.wizdom.in/wizdemo.html
[9] Mobi Eldit [online] http://www.trifonova.net/mobileEldit.php
[10] PC Penetration in India [online] http://www.mit.gov.in/download/CHAP5.PDF
[11] Mobile Penetration in India [online] http://www.pcworld.com/businesscenter/article/160007/india_shatters_monthly_mobile_subscriber_record.html
[12] GPL General Public Licence http://www.gnu.org/licenses/gpl.html
[13] XHTML {Online] http://www.w3.org/TR/xhtml1/
[14] Shiow-yang Wu, Jungchu Hsu, and Chieh-Ming Chen, "Headlight prefetching for mobile media streaming" in *Proc of 6th MobiDE '07*, pp 67 – 74, June 2007.
[15] Jin Jing, Abdelsalam Sumi Helal, Ahmed Elmagarmid "Client-server computing in mobile environments" *ACM Computing Surveys (CSUR)*, Volume 31 , Issue 2, pp 117 - 157, June 1999.
[16] Vincenzo Grassi "Prefetching policies for energy saving and latency reduction in a wireless broadcast data delivery system" in *Proc of MSWIM* '00, pp 77 - 84, 2000.
[17] Samir Abou El-Seoud1, Ashraf M. A. Ahmad1 and Hosam Farouk El-Sofany2, "Mobile Learning Platform Connected to Moodle using J2ME", *iJIM*, Vol 3, No2,pp. 46-54, 2009.

APPENDIX

Applications	*Open Source*	*Type of Questions*	*automatic grading*	*H & A*	*Multimedia support*	*Chat*	*Forum*	*wiki*	*Adaptive Learning*	*Mobile Phone Supported*	*Role*	*Other Resources*
Wizdom.in	no	MCSA, MCMA, T/F	yes	yes	images	no	no	no	yes	Java Enabled phones	students	
MLE Moodle	yes	MCSA, MCMA, T/F,SA	yes	yes	word, pdf, audio, video, flash	yes	yes	yes	partially	PDA, smart phones, Java Enabled phones	students	order question, flash card
MobiLP	can't say	MCSA,MCMA, T/F	yes	maybe	images	yes	yes	no	no	cell phones with browser, Desktop with browser	students, teachers	display of web material by teachers
MVClass (SJTU) & peking University	can't say	None	no	no	images, audio, video	yes	yes	no		PDA, tablets, Laptops, PCs	Students	Audio & video on demand on WLAN as well as Multimedia server
Maxdox	no, but free to create content on mobile	None	no	no	text & images static content	no	no	no	no	MIDP2.0	no roles	To view static offline multimedia & text content on mobile
Mobile ELDIT	no, but free ,content reading application	None	no	no	Text & images	no	no	no	no	PDA, Windows Mobile having browser & EWE package		Online & Offline access of text and Images on browser
University Mobile portal	can't say	None	no	no	no	no	no	no	no	cell phones	no roles	Only administration messages(SMS) are sended

Legend

MCSA-Multiple choice Single Answer	*Role* – Roles supported on mobile
MCMA-Multiple Choice Multiple Answer	*EWE*- A special virtual machine on PDA
T/F-True/False	*SJTU*-Shanghai Jiao Tong
SA-Short Answer	*ELDIT*- adaptable language learning platform
H & A – Hint & Answer	