Setting up an ASP.NET Development Environment for a Classroom Lab on Campus

Xiaosong Li **Henry Ren He**

School of Computing and
Information Technology
Unitec Institute of Technology
Auckland, NZ
xli@unitec.ac.nz

Last year, we introduced ASP.NET and Visual Studio .NET environment in our classroom lab. A Web server was set up for the Intranet on campus. It is configured for the client computers in the classroom lab. Visual Studio .NET is installed on the client computers. We have encountered two major problems related to the environment setting up. One is the permission problem. Another is the remote debugging problem. We have worked out a complete solution for the first problem, which has been proved a successful solution by a whole semester's teaching and learning. To investigate the solution for the remote debugging problem, a testing client-server environment is set up to test all the possible setting up options, which includes account options and workgroup options. We have found solutions for the second problem. In this paper we report and discuss our experiences, efforts, approach and our experiment results in solving these problems. We also report and discuss the impact of the environment settings to the students. We would like to share these with interested staffs in other colleges and universities so that they can learn from our experiences and not repeat the mistakes that we made.

Keywords

Classroom lab, ASP.NET, Visual Studio .NET, permission problem, network, debug problem, authentication method, client, server.

1. INTRODUCTION

Web Application Development (WAD) paper is a third year undergraduate course. It was originally offered using ASP 2.0. To meet the demands from both industry and students, we have changed the course content to cover ASP.NET and Visual Studio .NET (VS.NET). This change commenced with the first semester of 2003.

A teaching web server, which is a Windows 2000 advanced server with IIS, was set up for the classroom lab exercise on the campus. It is configured for all the client computers in the classroom lab. VS.NET is installed on all the client computers. VS.NET provides two ways to access the Web server: FrontPage server extensions and File share. The FrontPage server extension is used on our teaching server.

Since VS.NET is installed in our campus network environment, there are issues related to integrating VS.NET with the network that need to be addressed to make VS.NET work properly. In the first semester of 2003, we have encountered two major problems with the above ASP.NET development environment. The first one is the permission problem. We often get the following error message when we try to create a new Web project or open an existing Web project from VS.NET.

```
"The user name or password you en-
tered is incorrect, or you do not
have authorization to permit this
operation".
```

The fact is that our user name and password are definitely right. Another problem is the remote debugging problem. When we try to use menu Debug|Start to debug an ASP.NET file on the remote server within VS.NET, we always get one of the following two error messages:

```
"Error while trying to run project:
Unable to start debugging on the web
server. Unable to map the debug start
page URL to a machine name."
"Error while trying to run project:
Unable to start debugging on the web
server. Access is denied. Verify that
you are an administrator or a member
of the 'Debugger Users' group on the
machine you are trying to debug."
```

The fact is that we have added all the users to the 'Debugger Users' group. We even tried our administrator user. The result is the same. We believe our experience is common. It is very possible to

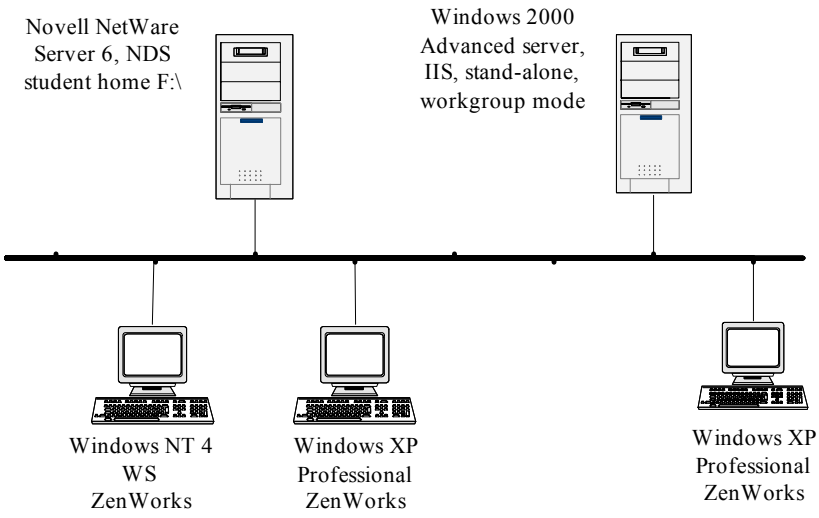

Figure 1. Basic Network Architecture

encounter problems when adapting commercial software to tertiary education needs. We expect our solutions are helpful to the people who are having similar problems. While we solve our particular problem, we also expect to establish a systematic approach on how to solve and avoid possible problems. The approach should be useful for adapting commercial software in the future.

In this paper we will report our experiences in solving above two problems. We first describe the network architecture in our teaching environment. We then describe and discuss our solution to the first problem. After that, we will describe and discuss our experiments and solutions for the second problem. Then the impact of the environment settings to the students is discussed. Those are followed by conclusions and future work.

2. THE NETWORK ARCHITECTURE

Like many other polytechnics in New Zealand, we are using Novell network operating system as well. The network architecture is very important for the settings of Visual Studio. The following diagram shows the conceptual network environment.

Windows 2000 Advanced server with IIS and FrontPage server extension installed is the web server for Visual Studio projects. Novell NetWare server 6 with NDS (Novell Directory Services) is the main authentication server and file server. To secure the network, Novell Policy Packages are implemented. One of the policies is "Dynamic Local User" policy. All workstations' operating systems are Windows NT workstation or Windows XP professional with Novell ZenWorks installed. All workstations are configured in workgroup mode. The workgroup name is the lab name.

When a student logs in the network from a workstation, system will check the student's name and password in NDS. If the student is an authenticated user, NetWare will create a temporary local user and its profile on the local computer based on the "Dynamic Local User" policy. For Windows NT workstation, the profile is created in C:\WINNT\Profiles*username* directory. For Windows XP the profile is created in "C:\Documents and Settings*username*" directory. For the security reason, this local user only has limited rights to access the C: drive. It only has "write" permission on c:\temp directory. When the student logs out of the network, the temporary local user and its profile will be deleted.

Each student has a home directory on the file server, which is mapped to F: drive. A student has full rights to his/her home directory.

3. THE PERMISSION PROBLEM

This problem is important for WAD teaching and learning. Of course we can avoid this problem by using the VS.NET as a text editor (or using any other text editor) and using FTP client to communicate with the Web server. In this way, we still can develop an ASP.NET Web application. However, the

students will miss the opportunity to learn all the powerful features of VS.NET and to get benefit from them. So this problem requires a solution imperatively.

When we run VS.NET on a client computer, VS.NET creates two directories on the local computer: one for solutions, the other for project cache. Regularly cleaning the cache directory partially solved the problem. Kiely (2003) confirmed and explained this approach. A cache directory is mainly created to support offline development. VS.NET also compiles the files to the cache directory for a project and then it uses the server extensions to copy them to the server, which causes garbage to accumulate slowly. Sometimes VS.NET and the cache directory get out of sync. Cleaning the cache directory allows a fresh new start. However this approach does not always work. We have to look for a complete solution.

We tried to analyse the reasons for this problem. The default location for the solution directory and the cache directory is on C: drive. Each student has her/his own cache directory. However all the students will share the same solution directory. This might cause a name confliction. For example, student A creates a new solution with the same name as an existing solution created by student B, student A's solution might override student B's solution. As a result, when student B tries to open the solution later, the error message described in the first section of this paper might be displayed, as student B is not the owner of the solution anymore. The fact that a student generally does not have full write permission on C: drive might cause the permission problem as well.

A solution to this is to change the location of these directories to F: drive, which is the student's home directory. A student has "Full Control" rights on his home directory. This can be done from VS.NET.

```
Change the default solution directory
to the new location: F:\Visual Studio
Projects by using the tools menu:
Tools->Options->Environment->Projects
and Solutions (Visual Studio projects
location)
Change the default cache directory to
the new location: F:\VSWebCache by
using the tools menu:
Tools->Options->Projects->Web Set-
tings (Location of Web project cache)
```

The above solution was implemented and tested on the Windows NT client computers in our classroom lab initially. They worked fine. Later on, all the Windows NT client computers were upgraded to Windows XP professional. At the beginning of the second semester of 2003, the above solution was implemented and tested on the Windows XP professional client computers in our classroom lab. They worked fine as well. For the whole second semester of 2003, we didn't get a single case of the permission problem.

Another possible solution is to work offline. Let the server administrator create a project for each student. Let the students open the project from the cache directory directly, edit and compile the project offline, and then copy the compiled code to the server by using FTP client. This approach has been tested. It worked fine. However, this approach has some limitations. First of all, it avoids the problem instead of solves the problem. Secondly it limits the number of projects a student can use. Thirdly it is not as convenient as working online.

4. THE REMOTE DEBUGGING PROBLEM

4.1 Analysis

The debugging problem needs to be solved as well. Otherwise, the students can't make use the powerful online debug feature of VS.NET. We didn't get the same problem on a local Web server. We didn't get the same problem for a Windows application either. So this is a remote debugging problem. According to (Goodyear et al 2001, p116), this should be easily achieved within ASP.NET and VS.NET environment. We have put in a lot of effort in solving the problem.

According to (Microsoft 2003a, Microsoft 2003b, Microsoft 2003c), to make remote debugging work properly, the following conditions must be satisfied.

1. The user name must be in the Debugger Users group on the Web server.
2. Both local and remote machines must be on a domain setup or, if they are on a workgroup setup, both machines must be running a Windows NT-based operating system.

Table 1. The experiment environment

Client	Server	Workgroup	User Group
Windows XP Professional	Windows 2000 Advanced Server	same workgroup	Debugger users/ Users Or Administrators
Windows XP Professional	Windows 2000 Advanced Server	Different workgroup	Debugger users/ Users Or Administrators

3. Install debugging components on the remote server.
4. If the process running as SYSTEM, you need administrator privileges to debug it. To debug without Administrator privileges, you need to add the actual user name and password to machine.config file on the server.

All the users have been added to the debugging group on the server, the condition 1 is satisfied. Our client computers are Windows XP professional on a workgroup setup and our server is Windows 2000 server configured as stand-alone server in workgroup mode. So the condition 2 should be OK. However, the web server belongs to a different workgroup (technical support group) from the client computers in the classroom lab. This is why file share mode cannot be used in this client-server environment. Further research is required to determine whether this will affect the remote debugging. We have installed all the required components on our Web server, so the condition 3 should be OK. We have checked the machine.config file on our Web server and found that, inside the processModel section, userName=”machine” and password=”AutoGenerate”. This means that the process is running as neither SYSTEM nor User Account. This setting is OK for a local Web server. However, it could be a problem for remote debugging. Currently both of the administrator users and the student users are getting error messages when they attempt remote debugging.

4.2 Approach

From the above, several uncertain factors need to be determined for remote debugging.

1. Whether the web server and the client computers should be in the same workgroup?
2. Whether an administrator user can debug remotely?
3. Whether adding a user without administrator privileges to machine.config file will enable the user debug remotely?

To determine the above factors, experimenting Web server configuration is necessary. However, changing our teaching server configuration will affect our current student accounts. Thus, affect our current teaching and learning. Therefore an experiment environment is needed. Our approach consists of the following steps:

1. Establish an experiment environment and verify that it simulates our teaching environment.
2. Test different user role and authentication method combinations for remote debugging.
3. Create an account on our teaching server, apply those successful settings to this account and test remote debugging.
4. Once step 3 is successful, apply the same settings to all the student accounts and test remote debugging in a semester break.
5. Once step 4 is successful, apply the same settings to all the student accounts in the following new semester.

4.3 Experiment Environment

Table 1 describes the experiment network environment. We use workgroup mode with one Windows 2000 Advanced server as web server. We use one Windows XP Professional workstation as client. This simulates the lab environment. Because there is no DNS server in the testing environment, we use the IP address in the entire test.

4.4 Experiment Procedure and Data

Before we can debug any code in a VB.NET application, you must enable the debugger (Jones 2002). Within an ASP.NET project, we need to check the web.config file to make sure that debug is set to true (<compilation defaultLanguage=”vb” debug=true>). We need to make sure that the project

Table 2. The experiment summary

Client Computer	Server Computer	Integrated Authentication	Basic Authentication	Results
Admin1	Admin1	Yes	No	Remote debugging works.
Admin1	Admin1	No	Yes	Remote debugging works. However, need manually attaching to the running aspnet_wp.exe process
User1 User2	User1 User2	Yes	No	Remote debugging works. Need to add the user name and the password to the machine.config file.
User1 User2	User1 User2	No	Yes	Remote debugging works. Need to add the user name and the password to the machine.config file. Need manually attaching to the running aspnet_wp.exe process
The workgroup doesn't make any differences in all the tests.				

is compiled in debug mode. For each project, we must also enable debugging using the Project Property Pages dialog.

Three users were created on server: **admin1**, in administrators group; **user1** and **user2**, in users group. All the users are in Debugger Users group. Three users **(admin1, user1 and user2)** were created on client computer as limited users. A subweb is created for each user on the server, and the user is configured as the administrator of the subweb respectively. Integrated Authentication and Basic Authentication were configured on subweb respectively. The client and server are configured in same and different workgroup respectively. Table 2 shows our experiment results.

4.5 Experiment Results and Discussion

From the above experiment summary we can conclude that

1. The web server and the client computers are not necessarily in the same workgroup.
2. A user in administrators group can debug remotely.
3. Adding a user without administrator privileges to machine.config file will enable the user debug remotely.

So we can solve the problem by creating a user on server and adding it to administrators group. Then students use this account to create web applications on their own subweb. But this solution is good not enough for our lab environment for security reason. By putting an ordinary user account to the machine.config file on the server, the user can debug remotely as well. This is a promising solution to our lab environment. However, the machine.config file only can take one user account. How to make this one user account work for all the students needs further research.

Another possible solution is to make each client computer a local server. This solution is not in favour for two reasons. The first one is that students won't have a chance to experiment the two tiers or three tiers client-server architecture. These architectures are very popular in industry. The second is that a student may float around over all the computers; it is not practical for a student to create a website on all the computers. So each student may need to carry a hard drive to the classes, which is inconvenient.

5. THE IMPACTS ON THE STUDENTS

We realised that lab environment settings do affect students' learning. We should encourage students to make use all the possible settings, so that they can get a complete picture of software.

From our observation and communication with the students, we found that the students have got different picture for ASP.NET from the different environment settings. This is highlighted when dealing with different working models in ASP.NET.

ASP.NET supports two working models: one page model, in which controls and VB.NET codes are integrated in a single Webform page; two page

model, in which controls are grouped in a Webform page and VB.NET codes are grouped in code-behind page. In the first semester of 2003, VB.NET was mainly used as an editor and FTP was mainly used for client-server communication because of the permission problem. As a result, one page model was used by the most of the students in their assignment. Some of the students even didn't know there is a two page model, although we have demonstrated the model in the class. On the other hand, in the second semester of 2003, the permission problem was solved, VS.NET was used to generate and compile the two page mode Webform. As a result, all the students used two page model for their assignment. Some of the students even didn't know there is a one page model for a while.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we described and discussed the two problems we have encountered in setting up ASP.NET development environment. The F: drive solution for the permission problem has been proven a successful solution. Our experiments have proven that there are solutions to the remote debugging problem.

What we have learned from this process are:

- As ASP.NET is a new software development platform, everybody needs time to learn its settings. This includes both experts and ordinary people.
- Lab environment settings do affect students' learning
- To achieve successful software settings in a classroom lab, good network knowledge is needed. This is consistent with Agarwal *et al* (2001).
- It is necessary to establish an experiment environment and try all the possible setting options before implementing new software settings with client-server system architecture in a classroom lab.

In the future, we'll try to implement the solutions for the remote debugging problem in our classroom lab. We also need to refine our approach in solving these problems to make it reusable.

ACKNOWLEDGMENTS

Our thanks to our Web technologist, Richard Eltringham , for providing useful information of our current Web server settings during the work of this paper.

REFERENCES

Anderson, R.E. (1992) "Social impacts of computing: Codes of professional ethics". Social Science Computing Review, 2(2):453-469.

Agarwal, K., Critcher, A., Foley, D., Sanati, R. and Sigle J. (2001) "Setting Up a Classroom Lab", The Journal of Computing in Small Colleges Volume 16, Issue 3, March 2001, pp281-286.

Goodyear, J., Peek, B. and Fox, B. (2001) Chapter 7 of "Debugging ASP.NET", ISBN 0735711410, Published by Que, 1st Edition, October 2001

Kiely, D. (2003) "Clean Out VSWebCach", Visual Studio Magazine January 2003 Issue. Accessed November 16, 2003. http://www.fawcette.com/vsm/2003_01/online/hottips/kiely/

Microsoft, (2003a) "Debugging Web Application on a Remote Server", Visual Studio .NET 2003 Online Help System. Accessed November 20, 2003.

Microsoft, (2003b) "ASP.NET Debugging: System Requirements", Visual Studio .NET 2003 Online Help System. Accessed November 20, 2003.

Microsoft, (2003c) "Setting Up Remote Debugging", Visual Studio .NET 2003 Online Help System. Accessed November 20, 2003.

Jones, A. R. (2002) Chapter 9 of "Mastering ASP.NET with VB.NET", SYBEX, ISBN 0-7821-2875-0