CAN A DISTRIBUTED ARCHITECTURE BE APPLIED TO PROFILE-BASED e-LEARNING?

Student Name: Matthew Dyson
Supervisor Name: Andrei Krokhin

Submitted as part of the degree of Software Engineering to the
Board of Examiners in the School of Engineering and Computing Sciences, Durham University

Abstract **–**

Background **–** This research is investigating the link between profile-based e-learning and distributed systems. Profile-based e-learning is a rapidly emerging field, whereby a user's previous learning history is analyzed and compared to that of other similar users, in order to determine the learning material that will have the greatest effect. The majority of systems favor an individual approach, and currently there is no model for interaction between implementations on different networks.

Aims **–** The aim of this project is to examine the relationship between these two areas in order to determine whether widely implemented e-learning systems based around user profiles can be interlinked through the use of a distributed system in order to improve the quality of learning material suggested to the end user, forming a network whereby systems can co-operate and share learning material.

Method **–** In order to establish whether the above relationship is viable, I have designed and implemented a client/server architecture based around a series of documented Remote Procedure Call functions which could form part of a widespread learning network. I have then examined the results of the material selection algorithms in a number of different cases in order to determine whether this approach is effective.

Results **–** Through a series of experiments and surveys, user opinions were gathered on the learning network. The majority of participants agreed that the system was visibly adapting itself to their learning style, and had benefited from using it. The algorithms implemented as part of this project were also judged to be an effective method of applying metrics to learning data over a network.

Conclusions **–** The system developed in answer to the research question does clearly demonstrate the ability for these different areas of the e-learning discipline to be combined with encouraging results, however judging the impact this may have on learning would require a much larger test implementation, which in itself introduces additional issues.

Keywords **–** Distributed system, e-learning, profiling, networking, social learning

I. INTRODUCTION

The field of e-Learning is rapidly developing into many new and diverse areas, as the need within all scales of business and education increases. A vast number of establishments are realizing the benefits of educating their clients through the medium of computer based training, as opposed to the vastly more expensive classroom teaching, guided by research from large multi-national groups such as the European Distance & E-Learning Network (EDEN) [1]. Within this discipline, there are numerous different avenues that are explored in order to achieve the greatest benefit to the end user (the person in receipt of the learning material), including two examined within this project – a distributed e-Learning network, and user profiling. A distributed e-Learning network involves the separation of the learning material onto multiple inter-connected servers, and implementing expansive search algorithms to span the entire network to find the optimum piece of material to display to the user, which can be selected through a profile-based approach, effectively keeping a history of the users reactions to different types of learning material, and utilizing 'machine learning' to determine the best material to show to them in future, and to other similar users.

This report documents my final-year research and implementation of a distributed e-Learning network utilizing a profile-based framework.

A. Motivation

Within many businesses and educational establishments, people are expected to use Computer Based Training (CBT), and a lot of work has been put into this area. In the 2010 paper "Computer-based training: capitalizing on lessons learned" by Bedwell and Salas [2], the acronym 'ACTIONS' is used to describe how successful a CBT installation will be, which includes the cost, how capable the system is at teaching and how interactive it is, as well as other factors. One element highlighted within this framework is the ability to quickly add and update the system as new research and information becomes available, as it is pointless to be teaching outdated material to users. This is not a problem on a small scale implementation, as a single server or workstation can be easily updated as and when is necessary, however once a CBT (or e-Learning, within the scope of this project) system is rolled out to anything on a larger scale, this process of keeping information up to date becomes a lot harder – distributing content to multiple machines without putting a massive load on a single server is extremely difficult. For example, a large business with multiple sites internationally may wish to offer a training exercise about a software package they use internally, which is achieved through a series of videos provided to each site. Whenever the software package is updated, a new video is distributed to each remote site, who then have to host it on their own internal network to allow employees access. This process of content distribution puts minimal strain on their internal network by hosting files on-site rather than centrally, however means that updating the material takes much longer than necessary. By using a client/server system, this lengthy process of content distribution is eliminated, as the company would only need to use a single server to distribute content to their entire network. This can be further extended to include a server on each individual site linked to a master server in order to create a distributed network of servers, to allow for regional variations and also for local training material to be hosted.

This idea can be expanded by introducing 'user profiling' into the system. At a simple local level, user profiling looks at a user's learning history and algorithmically decides which content will be of greatest benefit to the user. For instance, a user who is a visual learner will respond best to a piece of material containing diagrams and pictures, whereas an auditory learner would gain very little from this (further discussion in [3]). This can be achieved in many ways, including direct feedback from the user, or by examination of the users understanding after trial of different approaches. By building up an accurate picture of how the user is responding to contrasting methods, and comparing this to the information known about a particular article, a user profiling system should be able to select the learning material that a user will gain most benefit from viewing. Combining this with a distributed network of learning data greatly expands the pool from which a selection algorithm can choose material, therefore massively increasing the teaching capabilities of the system, but introduces further complexity to the design of the data structures and the network itself.

There are many issues surrounding creating an effective e-learning package, many of which are discussed by Shute and Towle in [4], who note Gagne's 'Nine Events of Instruction' from [5] which is arguably an extremely effective method of evaluating the usefulness of a system. There are arguments currently being put forward that suggests that people do not learn effectively when using an electronic system, and hence a major issue in the development of current-generation e-learning system is overcoming this to ensure the teaching facilitated through the software is absorbed by the end user. The nine-step process given by Gagne tries to quantify how the flow of a person learning should be reflected in an e-learning system, and hence is an extremely good measure of how effective a teaching solution is.

This project is targeted towards the specification and development of a fully distributed e-Learning network utilizing profile-based content selection algorithms in order to prove the link between these two areas of the e-Learning field, and demonstrate the potential of such a system within real-world implementations.

B. Aims

My proposed course of action within this project splits the necessary components down into three areas, each forming its own deliverable item. Firstly, the local server software containing the necessary functionality to hold learning material and perform sorting based on a given user history, which should form the basic objective to prove that this system is a viable approach. All the server functionality should be available remotely, through a defined specification that can be implemented in whatever manner is deemed fit.

Secondly, the networking layer necessary for multiple servers to interact with one another should be defined and implemented, including the adaptation of the 'local server' software to forward requests on to the network, and deal with responses to return the best material on the known network to the client. This will include overcoming the issues of trust, and of peer discovery on the network.

Finally, the client software for holding and parsing user history should be created, including the implementation of a feedback loop between the client and server in order that the content selection algorithms can utilize some form of 'machine learning' in order to better its own results in future searches.

II. RELATED WORK

The concept of e-learning systems has grown in popularity as the adoption of technology into businesses and homes has expanded. For many years it has offered a relatively cost-effective way for users to have an interactive learning experience, although many solutions found within business today are bespoke systems tailored towards the companies' individual needs. Several market leaders in the field have emerged, including some open source alternatives to commercial packages.

One such system is the proprietary VLE (Virtual Learning Environment) and course management system 'Blackboard Learn' [6], which aims to "enrich the education experience" that universities and other educational establishments offer to their students. Being aimed at the education market, Blackboard tailors itself by offering its content split down into individual courses, which can be assigned to users as necessary, and administrated individually by relevant staff members. One common open-source alternative to Blackboard is Moodle[1], which focuses on providing interactive quizzes to users, as well as a high level of syndication to allow data to freely flow in and out of the software as required by the individual implementation. Moodle has a large community developer following due to its high extensibility, as writing new modules to customize the software is extremely easy.

Some of these systems also introduce another element often discussed within the field of e-Learning - social interaction. Li, Lau, Shih and Li discussed this issue in [7], based upon the framework suggested by Qun Jin [8], both seeing the need for users to be able to collaborate and share learning progress with others. Many of the key issues surrounding the prospect of integrating current generation e-Learning into social media is discussed by Chatti, Frosch-Wilke and Jarke, who suggest that “we need federated, intelligent and social search engines that build on user recommendations” [9], summarizing the case for building next generation e-Learning software upon a social backend in order to improve upon current systems. They also suggest that “knowledge pull” (the process of automatically gathering content from sources available on the internet to widen the pool of available learning data) is the optimum solution to get the best results from a social search algorithm.

The concept of creating a distributed network for learning material is not entirely new, however there is currently very little investigation into the field. Sampson et al. [10] suggest a similar approach as part of their “Knowledge on Demand” project[2], whereby the roles of content authors, brokers and providers are separated into individual implementations; however they do not go as far as suggesting a truly distributed nature. They do, however, outline the need for a singular standard in order to ensure the interoperability of different e-Learning systems, and hold this as key to the wide adoption of any future system. Similar themes are also picked up by Westerkamp in [11], as he suggests a web-based e-learning network based around a centralized learning server with many clients. He highlights in particular the advantages of a system being available on the internet, including interactivity with other content sources and user data, giving LearnServe[3] as an example of how this can be implemented.

The ideas of social interaction and distributed networking in an e-Learning system are not new concepts, and have been proved individually on a variety of both small- and large-scale projects, however the combination of these two in order to create a profile-based distributed learning network is a relatively new concept, hence this will form the base of my project.

III. SOLUTION

My solution to the issues described above was to create two elements of a distributed network – the servers that can be interconnected to form a ‘cloud’ network of learning information, and the client implementation that can connect to a server within this network in order to query the data contained within it.

A. Architecture

The components necessary to perform the tasks within a distributed e-Learning network have been split into three distinct sections for implementation; the interaction between these components can be seen in Figure 1. Within this section, I will discuss the design and implementation of each individual component, going into further detail on the data contained within each, and how they interface with each other.

---

[1] Modular Object-Oriented Dynamic Learning Environment – http://moodle.org
[2] http://kod.iti.gr/
[3] http://dbis-group.uni-muenster.de/projects/LearnServe/

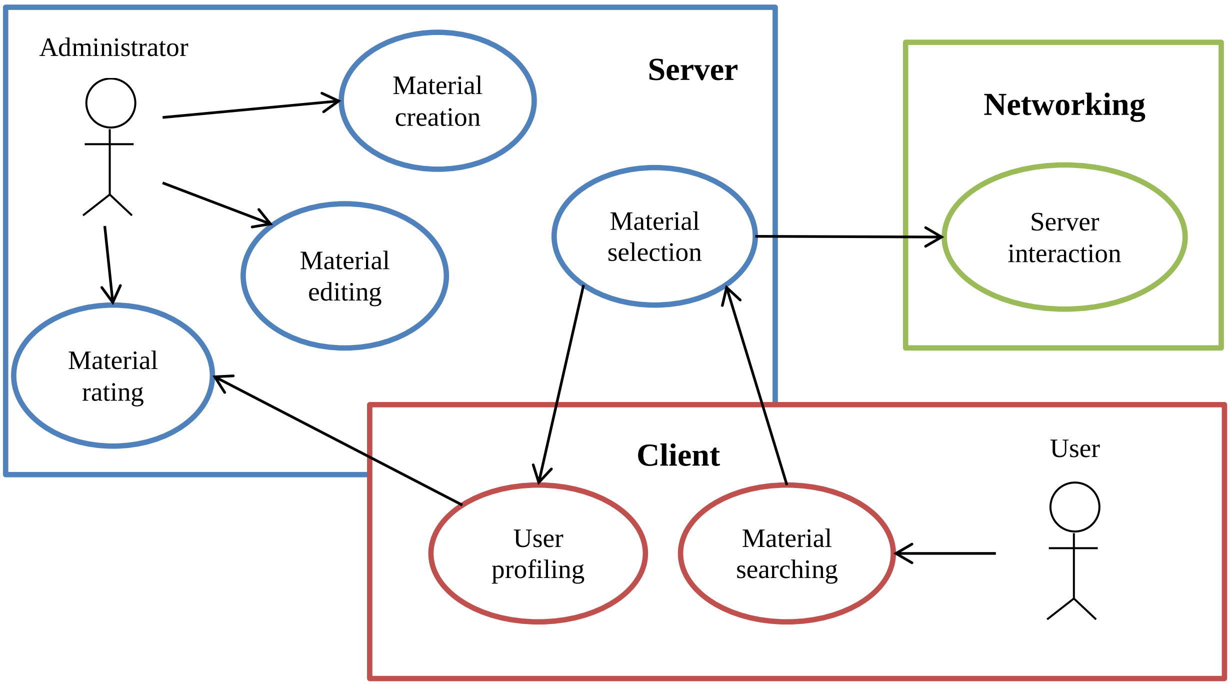

FIGURE 1: SYSTEM ARCHITECTURE OVERVIEW

1. Server

The server implementation has to perform several roles in order to take part in the network, and should be able to perform these tasks completely independently of any other resource on the learning network if necessary.

Primarily, the server is to provide connectivity to any client wishing to interface with it. This involves authenticating the client software to ensure they have permission to access the data on the network, and then responding to any request the client may then make for learning information. This is achieved through the second main feature of the server, the decision algorithm, which should be able to compare a user's history of learning data to the data contained on the local database (and on the network) in order to choose the most relevant

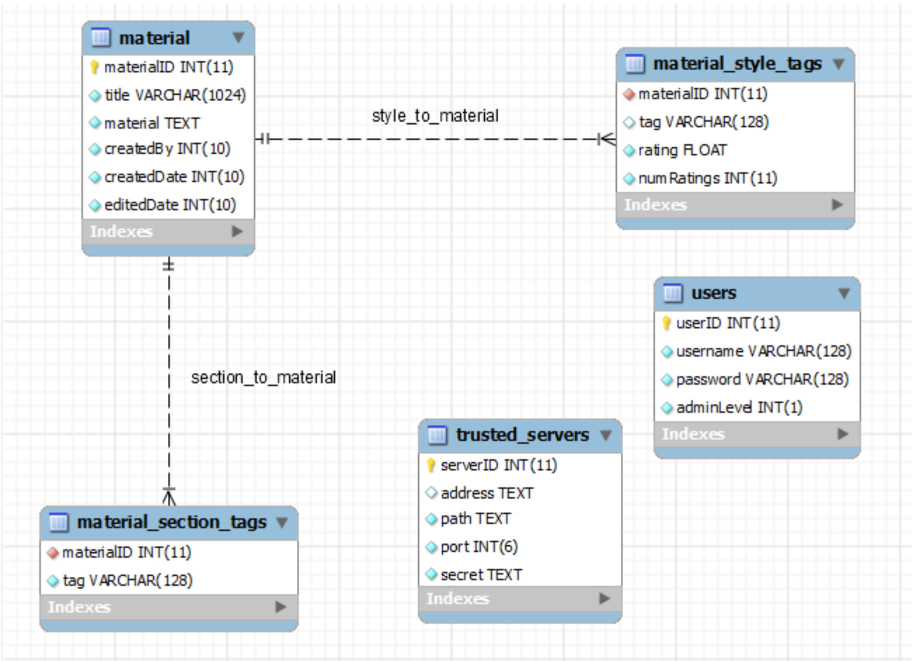

FIGURE 2: ENTITY RELATIONSHIP DIAGRAM OF SERVER DATABASE

piece(s) of information to return to the client (see Material Selection Algorithm). Finally, the server should be able to communicate with other known servers on the network in order to exchange information, in order that the client gets the best material suited to the users taste.

In order to implement the server on a widely accessible network, I chose to write the server software using PHP, which has a wide number of available Remote Procedure Call classes already available (to avoid re-inventing the wheel!), and also provides connectivity to the World Wide Web on a publicly accessible web server. The actual learning material on the server is contained within a MySQL database, which stores information on the tags an article is given, the structure of which is shown in Figure 2.

2. *Client*

The client, at its simplest level, implements the functionality provided by the server, whilst keeping the end user completely detached from the complexity of the computations being performed in the background. This involves implementing the correct remote methods from the server, and storing user history that can be used in the selection and rating algorithms.

In keeping with the server implementation, the client for testing was built using PHP tied to a MySQL database, utilizing the XML-RPC framework

The database (see Figure 3) contains information required to achieve the material selection and rating algorithms (see Material Selection Algorithm and Material Rating Algorithm respectively).

The client interface does not discriminate between different content sources, any material

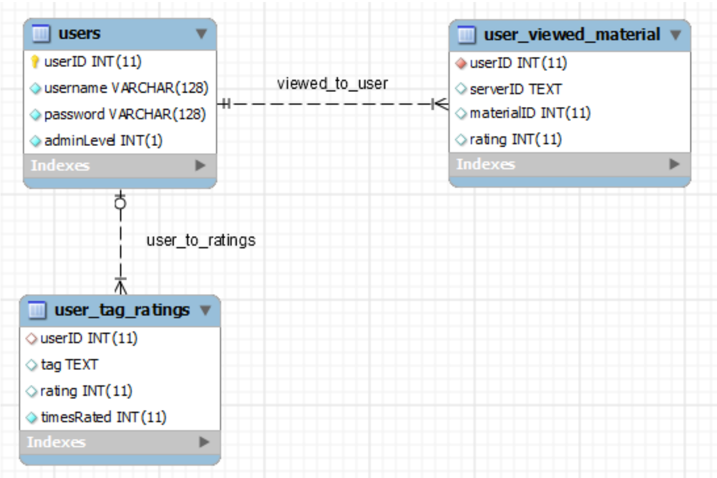

FIGURE 3: ENTITY RELATIONSHIP DIAGRAM OF CLIENT DATABASE

returned from the server is displayed according to the HTML styling information contained within it, and is sorted for display according to its suitability rather than any other factors – as seen in Figure 4, where the client has searched for “internet protocol” and has had 7 results returned, which have been sorted according to the users previous usage data and by relevance to the search query.

3. *Network*

The networking layer tying each instance of each component together must be a widely accessible format in order that the concept of a widely distributed network is realised – using proprietary standards would inevitably lead to confusion. Therefore I have designed and specified a set of Remote Procedure Call methods which achieves the functionality required of the network, and implemented these through the XML-RPC specification [12]. A list of these methods is shown in Figure 5. This ensures that different implementations of this specification can be written in any language on any platform, as long as they respond to the given methods. This is crucial to the adoption of any widely distributed system such as this one, as individual organisations may want to integrate into their own existing infrastructure, rather than being tied to a specific codebase.

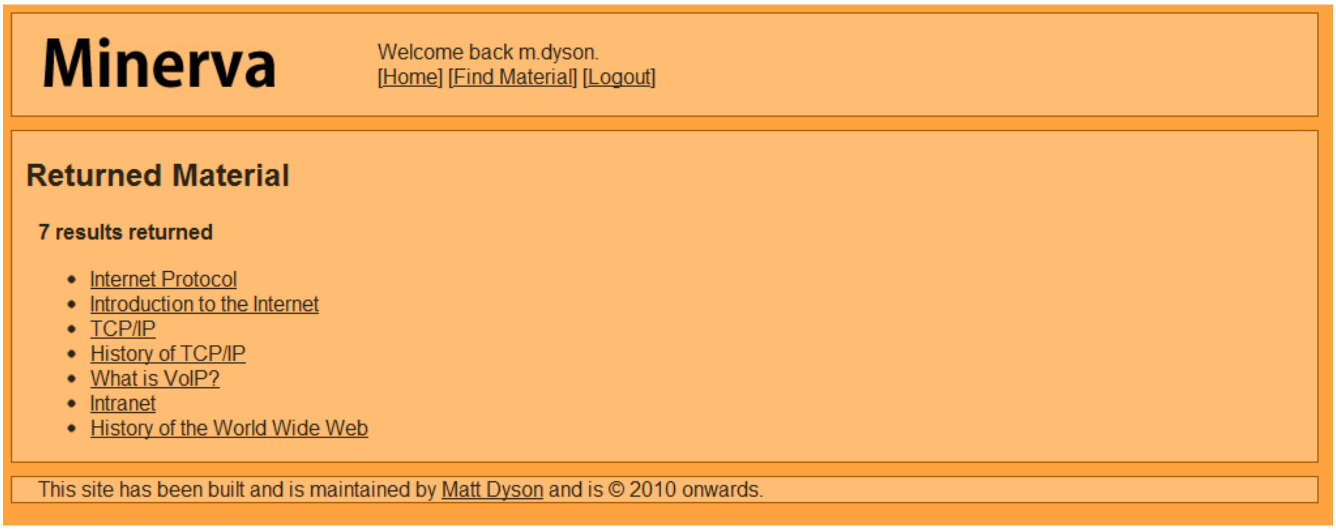

FIGURE 4: MATERIAL RESULTS PAGE

The major problem to overcome within this section of the development was the location and storage of information on peers. As a PHP script is, by design, only executed when an incoming request is received and is able to be processed, and this may happen simultaneously, it is near-impossible to store a running list of active peers, and would be impossible under this architecture to hold a constant connection (which is undesirable due to the processing overhead required). For this reason, a table of known and trusted peers is kept in the server database (see Figure 2), and during network polling each server is contacted in turn (threading is extremely difficult within PHP) using an external pre-communicated secret key, to ensure the two nodes trust each other. This eliminates the possibility of a 'rogue server' being able to access the network, and also allows for customisation of sharing rules on each server – a point which would undoubtedly be in contention in modern times of Intellectual Property law.

Method	Called by	Description
auth.checkAuth	Client	Checks if the connecting client is authorised
client.getMaterial	Client	Return the requested material, checks parameters whether this is a single piece, or a search
client.rateMaterial	Client	Rate (or pass on to the relevant server) the piece of material
tags.getAllStyleTags	Both	Return a list of all known style tags (includes parameter for partial search)
tags.getAllSectionTags	Both	Return a list of all known section tags (includes parameter for partial search)
server.getMaterial	Server	Return the requested material, checks parameters whether this is a single piece, or a search (handled differently as this is coming across the network)
server.rateMaterial	Server	Rate (or pass on to the relevant server) the piece of material (handled differently as this is coming across the network)

FIGURE 5: TABLE OF RPC METHODS

B. *Content Tagging*

A distributed network, by design, does not guarantee communication between any two nodes at any given time, and as such could quite easily become fragmented. In order to eliminate data duplication or redundancy within my proposed system, I needed to make sure that servers would work as efficiently whether or not they were connected to a network, and that repeated disconnection from other nodes would not affect the operation of the internal database. For this reason, it was essential to establish a method of applying information to each individual piece of material on the network without the necessity for a direct connection. The information that needed to be stored is relatively simple – each learning article on a server should contain information on how users have reacted to it in past, so the server can learn from this in order to best serve results in future (see Material Selection Algorithm). In order to keep the client implementation completely independent from the server, it is therefore also necessary to store information on how users react to material within the client. Through the design of the network, it is entirely possible that a client will not be holding a direct connection to the server it is displaying information on, therefore the need for separation of concerns is an important issue.

In order to achieve this separation, it is necessary to come up with a method that can be used to connect a particular aspect of an article to a user's rating of that aspect, which is done within this project by the concept of 'content tagging'. Each article within a server will be assigned 'tags' which relate to their content or style, similar to that of the HTML META 'keywords' element [13]. For instance a video on network packets may be assigned the content tags 'computer-science', 'networking' and 'packets' (describing what the content is teaching), and then the style tags 'visual' and 'video' (describing how the content goes about teaching it). By sharing these tags between servers, their use should propagate and fulfill the needs described above, so tag sharing functions were built into the RPC specification (see Figure 5), and were implemented through an 'auto-fill' form on the material creation pages, as shown in Figure 6, where the tag 'internet' has been pulled from a remote server and is being suggested to the user.

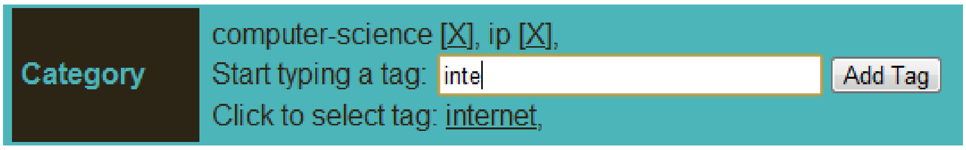

FIGURE 6: TAG AUTO-COMPLETION ON SERVER

C. *Material Selection Algorithm*

The material selection algorithm forms a major section of this project, and is designed to be implemented over a distributed network with the minimum of data being passed between nodes. The algorithm needs to take what information is has on the user, build a profile of what they are likely to respond well to, and then apply this to the search terms given in order to extrapolate the optimum material to return.

In order to achieve this, I designed and implemented an algorithm (see Figure 7) which relies on the history of a user being passed into the server (can be completely anonymous) in

```
function selectMaterial(searchTerms[ ], userHistory[tag -> rating])
   select items in database with content tag in searchTerms[ ]
   for each found article
      match[details] = article details
      match[contentTagsFound] = number of matching tags in database
      for each style tag associated to article
         if tag is in userHistory
            match[suitability] += tag rating from user
      add match[ ]  to matches[ ]

   send out to network, add results to matches[ ]
   sort matches[ ] according to suitability
   return quantity of articles requested
```

FIGURE 7: PSEUDOCODE OF MATERIAL SELECTION ALGORITHM

order to establish the optimal material that the user should respond well. This algorithm iterates through each piece of content within the local database containing content tags matching the search criteria, and examining each style tag associated with the content. If the user has previous given a rating to the tag being examined, the appropriate score from the users history is added to a 'suitability score' metric, which is eventually used to sort the articles by their relevance, taking into account the number of content tags matched, through use of a sorting algorithm. Matches brought in from a network-wide search can also be introduced and run through the local sorting algorithm, rather than accepting the suitability score given by a remote server, which could potentially be artificially raised.

This user history is obviously difficult to establish, and is the responsibility of the client implementation to handle in whichever way is decided best (see Material Rating Algorithm) – however when it is passed into the server it is provided as an array of tags linked to a floating point number between 0 and 1 (1 being a perfect match), for example – {{'video','0.654'}, {'analytical','0.436'}}. By allowing the client to define these relationships, some of the potential for servers 'cheating' the results is eliminated, as obviously the client does not want to send any data that could negatively influence its results.

D. Material Rating Algorithm

In order for the Material Selection Algorithm to perform a basic type of 'machine learning' in order to continually better its own results, a feedback loop is necessary from the client to the server containing the material in question, where information on the users history is shared by the client, and the server shares previous information on the material. The RPC specification for the methods involve each require a floating point number between 0 and 1 being passed (1 being a perfect score for the particular tag or article in question), and by storing the sum of these sends in both the client and server databases, along with the quantity of ratings received, the average rating can easily be identified. The feedback loop is achieved from the client to the server by the passing of the users rating (between 0 and1), and an array of their previous ratings for tags (a list of tags to rating, between 0 and 1), as shown in Figure 8. This algorithm takes each previous tag in the users' history, multiplies this by the rating the user has just given to the article in question, and applies this to the list of tags held against the article. This will lead to reasonably accurate scores being applied to the database, as a user with a low rating of a certain tag, 'visual' for example, applying a low rating to an article implies that the 'visual' element of that article is also low; therefore a low score will be applied to the 'visual' tag, and vice versa.

```
function rateMaterial(rating, userHistory[tag -> rating], article)
   for each tag in userHistory
     calculatedRating = rating * userHistory[tag]
     if tag exists for this article
        add calculatedRating to running total for article
        add 1 to quantity of ratings for article
     else
        insert tag into database with calculatedRating and quantity=1
```

FIGURE 8: PSEUDOCODE OF MATERIAL RATING ALGORITHM

The client effectively performs the reverse of this, by taking a list of all tags associated with the article from the server, multiplying this by the users rating for this particular article, and appending or inserting into the database as necessary. Some experimentation was required during the implementation of this algorithm, as a 'chicken or the egg' scenario is created – we need a user history to populate the learning material with information, and we need information about the article to build a user history. It was found to be most effective to seed the article with some false history on insertion, as any incorrect data would quickly be normalised out by the rating algorithms on both ends. This also highlights the fact that such a system would be much more reliable on a large-scale system, as being able to source information from hundreds or thousands of previous results is much more effective.

E. Testing

In order to carry out testing of the distributed system, it was necessary to set up a network that will simulate all the conditions that could occur. A map of this network is shown in Figure 9. By using this setup with a single client that could be switched between servers as necessary, it can easily be determined whether the servers are gathering the correct

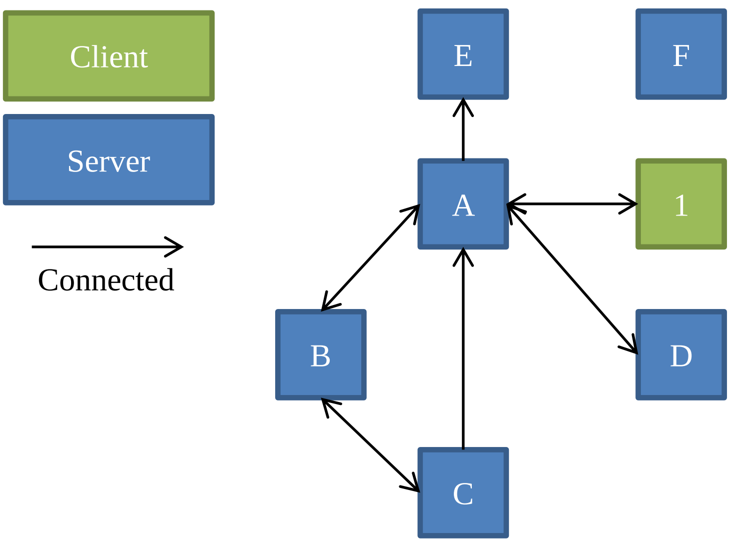

FIGURE 9: MAP OF TESTING ENVIRONMENT

information from appropriate sources, and whether the process of one server 'trusting' another is being performed correctly. Each server was seeded with individually identifiable learning data, and had appropriate tags assigned to it.

In order to perform initial testing of the content selection and rating algorithms, the system was deliberately set up so that a certain search term would return a specific result. For instance, the network would be set up with 5 results for a search, some of which would be a 100% match for the user, some would form a middle ground, and some would be a 0% match. By monitoring the server selection process, I was able to prove the algorithm was selecting the correct content and applying appropriate scores, however whether these scores accurately represented how a user would react to the article is a different matter, discussed within Experiment Results below.

IV. RESULTS

Within this section, I will detail the experimentation I performed on my completed system, and outline the results I have gathered through testing. Unfortunately the nature of this project does not lead towards a quantitative analysis, due to a user's perception of their education being entirely subjective, and therefore extremely hard to judge. Therefore, I have attempted to assess the effectiveness of my solution through a series of questionnaires and experiments.

A. Experiment Context

For the user testing, I performed a series of experiments on a one-to-one basis with 20 people, asking each one to use a pre-existing user account to research a given topic (requiring use of at least 15 different articles on the network), before asking a series of questions relating to their experience with the system. The users asked to take part in these experiments covered a variety of different age brackets (participants were all in the range 18-65), and all had at least some basic experience of computer systems, although only 7 participants had ever used any e-Learning software as part of their educational or professional development.

Each user was given a blank user account with no prior history stored, so the system was attempting to learn the users' style from their reactions alone, meaning that the adaptation to users' styles should have been at its most obvious. The learning network was set up in a completely connected graph containing 5 nodes, all with unique data that deliberately contrasted that on other nodes – for instance one server contained primarily learning material suited to a visual learner, whilst another contained only kinesthetic material, which would mean that any 'machine learning' observed was utilizing the network to its full capacity.

The test subjects were each asked 4 questions, and given a choice of five answers (Strongly Disagree/Disagree/Neither Agree or Disagree/Agree/Strongly Agree). The questions asked were:

1. I found the system easy to use
2. I felt the system was adapting to my learning style and the feedback I gave
3. I was aware of where the learning material was coming from
4. I am comfortable with what the system is learning about me personally

Half of the test candidates were also then asked to use the administration panel of one server in order to add a short piece of learning material in a subject area they felt comfortable writing about, and were asked the following questions:

5. I found the administration panel easy to use
6. I found the tags suggested to me useful
7. I understood what the panel was asking me to do

B. Experiment Results

The most important factor to establish through these tests was whether the user perceived the system to be learning from their feedback. The results from Question 2 are shown in Figure 10, and show a strong trend to users agreeing that the system was learning from their feedback, and was continually suggesting better learning material to them. It was interesting to note that as the series of experiments progressed, subjects' responses started trending towards the 'Agree' end of the scale, having started low. This suggests that the data held by the servers was also being approved on as it gathered information from each candidate, and therefore the feedback loop created was working in both directions – users coming to the system were benefiting from feedback given previously. This factor could have been eliminated by resetting the entire network between tests, however this does provide an interesting insight into how the system performs, as the process of 'machine learning' is clearly taking place, and is a better way of tagging and rating material than the initial seeding done to the network artificially.

The results of Question 3 (enquiring about how aware the user was of the source of the information they were being presented with) were overwhelmingly in the negative end of the scale, with 80% of candidates stating that they had no idea where the material had come from. Several enquired whether the information was being pulled 'live' from the internet, suggesting a similar system to the 'Knowledge Pull' suggested by Chatti, Jarke and Frosh-Wilke in [9] and discussed within Related Work.

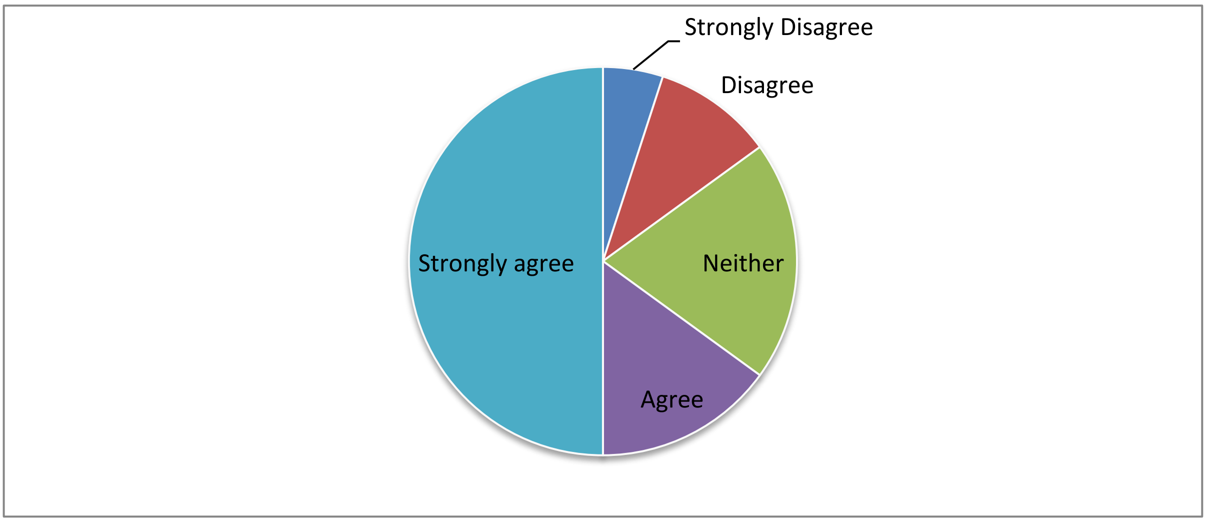

FIGURE 10: RESULTS OF QUESTION 2

One other element that was interesting to discover with this learning system was the participants' perception of how the software was learning about their patterns and using this to suggest better material in future. Once this process had been explained to them, they were asked Question 4 (I am comfortable with what the system is learning about me personally), and the response was between indifference and disagreement – many users expressed concern that some very personal information was being stored and shared around a network, although they agreed that the system was beneficial to their learning.

The questions asked of the candidates about the administration panel gave some interesting results regarding the structure of the panel. For instance, 50% of the candidates tested with the panel agreed that it was simple and easy to use, however only 2 candidates said that they found the tag suggestion feature useful (see Figure 11), the rest failing to see

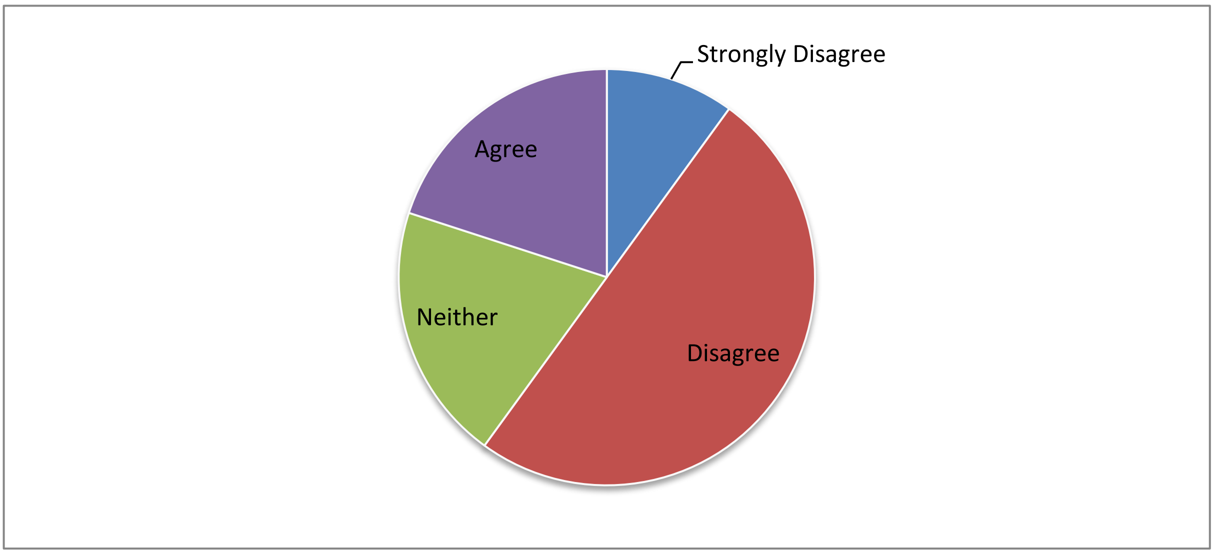

FIGURE 11: RESULTS OF QUESTION 6

the point of the suggestions. When asked about this, the candidates that didn't agree felt that the tags suggested to them were in some way appropriate, but when they failed to find one that matched their perception of the data presented, they were unwilling to add a new tag for fear of duplicating a pre-existing one. The results of the administration panel usability question (Question 5) also suggested that the GUI could be improved. Whilst all candidates were able to quickly find the section they required, and were able to achieve all tasks given to them, they felt that there was not enough explanation given of how the tags and other inputs would be used, a similar concern to that held by some members with regards to the client panel.

The results gathered during this series of experiments demonstrate that the concept of profile-based e-learning through this method of content tagging is effective in engaging users into the system, and delivers results to queries that are clearly tailored towards the users learning styles. Whilst there were some issues surrounding the usability of the graphical user interfaces, and confusion about the flow of information in the administration panel, the core aims of this project have been demonstrated, as the usability lies outside of the scope of this demonstration implementation. It is very encouraging to see that users of the system were not at all aware of the source of the information they were being presented with, suggesting that this method of distributing e-learning content over a network does not adversely affect the users experience with the system – a factor which is essential to the success of this project.

However, the results of the survey do suggest that further work is necessary in order to perfect the user-facing part of the content tagging system, suggesting that the implementation in its current form would not be suitable for use on a wide scale.

C. Analysis of algorithms

Whilst the algorithms documented within this report have foundations within mathematical probability, it is extremely difficult to prove their efficiency due to the subjective nature of their purpose. For this reason, I ran a series of test cases and measured the response of the system, plotting the results graphically, from which it should be easy to identify if the algorithms are producing the desired results.

The graph shown in Figure 12 plots the user score and article score for a single tag over 24 iterations of a rating being applied to the system. These are calculated by taking an average of the previous ratings available for a tag (for either an article or a user, depending on context) and therefore should accurately reflect the relationship between the entity in question, and the tag.

The article score was initially set to 0 (a single rating of 0%), and the user score was set

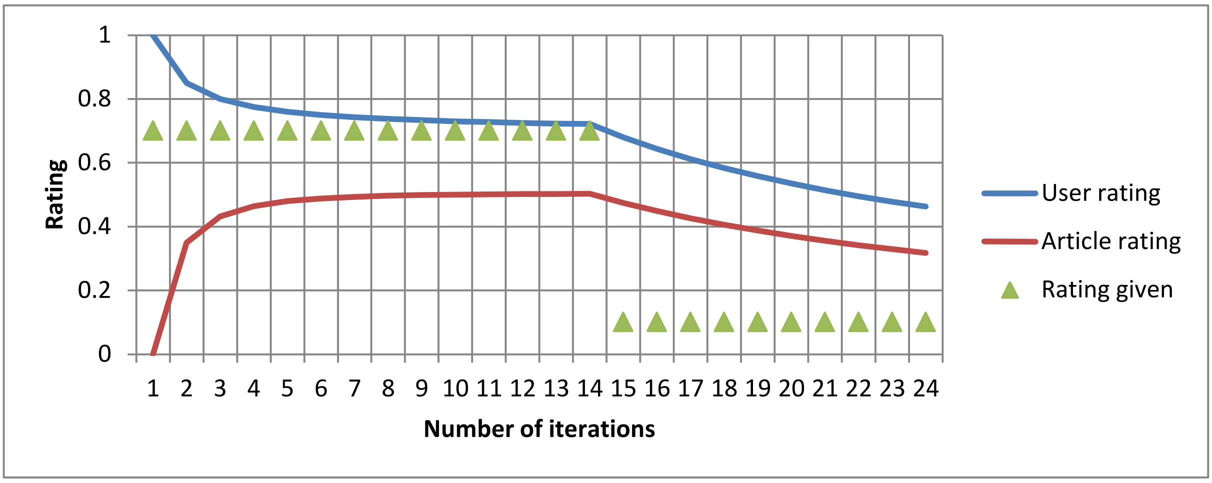

FIGURE 12: GRAPH OF RATINGS OVER TIME

to 1 (a single rating of 100%), and a series of ratings were then passed into the system as shown. For the first 14 iterations, a rating of 70% was given, and the user score can clearly be seen to trend downwards and level off at just over 0.7 as expected, whilst the article score is increasing. The rate at which the article score increases drops off as the user score falls, as expected – if the user applying the rating does not score the given tag highly, the article rating will not increase as much. After 14 iterations, the rating applied was dropped to 10%, and both metrics begin to drop off as expected.

These results demonstrate that the two algorithms (user ratings and content ratings) generating metrics on user interaction with the system are effective in representing opinions on the tags over time, and therefore can be used within the material selection algorithm to present the best information to the end user. By use of this cyclic process of ratings (user ratings directly affecting the content rating, and vice versa), a form of machine learning is created, whereby the system will continually evolve and adapt its results to ensure that the best content is always displayed. Figure 12 clearly demonstrates this, as the rating applied to the article is dropped, the two metrics (user and article rating) immediately begin to decrease as well.

D. Performance

When judging the performance of this implementation, there are two key metrics that can easily be analysed – the speed at which the algorithms return results, and the time taken to pass data over the network. As discussed above, the performance of the network is a factor outside the scope of this project, as the information is being passed over the internet and internal networks. The learning data and other information being passed across the network are kept to a minimum to ensure that transport times are kept to a minimum.

The performance of the algorithms is measurable, and in this case the run-time of the material selection algorithm is most important. The algorithm designed and implemented as part of this project applies a brute-force approach to searching (irrespective of the approach the DBMS[4] takes to record-finding), and hence when applied to a large quantity of articles and tags can lead to a substantial run-time. However, this eventuality should never occur, due to the processing being spread out over a large network – if the content is distributed evenly over a large number of server nodes, the search time on each will be negligible, and combined with an efficient sorting algorithm on the local server this will lead to extremely fast response times.

V. Evaluation

As demonstrated within the previous sections, my implementation of a distributed profile-based e-learning network has proved the viability of this as a solution to the problem of providing the best learning material to a user, whilst ensuring that the information presented and held within the system can be easily updated, and adequate privacy controls over the information are held. Within this section, I will discuss the strengths and weaknesses of the project, and will suggest some possible improvements.

A. Information Storage

The implementation of a distributed system carries with it a high level of complexity, such as the additional overhead required for inter-node communications, the extra design required to ensure a stable and reliable system, and the provisions necessary to eliminate the risk of data corruption (duplication, redundancy etc.). To eliminate this, I chose to treat each server as a completely separate entity, and only permit data to be transferred between nodes at the point when it is required. Whilst this approach ensured that no issues of data corruption could occur, it did add substantial overheads within both the networking and algorithm complexity (see Results: Performance) which when implemented over a large network could potentially lead to unacceptable wait times on requests, especially if synchronous requests are not possible. Due to the vast amounts of data involved in setting up a test machine, it was impractical to utilize more than 6 server nodes at the same time, leaving this potential pitfall untested.

The utilization of a 'tag-based' system for storing information and ratings for users and articles also overcame the primary issues of operating over a distributed network; however the process of crowdsourcing[5] information on tags introduces many more problems. A large number of these are addressed by Waltinger, Mehler and Heyer in [14], who suggest approaches such as 'tag-clouds' populated by creating links between separate tags, which can be achieved through implementation of some basic graph theory. The results gathered in my user survey and discussed within Experiment Results clearly show that my process of tag suggestion within the administration panels was effective in spreading pre-existing tags across the network, however the creation of these in the first instance was not an obvious process, something that could perhaps be improved by utilizing the ideas put forward within [14].

B. Rating system

As discussed above, the distributed architecture used for this project introduces a number of problems related to information sharing. I chose to overcome these issues within the content rating system by making use of the tags, and keeping the responsibility of ratings within the scope of the client software, where it can be specifically linked to a user, and passed on to a

[4] DataBase Management System – The software that controls use of the database
[5] Outsourcing a task to a large undefined group of people

server as necessary. Whilst this approach eliminates the risk of data redundancy across the network, it does massively increase the overhead network requirements for passing rating data between nodes, and also increases the storage need within the client system, which has to store a full list of tags available on the network corresponding to scores for each user on the system, although this could potentially be normalised into a more space-efficient format at the expense of processing power.

The implementation in its current form does, however, store enough information on a user and article to lead to efficient algorithms for further content rating, and more importantly – content selection. The data shown within Results: Analysis of algorithms clearly proves that the algorithms at work will select the best information available over the distributed network that fits the stored user profile, although the computational power necessary to achieve this does grow exponentially with the size of the network, the amount of content available, and the quantity of known tags on the entire system. Without further testing on a large-scale system involving many thousands of user-ratings per article, it cannot be determined how this algorithm will perform under load, however it is entirely feasible that the concepts defined within this project could be implemented in a less computationally complex manner.

C. Network

One substantial issue that I have only addressed on the most basic level within this project is the issue of trust between nodes on the distributed network. Although the implementation of the Remote Procedure Call structure includes a 'handshake' between servers, as well as simple authentication between servers and clients, the actual data passed between nodes is always presumed to be correct and is only subject to the most basic validation. Because of this, it would be relatively easy for a server node within the network to begin to 'fix' the results that it is giving out to clients, ensuring that its results would always come out on top for any search. This could be overcome by shifting the process of content rating over to the local server (the one directly connected to a client), allowing the network servers to choose their own best results, and then the local server using its own internal algorithms (a slightly different implementation of the material selection algorithm) to order the content returned irrespective of its source. This process would greatly increase the processing power required across the network, as effectively the same process is being performed twice for each result to a query – the local server taking the brunt of this processing, however this could help to eliminate the risk of nodes within the network returning faked statistics for its content.

One consideration during the design of this project was the architecture of the network – there are a number of advantages to using a truly peer-to-peer style instead of the direct connection that was implemented. Primarily, there would be less processing required to hold connections between individual server nodes, and a 'peer rating' could be implemented to ensure that only nodes returning valid and relevant results are kept within the network, whereby any rogue node would be disregarded by its peers. However, due to the nature of the information being shared, and the use-cases discussed within Introduction, this was deemed to be inappropriate within the scope of this project.

With this model of network architecture it would be relatively easy to perform asynchronous requests to further speed up the performance of the network, and also reduce any potential bottlenecks. The issue of locking within a multi-threaded environment is also not a risk with this design, as the DBMS only needs to lock certain records for a single update, rather than having to perform operations on multiple tables simultaneously.

Unfortunately the testing performed on the distributed network does not lead to any conclusive evidence due to the difficult nature of setting up a test network. In order to fully examine the performance and draw conclusions from the data, a large network (10+ server nodes with individual clients) each containing large amounts of learning data (multiple pieces on the same topic, each utilizing different learning styles) would be necessary, and then thousands of accurate user ratings would need to be fed into the network in order for the 'machine learning' element of the network to truly be demonstrated.

D. Interfaces

The structuring of this project into the three separate deliverables (server, network and client) proved extremely beneficial in the development stages, and this has been obvious right through to the testing. By separating the concerns of the network, each individual section can be evaluated separately, distinguishing exactly where the sections can be improved, and also gives some scope into demonstrating how the system could be used on a larger scale.

As the scope of this project called for emphasis on the back-end system to prove the concept of an e-learning network, the front-end systems were simple interfaces providing access to the functionality provided. Although some effort was put in to making these interfaces easy to use, in fitting with the common principles of interface design (such as those put forward by Shneiderman and Plaisant in [15]), the overall performance of the user interfaces did not encourage the users to interact with the system as well as hoped, leading to some negative feedback. This area could use some significant improvement to fully demonstrate the capabilities of the concept.

E. Data Privacy

One issue that has not been specifically addressed during the design phase of this project, and could prove to be a factor hindering any potential large-scale adoption of a distributed e-learning system is that of data privacy. Results gathered during the testing of the system demonstrated a significant spread of opinion over whether users were happy with the system storing information about their learning style, however since this data can be transmitted anonymously over the network, this is a minor concern. The other major factor when considering a system that shares content in this manner is that of protecting the information to ensure it cannot spread to nodes that aren't permitted to see it. Originally this project was going to take the direction of a truly peer-to-peer system, whereby each node was treated with equal rights to view information within the network, however this was deemed to be inappropriate when the content being distributed is potentially valuable intellectual property, therefore some kind of security was deemed necessary. The approach of enforcing server authentication and a direct connection as opposed to an open network could mean that the adoption of such a network would not be wide scale, however the primary target of this project would be towards the business market for providing content, and in this situation some control over the sharing of data does need to be factored in to any design.

VI. CONCLUSIONS

Within this project, I have discussed, designed and implemented a system whereby profile-based e-learning can be used over a distributed network, utilizing machine learning to ensure that the end user gets the best learning material available at that moment, with a view to forging a link between these two areas of the e-learning field. This network has then undergone testing and evaluation to determine whether the amalgamation of these two concepts does benefit the end user, and if the cost of this implementation is outweighed by the potential improvements made.

The results of the user study conducted do indicate that users of the system were finding the information presented to them beneficial, and that the network was working to continually adapt the results it was returning to the user. As discussed, the test network was small in comparison to the proposed implementation of such a system, and therefore cannot be used as concrete proof of concept; however it does demonstrate that the theory behind the profile-based e-learning distributed system is viable. The merits of implementing a distributed network for content, and also utilizing profile-based methods for content selection combined with machine learning to continually improve results are clearly visible, and this project has gone some way to demonstrating the capabilities of such a system, and the architecture implemented during testing does lend itself to implementation within a number of different fields.

For this reason, I would judge the project to have achieved its aims and therefore be successful. The concept of a distributed profile-based e-learning network is clearly achievable, and has clear benefits over the current-generation systems that adopt a single server or entirely client-based architecture for content distribution. By distributing the content over a large-scale network, problems involving server overloading and out-dated information can be eliminated, and through use of the profile-based system the best content will always be selected for the users based upon their individual learning styles – an outcome that is clearly beneficial to any provider of e-learning content. Use of machine learning within the content selection algorithms in this project have also demonstrated how the user history gathered from the entire network can be used to build up accurate profiles of user preferences and content style, a process which eliminates many of the problems found with e-learning systems. However, there are many different ways this concept can be expanded further, and improved upon to create an even more effective e-learning network.

As mentioned previously, the results achieved within this project indicate that this content distribution network in its current form lends itself towards a business application, whereby a company can offer Computer Based Training on a particular product to its customers or employees, and this system offers the ability for further servers to be added by 3rd parties. By expanding this network of available content, coupled with the distributed content selection algorithms, the most appropriate material for the end user will always be selected, a clear advantage in any implementation.

The content provided over the system during the development and testing phases was mostly static (formatted text and pictures), and was extended to include capability to embed videos, however within any learning environment it is arguably better practice to let the user take part in some kind of interactive experience. A computer platform gives a lot of scope this type of learning, and a large number of companies (such as MyMaths Ltd[6] and others) have approached this market and sell access to interactive online activities to schools. Businesses and other organisations are also approaching interactive experiences as a method for training (such as the strategies proposed in [16]), and pre-existing companies such as The LEGO Group are introducing product lines aimed at so called ‘serious play’ [17], suggesting the need for this in any next-generation e-learning system. This could easily be implemented within the distributed network created for this project, and results from these games could even be utilized as part of the feedback process – a user who achieves higher scores than others could be perceived to be learning more.

This also could be extended into the field of social interaction, a field which has been suggested in combination to e-learning by Anderson in [18]. By allowing users to interact with their friends whilst learning (much the same as a school classroom), there is the potential to vastly increase engagement with the system, enriching the learning experience for the user.

[6] http://www.mymaths.co.uk

One other element mentioned within this project has been the concept of 'knowledge pull', suggested in [9], being the idea of an e-learning system being able to pull information from the wider internet into its own searchable database in order to keep its content completely up-to-date and relevant. This is obviously an extremely difficult concept, as essentially a partially intelligent system is being created, which has to gather information and determine which categories its content falls in to, as well as how well the information can be trusted. Implementing this into the distributed system within this project would also introduce more problems, such as determining which server holds which piece of gathered material (given that in the architecture used here, the network may become fragmented, so any two nodes may not be able to read from each other), and also how to 'tag' the gathered content with appropriate style and content information.

Overall, this project has demonstrated that the fields of profile-based content selection algorithms and distributed systems are both compatible within the field of e-learning. Whilst the implementation within this project has been basic, the concept has been proved and can be used to significant results, albeit not fully proved due to the complexities of the networks required for full testing and evaluation. The introduction of new Web 2.0 technologies into the field also could provide significant advances that would lead to better user engagement with the systems, and hopefully will grow closer to achieving the goal of providing a completely personalised interactive e-learning experience to end users, benefiting both themselves and the authorities imparting the learning data.

VII. References

[1] European Distance and E-Learning Network. [Online]. http://www.eden-online.org/eden.php

[2] Wendy L Bedwell and Eduardo Salas, "Computer-based training: capitalizing on lessons learned," *International Journal of Training and Development*, vol. 14, no. 3, pp. 239-249, 2010.

[3] Richard M Felder and Linda K. Silverman, "Learning and Teaching Styles," *Engineering Education*, vol. 78, no. 7, pp. 674-681, 1988.

[4] Valerie Shute and Brendon Towle, "Adaptive E-Learning," *Educational Psychologist*, vol. 38, no. 2, pp. 105-114, 2003.

[5] Robert Gagne, *The Conditions of Learning, 4th Edition*. New York: Holt, Reinhard & Wiston, 1985.

[6] Blackboard Inc. Blackboard Learn Platform. [Online]. http://www.blackboard.com/Teaching-Learning/Learn-Platform.aspx

[7] Qing Li, Rynson Lau, Timothy Shih, Frederick Li, "Technology Supports for Distributed and Collaborative Learning over the Internet," *Transactions on Internet Technology*, vol. 8, no. 2, pp. 1-24, 2008.

[8] Qun Jin, "Design of a virtual community based interactive learning environment," *Information Sciences*, vol. 140, no. 1-2, pp. 171-191, 2002.

[9] M.A., Jarke, M. and Frosch-Wilke, D Chatti, "The future of e-learning: a shift to knowledge networking and social software," *International Journal of Knowledge and Learning*, vol. 3, no. 4, pp. 404-420, 2007.

[10] Charalampos Karagiannidis & Fabrizio Cardinali Demetrios Sampson, "An Architecture for Web-based e-Learning Promoting Re-usable Adaptive Educational e-Content," *Educational Technology & Society*, vol. 5, no. 4, pp. 27-37, 2002.

[11] Peter Westerkamp, "E-Learning as a Web Service," *Tagungsband zum*, vol. 16, pp. 113-117, 2003.

[12] Dave Winer. XML-RPC.com. [Online]. http://www.xmlrpc.com/spec

[13] World Wide Web Consortium. The global structure of an HTML document. [Online]. http://www.w3.org/TR/html40/struct/global.html#h-7.4.4.2

[14] Ulli Waltinger, Alexander Mehler, and Gerhard Heyer, "Towards Automatic Content Tagging," in *4th Int. Conf. on Web Information Systems and Technologies*, 2008, pp. 4-7.

[15] Ben Shneiderman and Catherine Plaisant, *Designing the user interface: strategies for effective human-computer interaction*.: Addison-Wesley, 2009.

[16] Robert S Becker. (2011) Interactive Games for Business Training. [Online]. http://it.coe.uga.edu/itforum/paper123/Interactive_Games_for_Business_Training.pdf

[17] The LEGO Group. Lego Serious Play. [Online]. http://www.seriousplay.com/

[18] T Anderson, "Distance learning -- Social software's killer ap?," in *Proceedings of the Open & Distance Learning Association of Australia*, vol. 31, 2005, p. 2006.

VIII. Table of Figures