Performance and configuration guide for the Microsoft Learning Gateway Solution on HP ProLiant servers

Overview

The Microsoft® Learning Gateway (MLG) solution comprises a set of Microsoft server products that deliver a web-based portal solution that is highly scalable, but also supports deployment in smaller school scenarios.

The solution is based on the Microsoft SharePoint Portal Server 2003 (SPS2003) and Windows® SharePoint Services (WSS) platform, and delivers relevant information and services via user roles and school membership. As part of the MLG solution, Microsoft has created a set of Web Parts and customizations that provide education-specific functionality on top of the standard SharePoint features.

Microsoft Class Server V4 is integrated within the portal to deliver an online Learning Management System (LMS) enabling teachers to create lesson plans, manage Classes and perform curriculum assessment online and offline. Students can participate in lessons and complete assignments offline via the portal, both from home or at school. Administrators can perform tasks such as defining the school infrastructure, adding Classes/students/teachers, managing student and teacher accounts, etc.

Additional optional functionality can be added. Microsoft Exchange 2003 can provide email, calendaring, etc. supported via a web browser through Outlook Web Access. Real-time collaboration using Instant Messaging can be provided through Microsoft Live Communication Server. With Microsoft Office 2003, Instant Messaging is an integrated component of the portal to provide user presence information.

This document is focused on the SPS2003, WSS and Class Server components of the MLG. It is intended to provide an overview of product functionality, deployment, operation and performance. Key topics include:

- Solution architecture
- Web Parts and workflow discussion
- Deployment and installation guidelines and recommendations
- Provisioning and the use of automated procedures
- Performance characterization results and analysis
- Examples of sizing typical scenarios

HP recommends that the information presented herein be used in conjunction with the Microsoft product and solution documentation, along with other information contained in white papers authored by HP and Microsoft. These documents are detailed in the section titled *For More Information*.

Audience

This paper is intended for people who will be proposing MLG solutions, providing installation services or consulting, and who may be assisting educational professionals and administrators in deploying the MLG educational solution on HP ProLiant systems. The document assumes that the reader has an understanding of SharePoint Portal Server 2003 (SPS2003), Windows SharePoint Services (WSS) and Microsoft SQL Server 2000; and is familiar with typical SPS/WSS administration and deployment tasks.

Solution architecture

The MLG solution utilizes the basic SPS/WSS portal functionality and services, and also provides a superset of capabilities. SPS/WSS technology is used to provide the portal framework and team collaboration web sites throughout the solution. The extra functionality provided by the MLG customizations and by the Class Server product provides a specific set of role-focused functions that support a formalized portal-based education process.

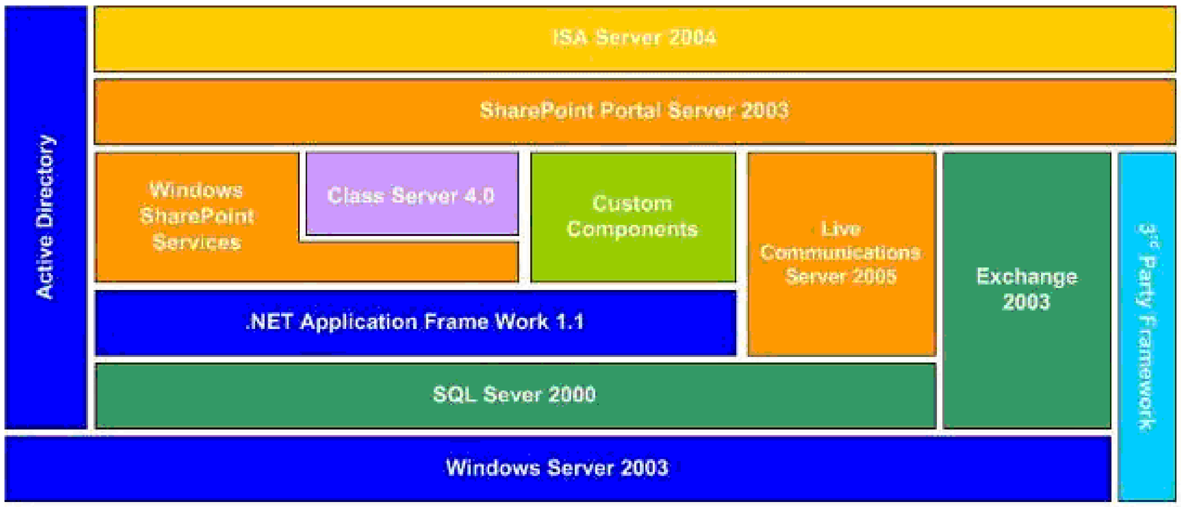
The MLG solution also capitalizes on the SPS/WSS highly-scalable and highly-available architecture. The SPS/WSS prescriptive configurations developed by Microsoft and HP, employing technologies such as load-balanced (WLBS) front-end servers and clustered SQL Servers, are equally appropriate for MLG. Therefore the HP ProLiant and HP BladeSystem 2P and 4P servers that HP recommends for SharePoint portal deployments are also ideal platforms for the MLG solution scale-out architecture. Example platform configurations are discussed in the later section titled *Sizing Solutions*.

The next few sections describe the logical architecture (total solution component layout), the physical architecture (specific MLG and Class Server components and purposes); and the typical server components that would comprise a solution.

Logical architecture

Customers can either implement MLG as an entirely new solution, or can leverage their existing Microsoft infrastructure to provide components of the solution (such as Active Directory, Microsoft Internet Security and Acceleration (ISA) Server, etc.). The Class Server V4 component can also be added to an exiting SPS/WSS portal deployment to provide (add) the education solution-specific functionality. The following diagram (Figure 1) illustrates the components of the full solution.

Figure 1. Learning Gateway logical architecture

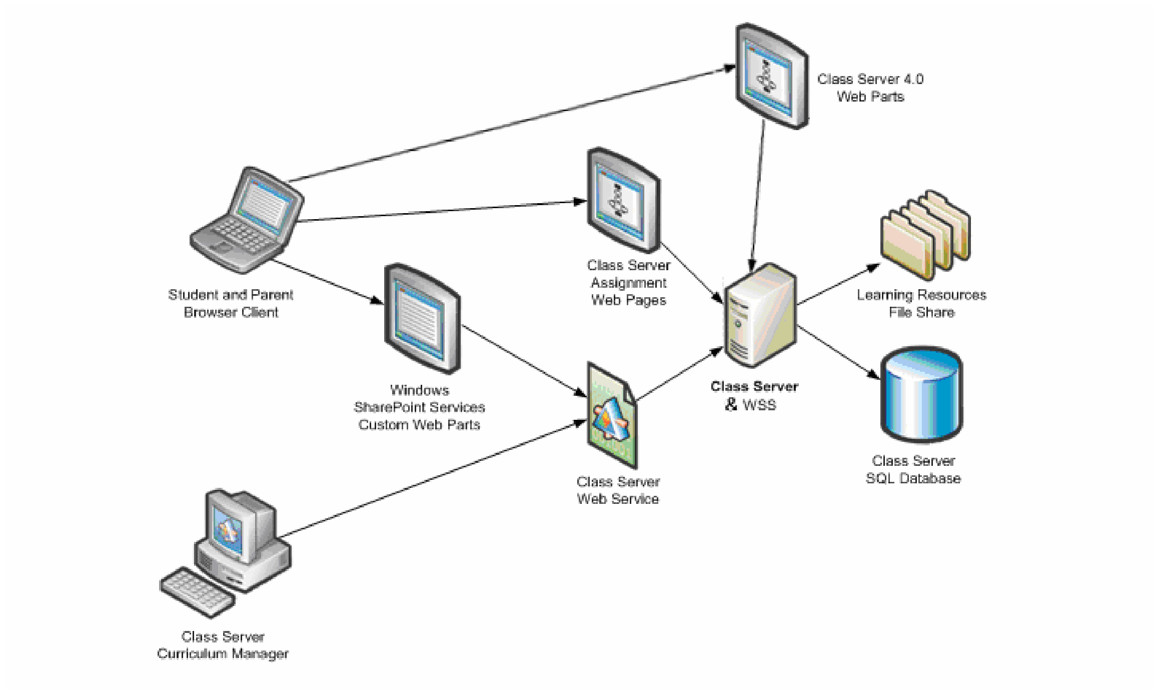


As noted earlier, this document assumes the user has knowledge of deploying SPS2003, WSS and SQL Server 2000; and will focus on the addition of the Class Server components and the required configuring, deployment and provisioning of the MLG solution. Other white papers discussing SPS2003, WSS, SQL Server, etc. are noted in the section *For More Information.*

Physical architecture

Figure 2, below, depicts the key physical components of the solution. The intent is to show how the technology provides specific capabilities to the different users – administrators, teachers, students and parents.

Figure 2. Class Server architecture



In general, users interact with the portal sites via both WSS and customized Web Parts on the portal (school) site pages hosted by the web services on Class Server/WSS front-end servers. Additional storage of information specific to the solution is also provided via a file store to contain Learning Resources, and the school databases on SQL Server. Note that Class Server must be installed on every front-end server that is running SPS/WSS. The following summarizes some key solution components, and a later section will further explore the overall portal and sites architecture.

Class Server and Custom Web Parts

The MLG contains a significant number of custom SharePoint and Class Server Web Parts that were developed specifically for this solution. The list below highlights some of the capabilities these provide.

- Sign-In – the Sign-In Web Part solicits user credentials when the user first accesses a school page. It is also present, but hidden, on other pages and can thus provide these credentials and the user role to other Web Parts that may display role-based information. For most real-world deployments, an ISA server will be used to provide user authentication forwarding features. As this method of forwarding credentials on behalf of the client works only for Basic authentication, all users will have to sign in to the portal; rather than use the AD authentication Class Server option.

- Assignments – these Web Parts form a significant part of the solution. Multiple related Web Parts provide assignment lists, views, properties, editing, student progress summaries, etc.
- Search – this is a superset of the WSS Search Web Part, allowing search by an education-related hierarchy (e.g. district, school, Class, etc.).
- "My…" information – this is a set of Web Parts that provide role-based information to the specific user. Examples are My Classes, My Assignments, My Communities and My News.
- Other Web Parts provide further user-specific information, such as lists of WSS team (Class) sites that a user has access to.

Class Server Assignment pages

These pages allow the teacher to create, view, modify and assign lesson curriculum material to students.

Class Server Learning Resource Material (LRM)

Curriculum materials will typically comprise so-called Learning Resources. These are files of a specific format (having an LRM file extension) and, once imported, are listed in a customized Document Library Web Part called the Curriculum Library. This library is created at the portal root level by Class Server when a school is first created via the Administration applet. These Learning Resource files represent tasks (lessons, homework, tests, etc.) that can be assigned to Class students. They are typically self-contained learning material and may include multiple-choice quizzes to test the student's grasp of the material. They can also therefore be graded (automatically, or by the teacher). Several 3rd party suppliers exist for these resources. A set of about 200 such LRMs was used during the performance testing of this solution, which were example LRMs provided by BroadEducation.com.

Administration

A dedicated administration applet and other related web pages allow administration of the solution. These are accessible by administrators, and by teachers who have been granted an administrator role. An administrator will normally perform tasks such as the initial school infrastructure definition, adding users (teachers, students, and parents) and defining the Classes. A teacher could be granted administration rights so they could access and modify Class members (students) and other Class-related tasks. A walk-through of typical administration tasks, and the use of automated provisioning, is presented in a later section.

Class Server SQL databases

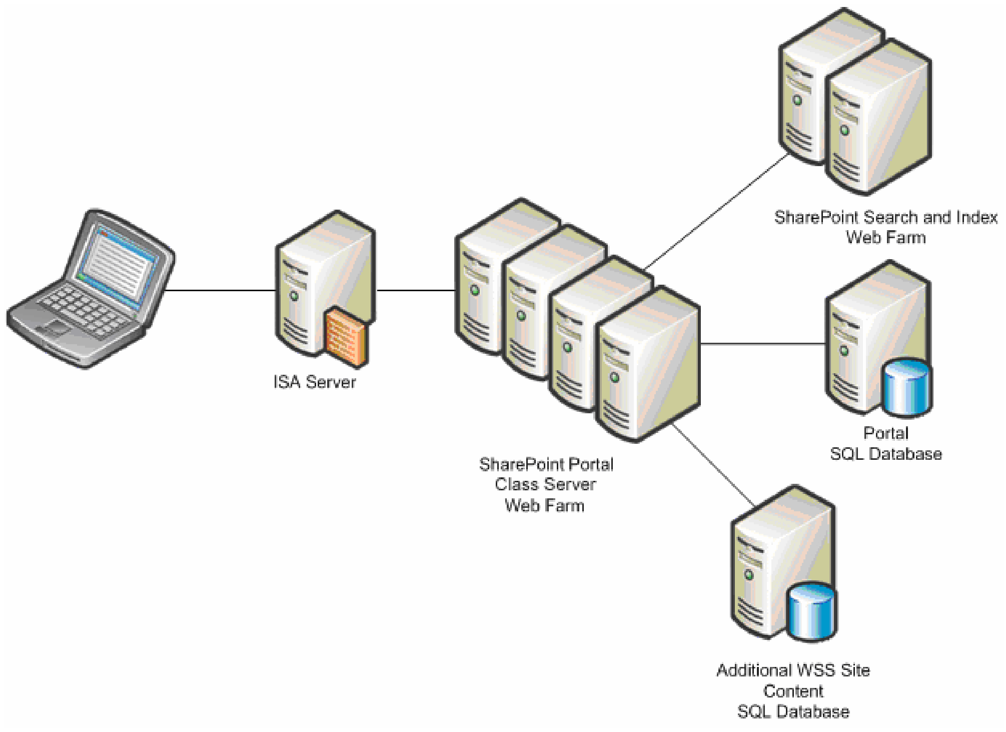
In addition to the databases created and required by SPS/WSS, Class Server also creates and maintains one database per school. In a North American deployment to support a specific school district, a modest configuration with a single active SQL Server may be adequate to handle the required number of school databases. However, in other geographies (such as EMEA) where large numbers of schools are all related to a country's Ministry of Education, it will be more appropriate to deploy the solution across a number of "satellite" sites – each supporting "local" schools. Example platform deployments are discussed in the later section titled *Sizing Solutions*.

The next section describes how the solution components would be deployed on an example hardware configuration.

Server components

The illustration below, Figure 3, shows an example server configuration to support the MLG (SPS/WSS/Class Server) functionality. SharePoint Portal Server will typically be configured as a Web Farm to provide scalability, performance and high availability. As the number of concurrent users increases, additional servers can be added to the farm. This scale-out approach does not impact the site hierarchy (see Figure 4 – Site Hierarchy), as each front-end server in a farm will contain identically-configured virtual webs extended into WSS/SPS. As with traditional SPS and WSS portal deployments, the notion of "role-based servers" is employed.

Figure 3. Solution server components



Web farm

This comprises a number (minimum of two for high-availability) of front-end servers running WSS and Class Server (and optionally SPS). These are typically 2P servers (e.g. HP ProLiant BL20p or HP ProLiant DL380) and provide the web services (solution functionality) to the clients. The number of servers in the farm is predicated by the required total throughput (performance), discussed in more detail in the *Performance* section.

Search and Index servers

If SharePoint Portal Server (SPS) is included in the solution to provide search/indexing, personal web sites (My Site), content aggregation, etc. for a school district, then the search and indexing roles should be hosted by separate servers for best performance. These are typically 2P servers (e.g. HP ProLiant BL20p or HP ProLiant DL380).

SQL Database servers

The MLG solution creates and utilizes a number of SQL Server databases – comprised of those to maintain the context of the various portal and team sites, the Class Server school databases, and the WSS site content databases. Thus, in a large solution deployment, it may require more than one active SQL Server to ensure required performance. Depending upon the specific situation and geography, either an “N+1” active/passive SQL cluster (centralized deployment), or a “hub and spoke” (distributed) model may be appropriate. For example, in a large country-wide scenario a central admin site with multiple school area satellites would likely be used, each site having their own SQL Servers. These can be 2P servers (e.g. HP ProLiant BL20p or HP ProLiant DL380), or 4P servers (e.g. HP ProLiant BL45p or HP ProLiant DL580).

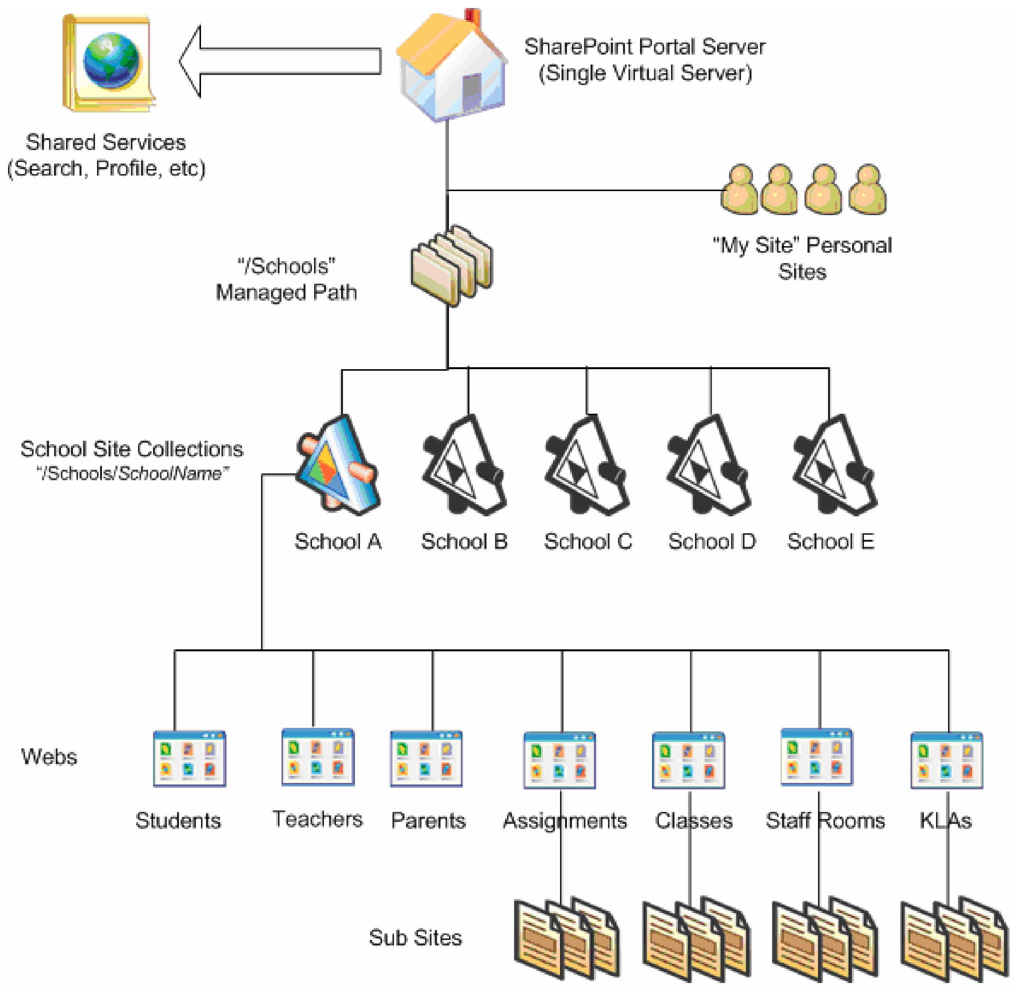
ISA Server

ISA Server provides functions such as namespace integration, pass-through authentication caching, firewall, intrusion detection, etc. As this server is exposed to the external network, it is strongly recommended not to add any additional services to this server. The MLG solution supports Basic Authentication over SSL with limited support for Windows Integrated authentication (Kerberos/NTLM). To provide the “single sign-on” facility to access all areas of the portal, ISA Server is used to consolidate all services within a single namespace while enabling Pass-Through Authentication. Multiple ISA servers may be required to ensure high availability. These are typically 2P servers (e.g. HP ProLiant BL20p or HP ProLiant DL380).

Portals architecture

Figure 4, below, illustrates the portal site hierarchy for the MLG solution, and how the school site collections and related web sites fit within the WSS and SPS portal architecture.

Figure 4. SharePoint site hierarchy



The home page for users within the portal is the role-based WSS site for the particular school that the user belongs to. For example, a teacher's home site will be the "Teachers" WSS site for their school (see above illustration). By using WSS as the primary home page for users, the portal is able to provide a unique user experience across the schools. The architecture also enables a large number of school portals to be hosted on a server farm.

In the solution shown above, SPS provides shared services to the WSS school sites in the form of search, personal "MySite" and centralized document lists.

Custom Web Parts

As discussed above, many Web Parts display information based on the specific user credentials. This relates to both the user as an individual, and the user's role in the education system (e.g. teacher vs. student). These are generally referred to as interstatial Web Parts, and while the visual results they display on a page may appear similar to the more simple WSS Web Parts, there is often significant logic (Web Part code) behind their operation. They can therefore be somewhat more resource consumptive than the simpler WSS Web Parts. The following describes two such examples.

My Classes

At first glance, this appears to display a simple list of hyperlinks relating to Class sites. Mousing-over a link will show the team site (Class) URL on the browser status line, similar to the way a simple "My Links" WSS Web Part will do. However, there is additional logic behind the scenes to cover two specific needs.

- When a teacher clicks on a Class "link" in this list for the first time they are presented with a screen facilitating creating the team site for that Class. Once this site creation is complete (taking typically less than 20 seconds) all users related to that Class can access it (i.e. the Web Part now knows the site exists and can direct users to it). Note that these Class team sites can also be created automatically using the solution's SyncTeamSites feature. SyncTeamSites can be scheduled for each school, thus obviating the need for manual creation of Class sites. This method would be especially useful in larger schools with many Classes.
- Any user (teacher, student, parent) will only see Class links in this list that relate to them (Classes a teacher is responsible for, or that a student is a member of). The Web Part therefore needs to know the user information and display appropriate results. The links are also provided as virtual references, rather than full path references. This Web Part therefore displays a dynamic list of sites relevant to the specific user at that time. As students progress through the school system and are added to or change Classes, the list will always reflect their current situation with respect to specific Class membership.

The functionality is therefore more complex than a simple hyperlink list.

My Assignments and Summaries

These appear as a multi-column object list or summary/count list respectively. However, their content is also dependent on the specific user and their role. The lists are most typically sorted based on the "state" of an assignment as it relates to the user. For example:

- A teacher's My Assignments list will show assignments grouped by states of "assigned to Class", "ready to grade" and "returned". These reflect the states from the teacher's viewpoint of (respectively) creating and issuing the Class assignment, the student(s) having completed it and returned it for grading, and the teacher grading it and returning it to the students.
- Conversely, a student's My Assignments list will show grouping based on states of "to do" (assigned to Class members and due for completion), "submitted" (performed by student and returned for grading), and "complete" (graded by teacher and returned to student).

The summary Web Part shows each "state" and the number of assignments in that state. It also allows the user to quickly select an assignment list in a certain state (e.g. show the "To Do" list, sorted by due date). The functionality is therefore much more complex than a simple WSS object list Web Part and can consume commensurately higher resources (mainly CPU).

Assignments workflow

The lifespan of an assignment (assign, perform, grade, etc.) can best be thought of as a workflow. The various Web Parts that facilitate assignments are therefore potentially complex and consume more server resources (notably CPU) than simple WSS Web Parts. However, the incidence (frequency) of these Web Parts being used is likely to be much less than other common functions such as browsing/reading material, discussions, reading news, etc. The performance and solution configuration/sizing implications are discussed in detail in the *Performance* section.

The following describes the typical “workflow” of assignments, so as to explain some of the more complex functions that the related MLG Web Parts perform.

Create Assignment

A teacher will typically browse the Curriculum Library to select a relevant Learning Resource (LR) to assign to a Class, this LR relating to the Class and subject. It may be set as a learning exercise (read), a quiz or homework assignment, etc. The teacher will then click “Assign” and be presented with a list of Classes they are responsible for. Having chosen a Class, they will now see a list of Class students. The default would be to assign the LR to all Class students. This action will change the state to “assigned” and the assignment will appear as “Due” in each Class student’s “To Do” sub-list in their “My Assignments” list. It will also appear in the teacher’s “My Assignments” list as “assigned” as discussed earlier.

It is important to note that this assignment is now tracked through the system via an “Assignment ID” (AID) as initiated by the teacher; and also by a related “Paper ID” (PID) for each student’s individual instance of the assignment. So in a Class of 30 students, the teacher initiating a single assignment results in 30 instances of that assignment (one per Class student) being created. A teacher can also monitor assignment progress by individual students by drilling-down into their “assigned” list. The implication is that there are now 30 more items in the system that the various Web Parts need to track and display appropriately in a number of ways. Multiply that example 30 students by the number of assignments per Class per day, Classes per school, etc. and the number of such workflow-related items to manage and display across the whole system quickly grows to a large number and can be resource consumptive. Note that there is an option in Class Server to restrict the size of the assignment returned from the student (default is 10MB). Reducing this value will lower resource and network bandwidth consumption to some degree.

Perform Assignment

Each Class student would now pick the “Due” assignment from their To Do list and perform the related Learning Resource (LR). This is accomplished by another custom Web Part that enables viewing and interacting with the LR (e.g. answering imbedded multiple-choice questions). Once complete, the students each click to return the completed assignment. It again changes state (completed) for each instance (specific student’s copy, tracked by AID/PID) of the assignment, and each is now ready for grading by the teacher.

Grade each student Assignment

Each student’s completed assignment will now be listed in the teacher’s “ready to grade” list, sorted by Assignment and then by Student. The teacher can dispose of these in a number of ways. If the assignment (LR) is auto-graded, the teacher may choose to simply “return all” (implicit auto-grade and return all) in the list. They may also choose to open/read any individual student’s assignment and enter comments. In either case, the state of each student’s instance of the assignment will once again change state (returned). The grade awarded will also be applied to each student’s school/Class record.

Review Assignment

Each student can now review their graded assignment, displayed in their returned list. The workflow is now complete.

As can be seen, the solution Web Parts that handle the workflow and display assignment lists have significant work to do. Each such Web Part must take into account the user identity, user role, and state of any given instance of an assignment; and also sort them appropriately into the relevant display lists dynamically. As these Web Parts are more resource consumptive (CPU) it is important to develop as accurate a workload description as possible when sizing a solution. This is described further in a later section covering performance and sizing.

Deployment and installation

The MLG solution will usually be deployed onto a web farm comprising a number of front-end servers. These will already have WSS (and optionally SPS) installed if a SharePoint portal installation already exists, and the customer wishes to add the MLG functionality. In the case of deploying the MLG solution from scratch, WSS (and SPS if required) must be installed prior to installing Class Server. The process for this is extensively documented in the *Microsoft Solution for Intranets* documentation set, authored by HP, and available from the Microsoft web site as noted in the *For More Information* section at the end of this document. The deployment and installation recommendations presented herein assume that the WSS/SPS installation and configuration has already been successfully accomplished and tested.

Deployment

The MLG solution will typically be deployed onto a set of servers that resembles a WSS/SPS "medium" or "large" prescriptive configuration, designed for high availability. This will comprise a number (minimum of two) of front-end servers acting in a cluster via WLBS. Databases will be on SQL Server active/passive or "N+1" clusters. Separate Index and Search servers may also be deployed, as well as an optional file store server to contain LRM and other reference material. Example platform deployments are discussed in the later section titled *Sizing Solutions*.

When following the guidelines below, note that Class Server must be installed on <u>all</u> web front-end servers that have WSS/SPS installed. The Class Server product installation is a simple, quick process. A later section describing Provisioning will document an automated process that ensures consistency of content and configuration across all such front-end servers.

Installation

Before installing Class Server on each web front-end server, there are a number of prerequisites:

- WSS (and optionally SPS if required) must be installed and configured on each front-end server.
- As the MLG solution runs from the Default web site, this default virtual web <u>must</u> exist within IIS on each front-end server. If, for whatever reason, the default web site has been deleted it is possible to create a new site and edit the site metadata such that it appears as

the default web site (site identifier = 1 in the IIS meta-base). Microsoft has published an article describing this process.

- ASAPI extensions must be enabled on each front-end server. Within the IIS Services Manager Console, click on the folder "Web Service Extensions" in the left pane, and then click to enable ASAPI Extensions in the right pane.
- HP recommends using the WSS Central Administration Console to extend the Default web site into WSS prior to installing Class Server. This ensures that the Class Server product installation and subsequent configuration can be completed without interruption. The configuration phase (and creation of schools) utilizes WSS functionality and needs to add required web pages and Web Parts to the Default web site virtual web folder structure. The Default web site must have already been extended into the WSS portal before this can successfully occur.

To install Class Server V4, execute the following steps:

- Insert the Class Server CD in the drive. In the **Welcome** screen, click **Next**.
- On the **Select a Setup option** page, click **Install Class Server**, click **Next**.
- Follow on-screen instructions until you get to the **Install and configure Class Server** page.
- On this page, click **Install Class Server**.
- Follow the on-screen instructions, and then click **Finish**.

You can continue to install Class Server on each front-end server, but do not execute the "Configure Class Server" operations on any of these subsequent servers. Initial school configuration on the first front-end must be performed via the "Configure Class Server" option, however configuring the remaining servers is best accomplished using an automated provisioning methodology. This will ensure an error-free and consistent configuration across all front-end servers.

The configuration/provisioning process, and the use of automated procedure, are discussed in the following sections.

Provisioning

Once Class Server is installed, there are a number of administration and provisioning tasks to perform. These involve defining a first school, its Classes, students, teachers, etc. and then adding further schools as needed. Learning Resource materials can also be imported to the Curriculum Library.

Administration tasks

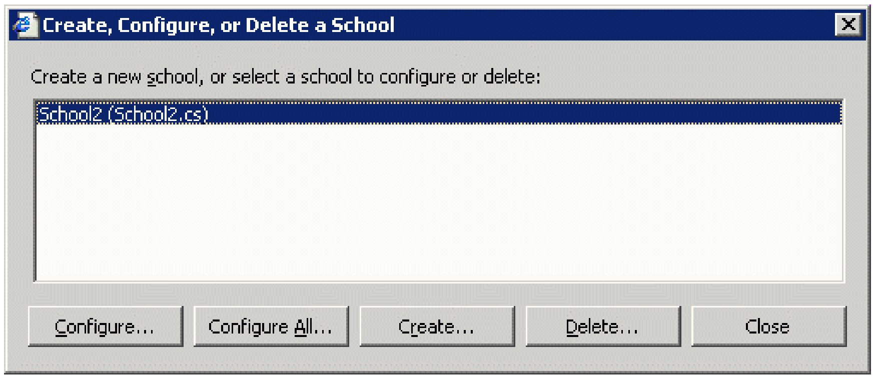
The first step is to define the first school on one of the front-end servers, and then to add the users and Classes for that school. Complete all the steps below up to "Adding Learning Resource material" on one of the front-end servers. You will then use automated procedures to clone the relevant server and school configuration information to the remaining front-end servers.

Creating Schools

To create the first school by using the Class Server Configuration option within the installation/configuration procedure, continue from the installation procedures described above. At this point you should be on the **Install and Configure Class Server** page.

Click **Configure Class Server**. (Note – If on completion you exited the installation procedure above, you can also start Class Server configuration by double-clicking the **Class Server Configuration** icon on the desktop). You should now see a prompt screen as shown in Figure 5, below.

Figure 5. Configure Class Server - Schools


Create, Configure, or Delete a School

Create a new school, or select a school to configure or delete:

School2 (School2.cs)

Click on **Create...** to define a new school. The example above shows that a school (School2) has already been created. To make modifications to a highlighted school, you would click on **Configure...**

If prompted to create a new school database, click **Yes**. You will only need to create a school database once, as when configuring other front-end servers the correct database name will be supplied via the automation procedures detailed later. Note that each school must have a unique database name. You will now see the school configuration screen as shown in Figure 6, below.

On the **School Information** tab, type the name of the school (e.g. Nashua High School) in the **School title** box.

Select your geographic region (e.g. United States) from the **Region** list.

Select **SharePoint site** as the Web site option from the **School Web Site** list.

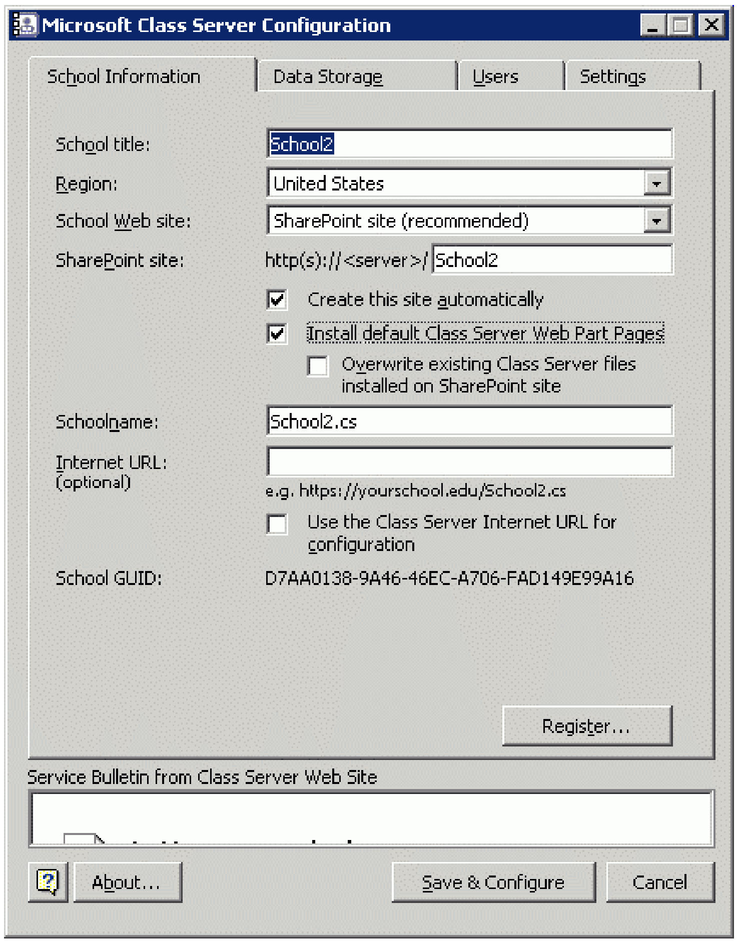
Enter a short school name (e.g. Nashua) as the site name in the **SharePoint site** box. The **Schoolname** field will be automatically filled-in.

Check the **Create this site automatically** and **Install default Class Server Web Part Pages** boxes.

Take note of the **School GUID** as shown on this screen, and the school it refers to. You will need that GUID string when setting up the automated provisioning procedures. Once the school has been created, the school GUID may also be obtained from the Windows Registry from the following key:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Class Server\Server\Schools\School_GUID

Figure 6. School configuration – School Information



School title: School2
Region: United States
School Web site: SharePoint site (recommended)
SharePoint site: http(s)://<server>/ School2
[✓] Create this site automatically
[✓] Install default Class Server Web Part Pages
[ ] Overwrite existing Class Server files installed on SharePoint site
Schoolname: School2.cs
Internet URL: (optional)
e.g. https://yourschool.edu/School2.cs
[ ] Use the Class Server Internet URL for configuration
School GUID: D7AA0138-9A46-46EC-A706-FAD149E99A16

Now click on the **Data Storage** tab to enter database and storage information. These will likely be the same for all schools served by the web farm, except for the unique school database name.

Enter the name of the **Database Server**. This may be a SQL Server name, or the cluster alias name of a SQL Server cluster.

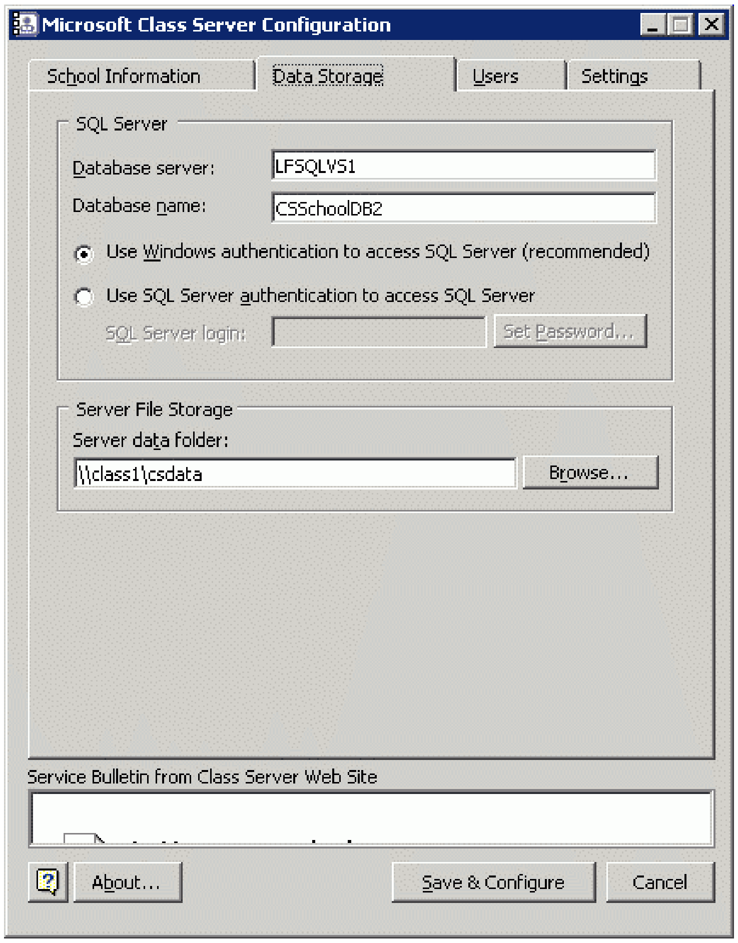
Enter a unique **Database name** for the school. Each school has its own database.

Check the **Use Windows authentication** button, unless your IT practices dictate otherwise. This will use Basic authentication over (optional) SSL.

Enter the name of a network share to use as the **Server File Storage**. This is of the form \\<Server>\<Sharename>.

Refer to the example in Figure 7, below, and enter the required data.

Figure 7. School configuration – Data Storage


Microsoft Class Server Configuration

SQL Server

Database server: LFSQLVS1

Database name: CSSchoolDB2

[✓ Use Windows authentication to access SQL Server (recommended)]

Use SQL Server authentication to access SQL Server

SQL Server login:

Server File Storage

Server data folder: \\class1\csdata

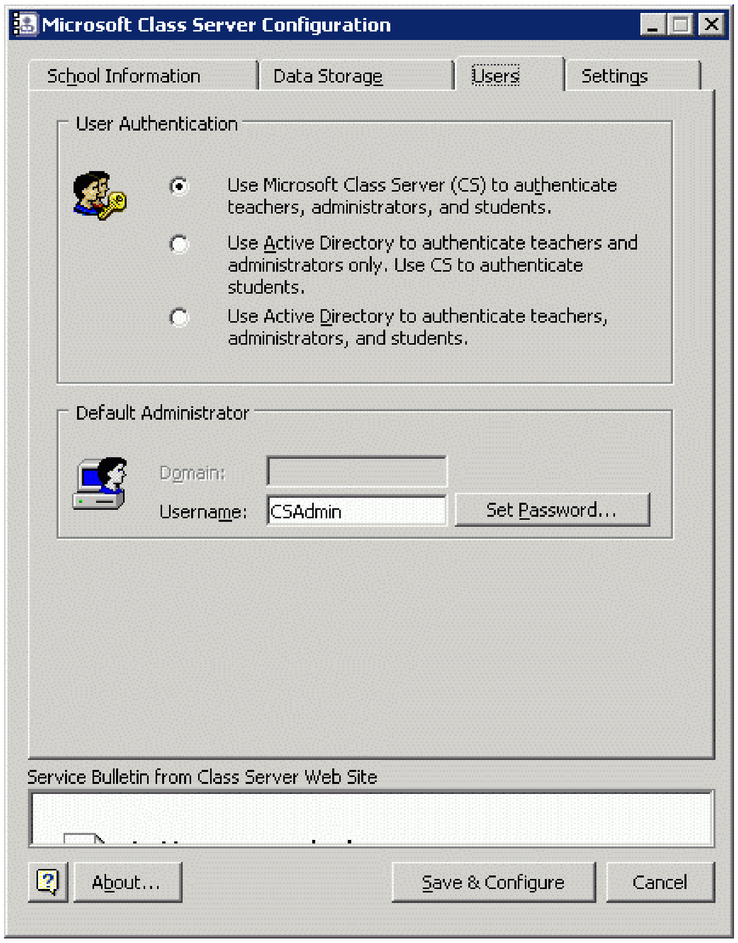
Next, click on the **Users** tab to enter authentication choices and the Administrator password.

Click on a radio button to pick your desired **User Authentication** methods. There are three choices, providing selection of either using Class Server sign-in authentication, using Active Directory, or using a mix of those two methods.

Enter a username and password for the **Class Server Administrator** account. The default username is CSAdmin.

Refer to the example in Figure 8, below, and enter the required data.

Figure 8. School configuration – Users



Microsoft Class Server Configuration

User Authentication

- [✓] Use Microsoft Class Server (CS) to authenticate teachers, administrators, and students.
- [ ] Use Active Directory to authenticate teachers and administrators only. Use CS to authenticate students.
- [ ] Use Active Directory to authenticate teachers, administrators, and students.

Default Administrator

Domain:

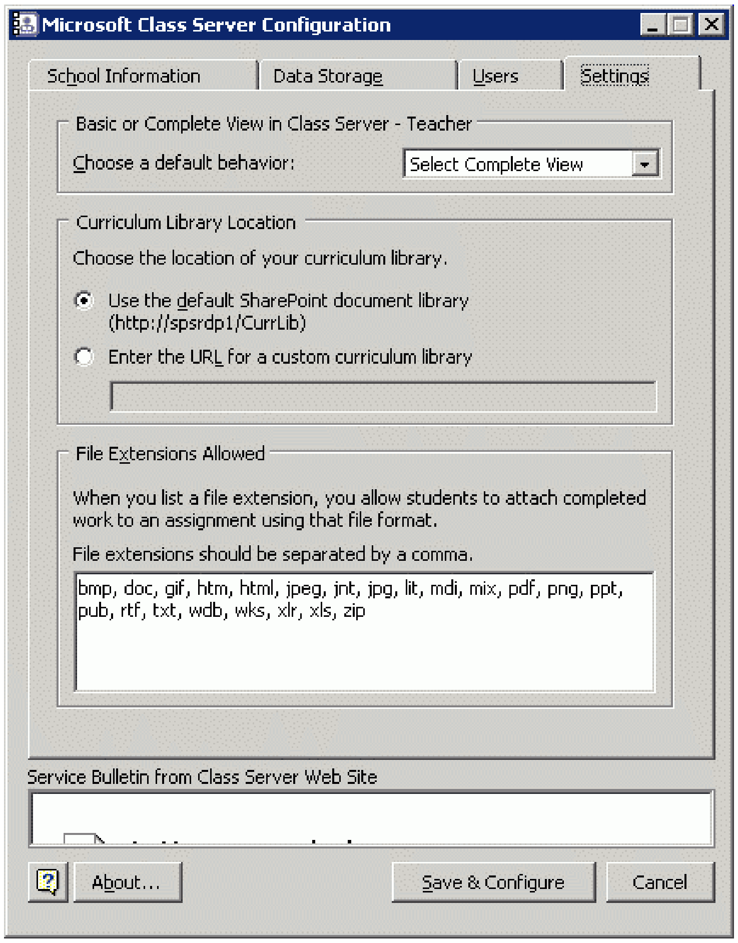
Username: CSAdmin

Click the **Settings** tab and enter any changes needed. It is likely that the default settings will be acceptable.

Using the default SharePoint document library for the Curriculum Library location is recommended, as this will place the library at the web farm root level and it will be accessible in the same manner from all schools (common material is shared).

Refer to the example in Figure 9, below, and make any required modifications.

Figure 9. School configuration – Settings



Basic or Complete View in Class Server - Teacher

Choose a default behavior: Select Complete View

Curriculum Library Location

Choose the location of your curriculum library.

(●) Use the default SharePoint document library (http://spsrdp1/CurrLib)

( ) Enter the URL for a custom curriculum library

File Extensions Allowed

When you list a file extension, you allow students to attach completed work to an assignment using that file format.

File extensions should be separated by a comma.

bmp, doc, gif, htm, html, jpeg, jnt, jpg, lit, mdi, mix, pdf, png, ppt, pub, rtf, txt, wdb, wks, xlr, xls, zip

Service Bulletin from Class Server Web Site

Finally, click the **Save & Configure** button, then enter any requested passwords. The school database will be created and specified configuration parameters will be stored. The process typically takes less then 30 seconds.

When configuration is complete, the **Microsoft Class Server URL Listing** dialog box appears. Make a note of the URLs in this list, as you will need them later.

You can now continue with the remaining administration operations using the **Class Server Administrator** web page. To start this, enter the following address in your browser:

http(s)://<servername>/<school>.cs/Admin

where <servername> is the system name of the first front-end server you are configuring, and <school> is the short school name you entered to define the school web site. This is also the Administrator URL shown in the Microsoft Class Server URL Listing box, cited above. Note that later, when you have configured all front-end servers, the <servername> can be the web farm cluster alias name (e.g. District).

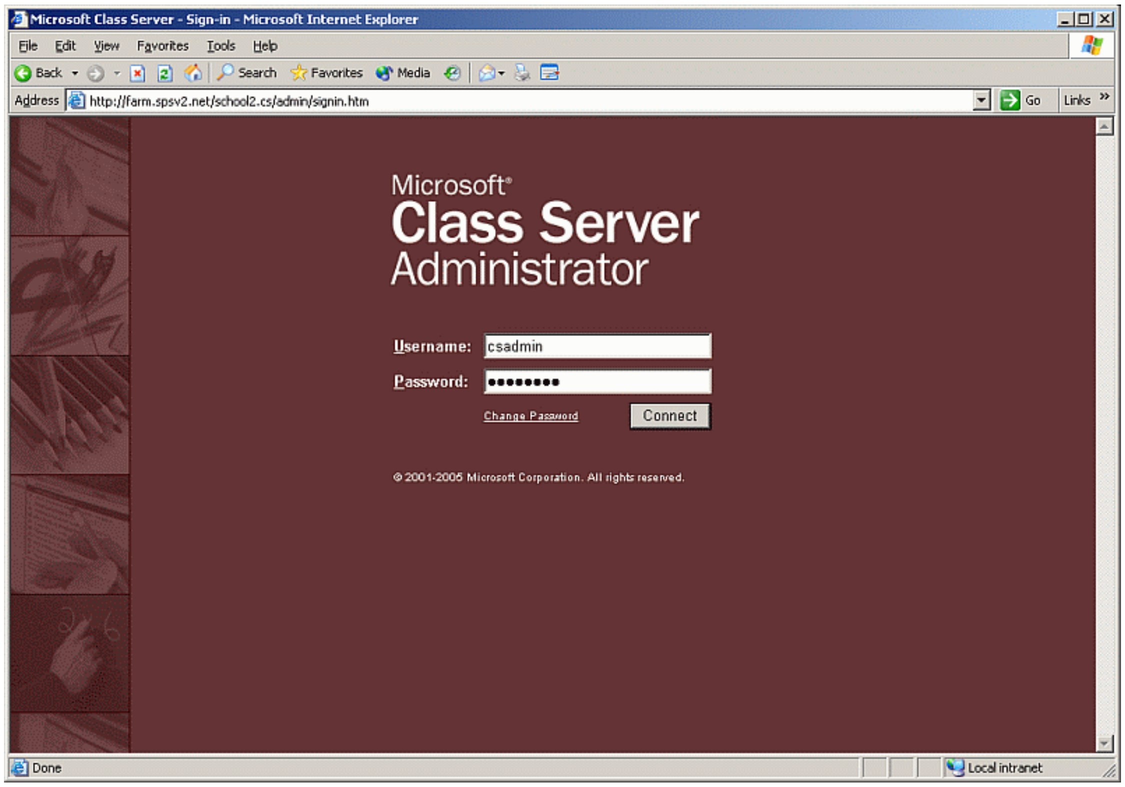
Note that if you need to utilize SSL (Secure Sockets Layer) on Class Server, this requires that you have the server NetBIOS name on the certificate – or else you will not be able to configure the school. This method is not compatible with ISA server which requires the FQDN name of the website that will be published through the ISA server. One way to solve this dilemma is to use one NetBIOS certificate-name to be used when configuring schools in Class Server, then change it to an FQDN certificate-name once configured and when publishing it through ISA.

Adding users and classes

Perform the following steps to enter example users and a Class so you can test the school configuration. You can add the real users and Classes later using an automated provisioning methodology, described in detail in a later section of this paper.

When the Class Server Administrator web site opens, sign in with the username and password you specified during Class Server installation. The default username is CSAdmin.

Figure 10. Class Server Administrator – Sign In

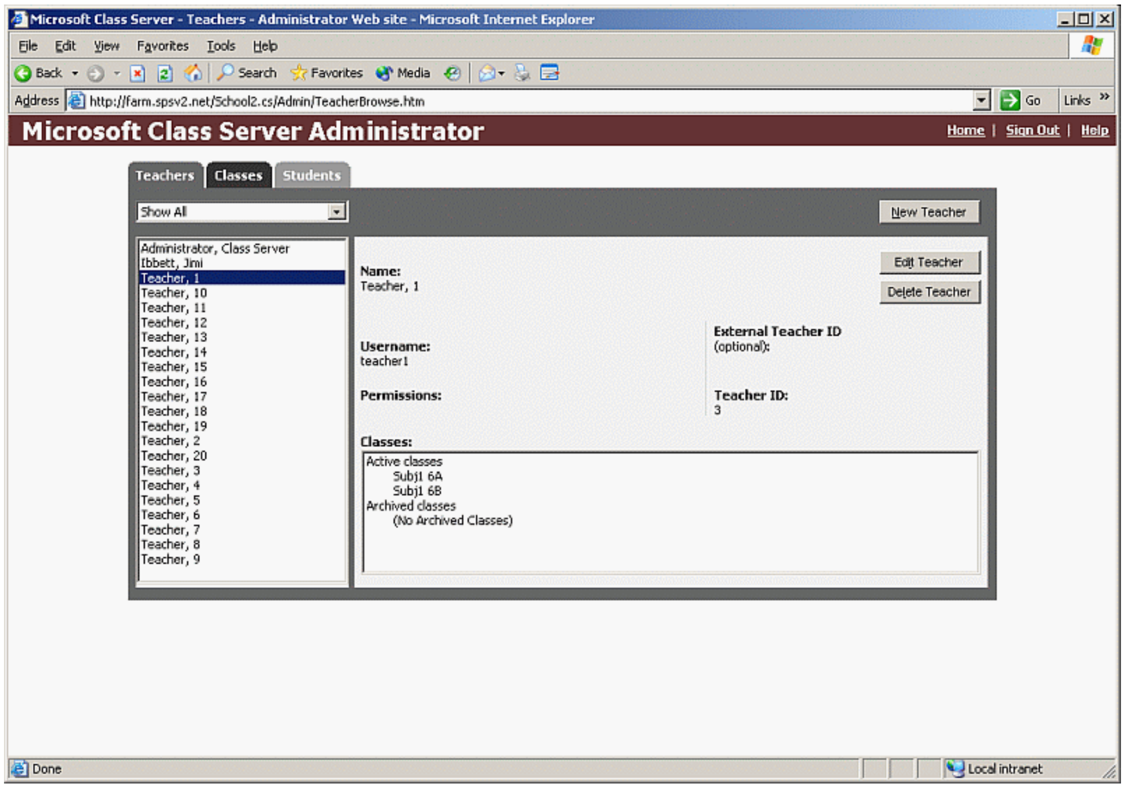

Microsoft Class Server - Sign-in - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://farm.spsv2.net/school2.cs/admin/signin.htm
Go Links
Microsoft® Class Server Administrator
Change Password
Connect
Done
Local intranet

Username: csadmin
Password: ●●●●●●●●●

© 2001-2005 Microsoft Corporation. All rights reserved.

Once login is complete, click on **Teacher Information** and then add a test teacher, e.g. Teacher1. Figure 11, below, illustrates the Teacher Information Administration page. In the example screen shown, a number of teachers have already been added, as well as Classes for which they are responsible. When first used, this screen would only list the Administrator. Click **New Teacher** and enter required information.

Figure 11. Class Server Administrator – Teachers


Microsoft Class Server - Teachers - Administrator Web site - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://farm.spsv2.net/School2.cs/Admin/TeacherBrowse.htm
Go Links
Microsoft Class Server Administrator
Home | Sign Out | Help
Teachers Classes Students
Show All
New Teacher
Edit Teacher
Delete Teacher
Done
Local intranet

Administrator, Class Server
Ibbett, Jimi
Teacher, 1
Teacher, 10
Teacher, 11
Teacher, 12
Teacher, 13
Teacher, 14
Teacher, 15
Teacher, 16
Teacher, 17
Teacher, 18
Teacher, 19
Teacher, 2
Teacher, 20
Teacher, 3
Teacher, 4
Teacher, 5
Teacher, 6
Teacher, 7
Teacher, 8
Teacher, 9

Name:
Teacher, 1

External Teacher ID (optional):

Username:
teacher1

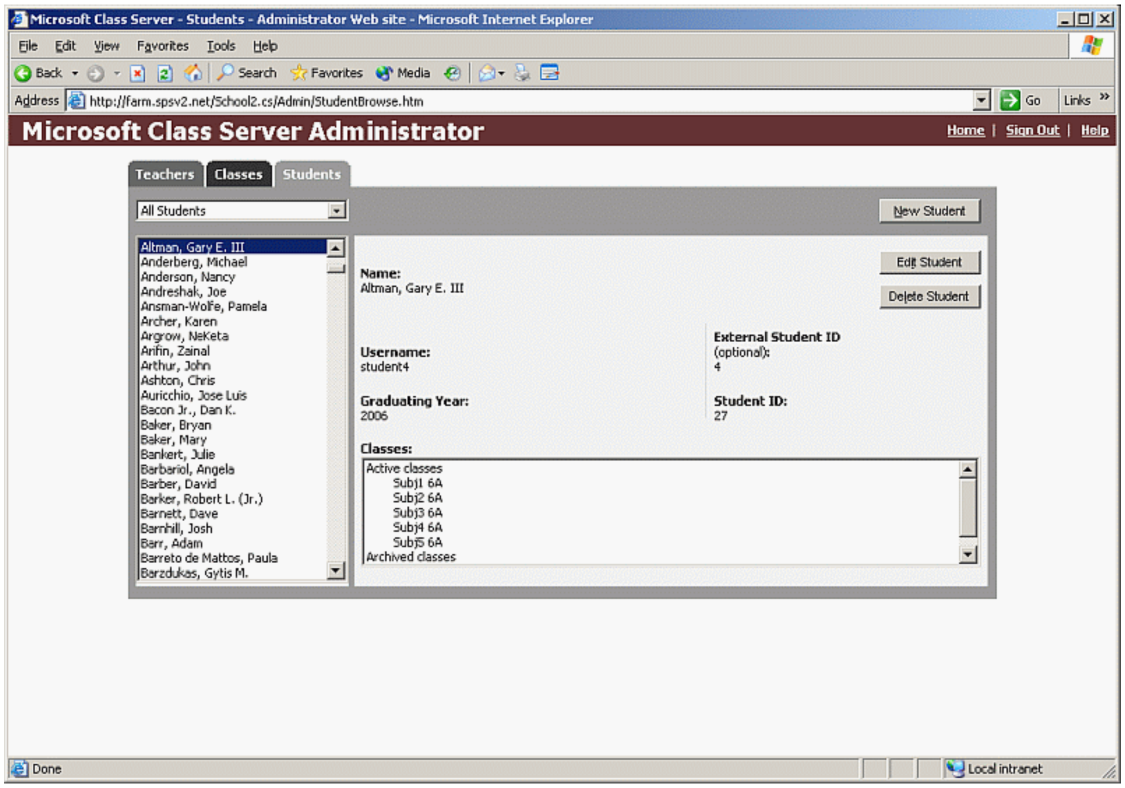
Permissions:

Teacher ID:
3

Classes:
Active classes
Subj1 6A
Subj1 6B
Archived classes
(No Archived Classes)

Now click on the **Students** tab and then add a test student, e.g. Student1. Figure 12, below, illustrates the Student Information Administration page. In the example screen shown, a number of students have already been added, as well as their current active Classes. When first used, this screen would not contain any information. Click **New Student** and enter required information.

Figure 12. Class Server Administrator – Students


Microsoft Class Server - Students - Administrator Web site - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://farm.spsv2.net/School2.cs/Admin/StudentBrowse.htm
Go Links
Microsoft Class Server Administrator
Home | Sign Out | Help
Teachers
Classes
Students
All Students
New Student
Edit Student
Delete Student
Done
Local intranet

Altman, Gary E. III
Anderberg, Michael
Anderson, Nancy
Andreshak, Joe
Ansman-Wolfe, Pamela
Archer, Karen
Argrow, NeKeta
Arifin, Zainal
Arthur, John
Ashton, Chris
Auricchio, Jose Luis
Bacon Jr., Dan K.
Baker, Bryan
Baker, Mary
Bankert, Julie
Barbariol, Angela
Barber, David
Barker, Robert L. (Jr.)
Barnett, Dave
Barnhill, Josh
Barr, Adam
Barreto de Mattos, Paula
Barzdukas, Gytis M.

Name:
Altman, Gary E. III

Username:
student4

External Student ID
(optional):
4

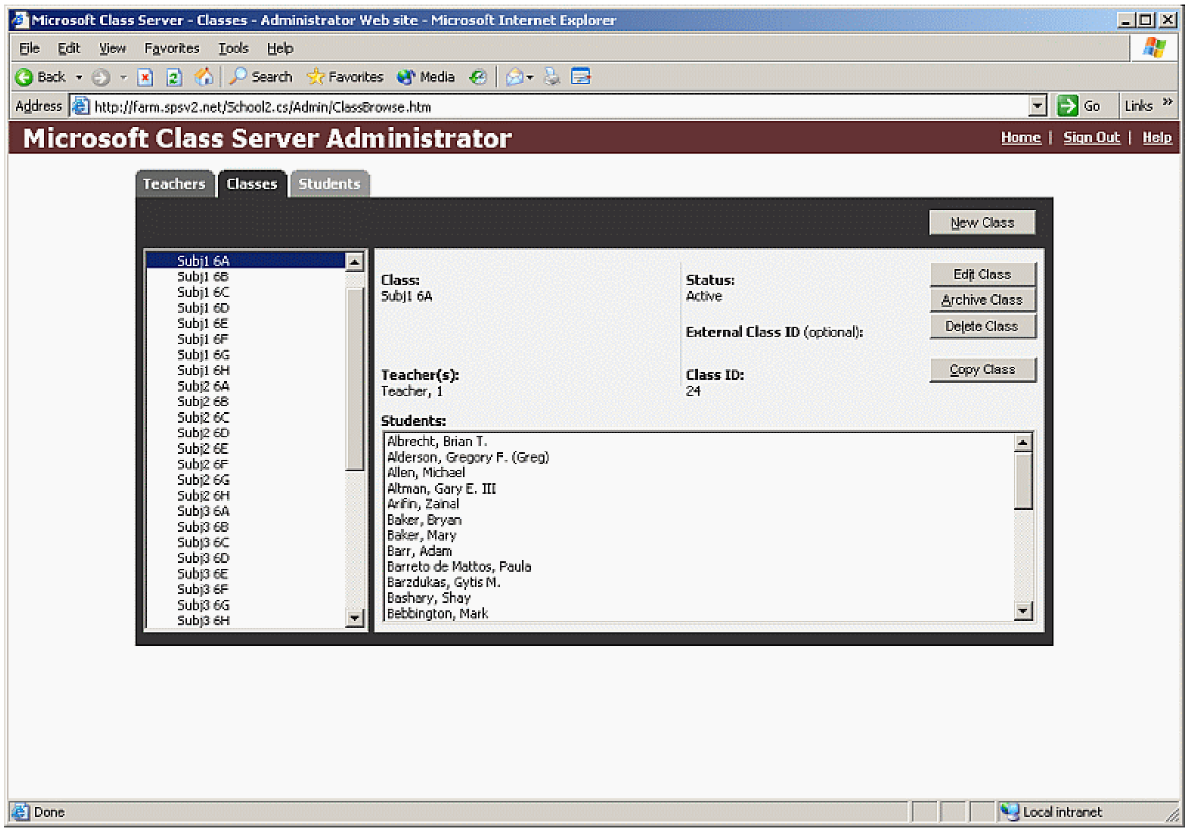
Graduating Year:
2006

Student ID:
27

Classes:
Active classes
Subj1 6A
Subj2 6A
Subj3 6A
Subj4 6A
Subj5 6A
Archived classes

Click on the **Classes** tab to add a test Class, e.g. Class 1. Figure 13, below, shows a listing of current Classes (used for HP lab tests), with the students and teacher(s) being listed for the highlighted Class. When first used, this screen would not contain any information. Click **New Class** to add a test Class and assign the test teacher and test student to that test Class.

Figure 13. Class Server Administrator – Classes


Microsoft Class Server - Classes - Administrator Web site - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://farm.spsv2.net/School2.cs/Admin/ClassBrowse.htm
Go Links
Microsoft Class Server Administrator
Home | Sign Out | Help
Teachers Classes Students
New Class
Edit Class
Archive Class
Delete Class
Copy Class
Done
Local intranet

Subj1 6A
Subj1 6B
Subj1 6C
Subj1 6D
Subj1 6E
Subj1 6F
Subj1 6G
Subj1 6H
Subj2 6A
Subj2 6B
Subj2 6C
Subj2 6D
Subj2 6E
Subj2 6F
Subj2 6G
Subj2 6H
Subj3 6A
Subj3 6B
Subj3 6C
Subj3 6D
Subj3 6E
Subj3 6F
Subj3 6G
Subj3 6H

Class: Subj1 6A

Status: Active

External Class ID (optional):

Teacher(s): Teacher, 1

Class ID: 24

Students:
Albrecht, Brian T.
Alderson, Gregory F. (Greg)
Allen, Michael
Altman, Gary E. III
Arifin, Zainal
Baker, Bryan
Baker, Mary
Barr, Adam
Barreto de Mattos, Paula
Barzdukas, Gytis M.
Bashary, Shay
Bebbington, Mark

Modify the teacher and student entries, if required, to make sure the teacher has access to the Class, and the student is enrolled in the Class.

Click **Sign Out** to leave Class Server – Administrator.

Testing the school configuration

Having entered a Class, teacher and student you can now test the school that you have created and configured. Proceed as follows:

- In your browser, enter http://<servername>/<school>, e.g. http://CSServer1/Nashua. This will bring up the school web site. Note that this is the School Web Site URL from the **Microsoft Class Server URL Listing** dialog box, shown earlier.
- Sign in as Teacher1, and then click **My Assignments**
- Create an assignment for Class1, and assign it to Student1
- Now sign in as Student1, and complete the assignment.
- Sign in again as Teacher1, and grade the returned assignment.
- The above should validate that all is working correctly. You can also experiment with other workflows and Class Server Web Parts.

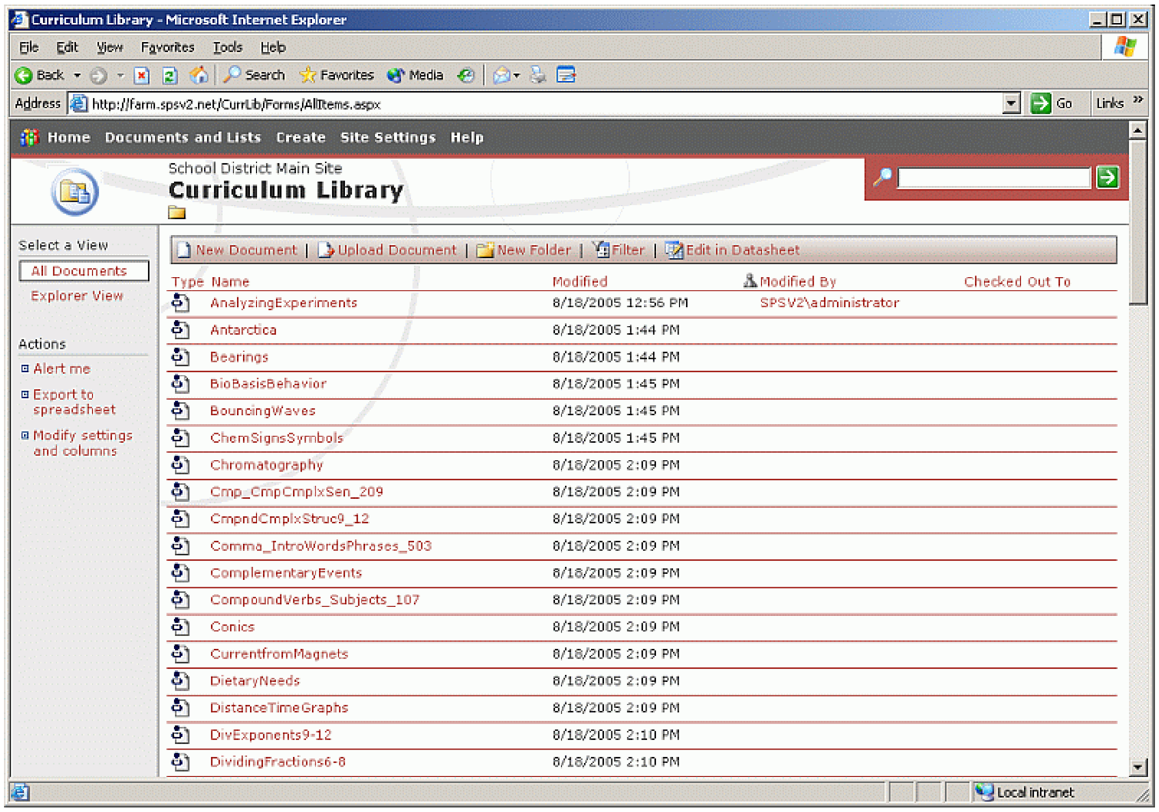
Adding Learning Resource Material

Adding Learning Resource Materials (LRMs) to the Curriculum Library can be accomplished in several ways. You can perform this step once a single front-end server has been fully configured (and a school created), or later when all front-end servers are configured, and also at later times when new material needs to be imported.

Class Server includes a Learning Resource Upload Tool. This is both a command-line tool that you can use to import multiple LRMs to a SharePoint document library, and a Web Page tool that teachers can use to import individual LRMs to the Curriculum Library. Refer to the *Class Server Administrator Guide* (PDF or HTML version) for full details.

You can also browse to the Curriculum Library (default URL is http://<servername>/CurrLib) and click on Upload Document, or Upload multiple files. When uploading multiple files, it is best to do this in modest groups of files, depending on the total file size. Once imported to the Curriculum Library, the LRs are available for use and can be viewed, edited and assign to Classes. Figure 14, below, shown the Curriculum Library web page.

Figure 14. Curriculum Library Web Page


Curriculum Library - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://farm.spsv2.net/CurrLib/Forms/AllItems.aspx
Go Links
Home Documents and Lists Create Site Settings Help
School District Main Site
Curriculum Library
Select a View
All Documents
Explorer View
Actions
Alert me
Export to spreadsheet
Modify settings and columns
New Document | Upload Document | New Folder | Filter | Edit in Datasheet
Local intranet

Type	Name	Modified	Modified By	Checked Out To
	AnalyzingExperiments	8/18/2005 12:56 PM	SPSV2\administrator	
	Antarctica	8/18/2005 1:44 PM		
	Bearings	8/18/2005 1:44 PM		
	BioBasisBehavior	8/18/2005 1:45 PM		
	BouncingWaves	8/18/2005 1:45 PM		
	ChemSignsSymbols	8/18/2005 1:45 PM		
	Chromatography	8/18/2005 2:09 PM		
	Cmp_CmpCmplxSen_209	8/18/2005 2:09 PM		
	CmpndCmplxStruc9_12	8/18/2005 2:09 PM		
	Comma_IntroWordsPhrases_503	8/18/2005 2:09 PM		
	ComplementaryEvents	8/18/2005 2:09 PM		
	CompoundVerbs_Subjects_107	8/18/2005 2:09 PM		
	Conics	8/18/2005 2:09 PM		
	CurrentfromMagnets	8/18/2005 2:09 PM		
	DietaryNeeds	8/18/2005 2:09 PM		
	DistanceTimeGraphs	8/18/2005 2:09 PM		
	DivExponents9-12	8/18/2005 2:10 PM		
	DividingFractions6-8	8/18/2005 2:10 PM		

Automated provisioning

At this point you have fully configured one front-end server, defined a school, entered test Classes and users, and validated correct operation. You can now both configure remaining front-end servers, and also enter “real” Class and user data. This is best accomplished via an automated process provided by Class Server using the automated provisioning tool

CSProvision.exe. Microsoft supplies a Class Server Provisioning Guide (in HTML format), which is somewhat detailed and may take some study to gain full understanding and familiarity with all that is possible. The following sections present practical examples of how to use this facility to define servers, schools, Classes and users in an automated, repeatable manner.

There are multiple benefits to using this method. It ensures ease of managing and maintaining school, Class and user definitions in one place. It ensures all front-end servers are configured identically. It avoids errors due to transposition or typos. It can optionally create a log file that can help quickly spot and fix errors.

The process uses the CSProvision.exe file (installed by Class Server), operating on an XML data file which includes appropriate commands and definitions. In general, you will want to use it to define five specific groups of items:

- Server configuration (front-end servers)
- School(s) definition
- Class definitions
- Students and parents
- Teachers

The following describes the process and presents parts of a practical XML file showing how to define each of the above items. This example file was used in the HP performance labs to configure the test systems. The full test file was 1600 lines of XML data (1 school, 20 teachers, 40 Classes, 800+ students), however only representative file fragments will be shown to illustrate syntax and operation.

Rather than a single overly large XML file, such as the test file noted above, you may wish to split it into multiple files.

- You can use a single XML file to contain the server and school(s) definitions. This must be applied to <u>all</u> front-end servers, as when invoked it creates data that is in part stored in the server Windows Registry. This file thus ensures all servers are configured identically and each has knowledge of all schools (school GUIDs, database names and access information) that could be accessed from the web farm servers.
- You could then use a single XML file to define the Classes and users for <u>each</u> school, thus keeping files to a reasonable size. Each such file would contain an appropriate "SignIn" command to the related school, described in detail below, or require a /SignIn switch and appropriate credentials on the CSProvision command line. These school Class/user definition XML files can be invoked from <u>any</u> front-end server, as the relevant data is stored in the school's SQL Server database.

Required data

You will need various data definitions and values, most already used during product installation and configuration, as these must be entered into the XML data file. Prior to creating the XML file(s), make a note of the following:

- School GUID (see above for how to obtain this from the registry)
- Database server name and school database name
- School Region

- File store location
- Authentication method

A full list of the parameter values needed is shown in the *Defining the School* section, below. These values relate to the data you specified when originally defining the school via the Administration applet.

Running CSProvision.exe

Open a command line window and navigate to the folder containing CSProvision.exe. By default this is in the following location:

```
C:\Program Files\Microsoft Class Server\Server\Configure\Tools\CSProvision.exe
```

You will also need to decide on a folder location for the XML definition file and for the log file produced by the operation. Having navigated to the folder above, invoke CSProvision using the command:

```
CSProvision /log <folder>logfile.txt <XML folder>\<provision-filename>.XML
```

The provisioning program will run and execute the instructions in the XML file. A scrolling on-screen log will show progress. Should an error occur, refer to the created log file for details. This will indicate the exact error and the "cell reference" (e.g. "B3" – the XML spreadsheet column/row) of the command or data definition causing the error. Our experience during testing was the most likely cause of error was incorrectly entering data field names – either a simple typo or not exactly following the required 'case'. Most entries are case-sensitive. Use the spelling/case exactly as showing in the provisioning guide.

Some required parameters, such as passwords, are security sensitive. These can be defined using macro replacement in the XML file (e.g. "%AdminPwd%") and then passing the name and value as part of the command line string (e.g. CSProvision /AdminPwd=manager). Pass-through authentication is also possible. Refer to the provisioning guide for more details.

The CSProvision executable and related XML file support a wide range of definition and modification commands, enabling a structured approach to maintaining all these data. You are encouraged to refer to the Class Server Provisioning Guide (CSPG) (HTML file) to discover the full extent of this provisioning tool's capabilities. The guide is written as more of a reference document than a "how-to" guide, but does contain example syntax.

XML file creation

The following sections are offered as practical, tested examples of how to use the provisioning functions and required data definitions to create and populate a school. Fragments of the definition XML file used during HP performance lab testing are included to illustrate relevant commands and data name/value content. Users who are less-familiar with XML-format files may find it easier to create, view and edit the XML file using Microsoft Excel and saving the file as an XML spreadsheet (XMLSS). The Microsoft CSPG document also recommends using this format. The examples below are shown as XMLSS files viewed via Microsoft Excel, rather than showing the usual hierarchical indented XML format.

The example XML file was written in discrete sections that define each portion of the solution (server, school, users, and Classes). Commands to skip sections (!Skip and !EndSkip) can be inserted around parts that do not need to be executed. For example, if you have made changes or additions only to the Classes definition section, then other sections need not be executed. This "section skip" will be seen throughput the examples presented below.

NOTES

The XML file shown was created to provision the HP performance lab systems, but the examples it presents are broadly applicable to real-world needs.

The XML file examples below use a number of MS Excel cell formatting features to improve readability. These are purely cosmetic, but follow similar conventions to those shown in the Class Server Provisioning Guide document.

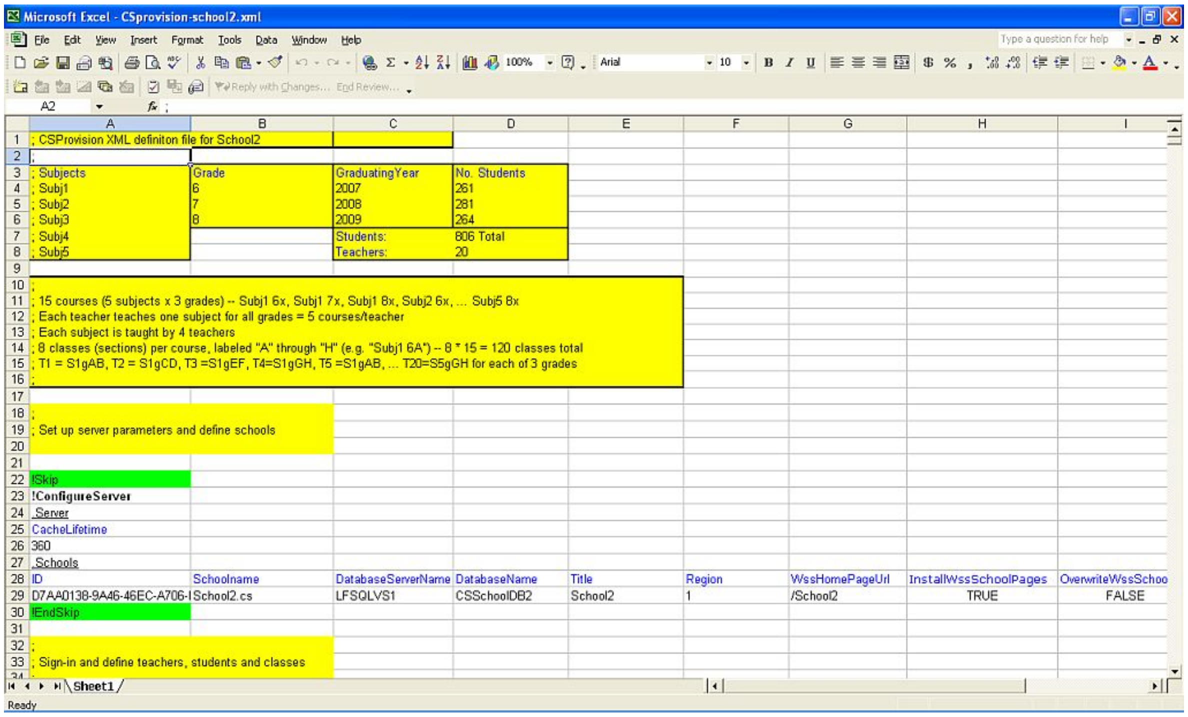
- The !Skip and !EndSkip command cells have a green background to make them stand out (e.g. !Skip and !EndSkip).
- The top-level commands are shown bolded (e.g. **!ConfigureServer**). Such commands all begin with an exclamation point.
- Objects relating to the top-level command all begin with a period character, and are shown underlined (e.g. .Server or .Schools).
- Name/value pairs relating to the object are defined in column/row format. Thus all relevant definition names are across columns in a single row, with related definition values across the columns in subsequent rows. Data names are shown in blue text (e.g. SchoolName and WssHomePageUrl) for clarity. Note that data names contain no spaces and are case-sensitive. They must be entered *exactly* as described (spelling and case) in the CSPG documentation.
- Comments can be included, each preceded by a semicolon character (e.g. ;comment). These can appear on any line, or at the end (final column) of name/value definitions as shown in the examples.

Defining the Server

The purpose of defining the server parameters is to ensure all front-end servers are configured identically and can thus operate in the WLBS front-end cluster in a predictable and consistent manner. Figure 15, below, is the top portion of the example XML file that shows some initial comments (explaining the overall purpose of the file), and the server definition commands. The CSPG lists all available commands and definitions for the server, and the default values for each parameter. If the default is acceptable, it need not be present in the XML file.

In the XML fragment shown in Figure 15, below, rows 23 through 29 define the Server and a School. The Skip/EndSkip commands in rows 22 and 30 should be removed to execute this section. It should only be executed once on each server. If you need to add more schools, then each must first be added by the CSAdmin web page and the school GUID (and other relevant parameters) for each school should be noted for inclusion in the XML file. Then re-execute the edited XML file via CSProvision.exe.

Figure 15. XML file – Server and School definitions



Microsoft Excel - CSprovision-school2.xml

A2 ;

	A	B	C	D	E	F	G	H	I
1	; CSProvision XML definiton file for School2								
2	;								
3	; Subjects	Grade	GraduatingYear	No. Students					
4	; Subj1	6	2007	261					
5	; Subj2	7	2008	281					
6	; Subj3	8	2009	264					
7	; Subj4		Students:	806 Total					
8	; Subj5		Teachers:	20					
9									
10	;								
11	; 15 courses (5 subjects x 3 grades) -- Subj1 6x, Subj1 7x, Subj1 8x, Subj2 6x, ... Subj5 8x								
12	; Each teacher teaches one subject for all grades = 5 courses/teacher								
13	; Each subject is taught by 4 teachers								
14	; 8 classes (sections) per course, labeled "A" through "H" (e.g. "Subj1 6A") -- 8 * 15 = 120 classes total								
15	; T1 = S1gAB, T2 = S1gCD, T3 =S1gEF, T4=S1gGH, T5 =S1gAB, ... T20=S5gGH for each of 3 grades								
16	;								
17									
18	;								
19	; Set up server parameters and define schools								
20									
21									
22	!Skip								
23	**!ConfigureServer**								
24	.Server								
25	CacheLifetime								
26	360								
27	.Schools								
28	ID	Schoolname	DatabaseServerName	DatabaseName	Title	Region	WssHomePageUrl	InstallWssSchoolPages	OverwriteWssSchoo
29	D7AA0138-9A46-46EC-A706-I	School2.cs	LFSQLVS1	CSSchoolDB2	School2	1	/School2	TRUE	FALSE
30	!EndSkip								
31									
32	;								
33	; Sign-in and define teachers, students and classes								

Sheet1

Ready

Refer to rows 23 through 26 in Figure 15, above. These commands and name/value pairs define parameters for the Server. The block begins with the command "**!ConfigureServer**", then the object ".Server", then a single name/value pair to define the server Cache timeout (CacheLifetime) with a value of 360.

Defining the School

The above is followed in row 27 by the schools object ".Schools" and a list of parameter names in row 28. The values for a school (School2 in this example) are in row 29. Subsequent school definitions would be immediately below this (e.g. you would insert rows after row 29 and add further school definitions). The figure was cropped for clarity and does not show all rows in the example school definition. The full set of example parameters are listed below as "name=value" pairs for completeness. See the CSPG document for a list of all possible parameter values and meanings (e.g. USA region is a Region parameter value of 1).

- ID = <GUID> (this is the school GUID string you obtained from the Registry for this specific school). Enter the GUID string exactly as obtained from the Registry.
- Schoolname = School2.cs
- DatabaseServerName = LFSQLVS1 (this was the SQL Server cluster alias name in the HP lab)
- DatabaseName = CSSchoolDB2 (default name was assigned by CSAdmin when school was created)
- Title = School2
- Region = 1 (USA region)

- WssHomePageUrl = /School2
- InstallWssSchoolPages = TRUE (install WSS pages and Web Parts when school is defined on the server)
- OverWriteWssSchoolPages = FALSE
- AdminPassword = managers (the password can be passed as a “%parameter%” – see the CSPG document)
- HomePageEnabled = TRUE
- LRPath = \\class1\csdata (network location of a share to hold the Learning Resource data)
- DefaultBackupDir = \\class1\csdata
- ADmode = 0 (all users validated via sign-in, can be changed to use sign-in, AD, or both as detailed in the CSPG document. Must be 0 if ISA server used – see earlier discussion regarding ISA)

The above server and school definitions must be applied to all front-end servers, and may thus best be contained in a single XML file for this purpose.

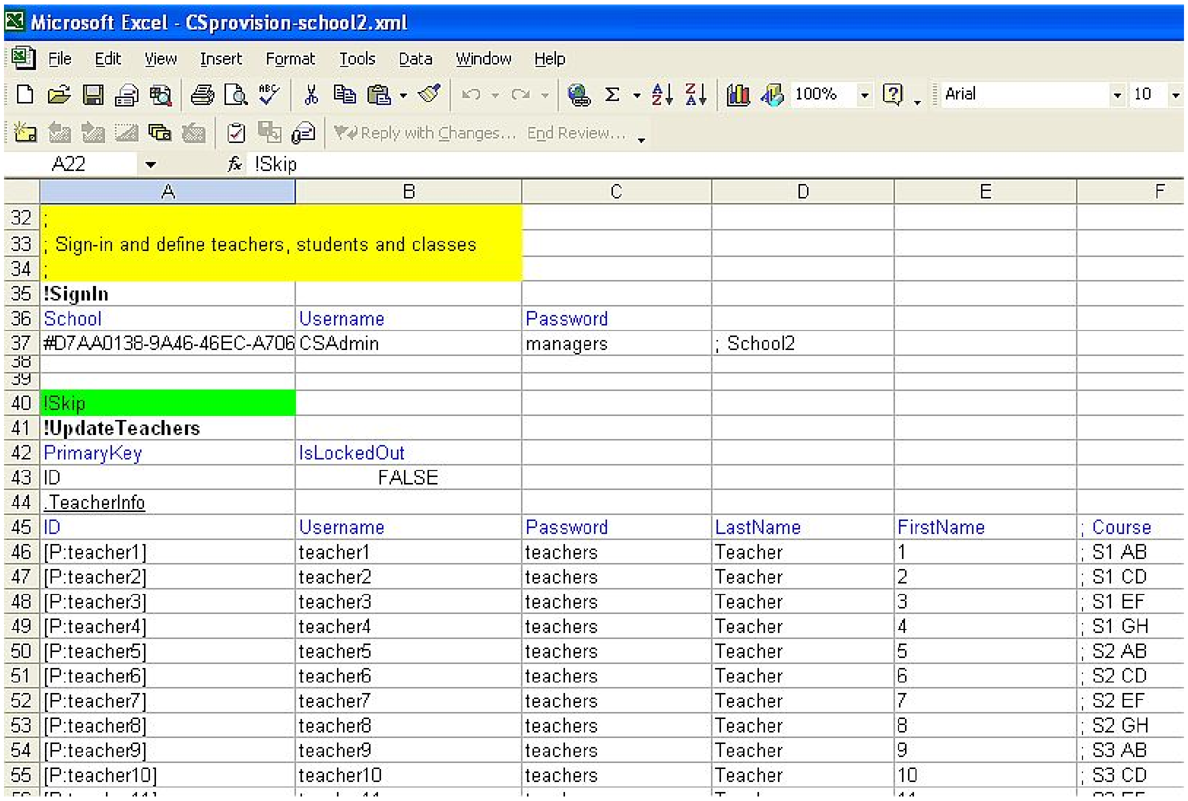
Sign-into a school

The following sections show how to add teachers, students/parents and Classes to a school. As discussed earlier, these could be in a single XML file, or in separate XML files for each school. The examples below are from a single file. As these data are specific to a school, you must first “sign in” to the school as shown in rows 35-37 in Figure 16, below. The **!SignIn** command is followed by a set of name/value parameters for the school, defining the school specifier (via its GUID), CS Administrator username and password. Note that the school specifier is the school GUID *proceeded by a hash sign* (e.g. #<GUID>). This is also equivalent to the School ID formed from the School specifier [Sch:<school-name>]. The comment in column D of the example notes that this GUID referred to School2. Again, the Administrator sign-in password can be passed via a parameter for security.

Defining Teachers

The syntax for adding teachers is shown in Figure 16, below, beginning at row 41. The !Skip at row 40 (and corresponding !EndSkip) should be removed to execute this section.

Figure 16. XML file – Sign-In and Teacher definitions


Microsoft Excel - CSprovision-school2.xml
File Edit View Insert Format Tools Data Window Help
100%
Arial
10
Reply with Changes... End Review...

A22 fx !Skip

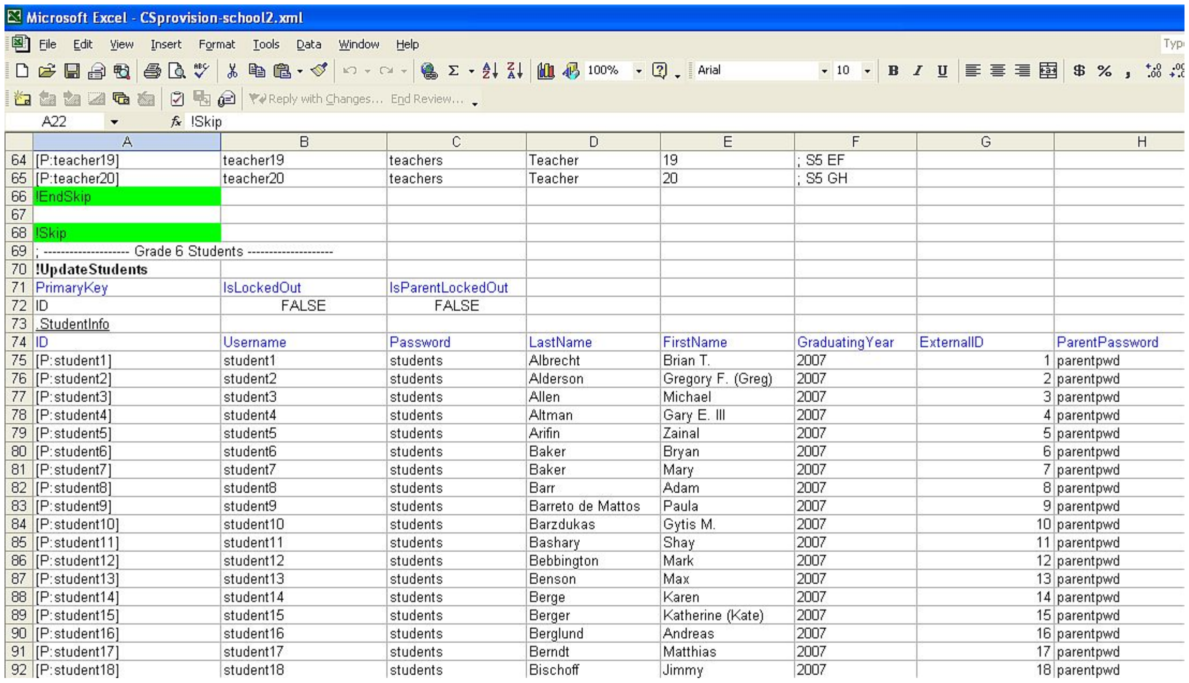
	A	B	C	D	E	F
32	;					
33	; Sign-in and define teachers, students and classes					
34	;					
35	!SignIn					
36	School	Username	Password			
37	#D7AA0138-9A46-46EC-A706	CSAdmin	managers	; School2		
38						
39						
40	!Skip					
41	!UpdateTeachers					
42	PrimaryKey	IsLockedOut				
43	ID	FALSE				
44	.TeacherInfo					
45	ID	Username	Password	LastName	FirstName	; Course
46	[P:teacher1]	teacher1	teachers	Teacher	1	; S1 AB
47	[P:teacher2]	teacher2	teachers	Teacher	2	; S1 CD
48	[P:teacher3]	teacher3	teachers	Teacher	3	; S1 EF
49	[P:teacher4]	teacher4	teachers	Teacher	4	; S1 GH
50	[P:teacher5]	teacher5	teachers	Teacher	5	; S2 AB
51	[P:teacher6]	teacher6	teachers	Teacher	6	; S2 CD
52	[P:teacher7]	teacher7	teachers	Teacher	7	; S2 EF
53	[P:teacher8]	teacher8	teachers	Teacher	8	; S2 GH
54	[P:teacher9]	teacher9	teachers	Teacher	9	; S3 AB
55	[P:teacher10]	teacher10	teachers	Teacher	10	; S3 CD

The **!UpdateTeachers** command and related data is followed by the .TeacherInfo object. Actual teacher name/value data begins on row 45. The illustrated Username, Password, LastName and FirstName were used for test scripting simplicity in the HP labs; and real names and passwords would appear here. The Username, however, can be any unique string. Note that the ID specifier is formed from this Username (i.e. [P:<username>]) and must be defined in this manner, as it is a Class Server ID specifier. ID specifiers are required anywhere an ID is needed (PersonID, ClsID, etc.) but not where data objects are needed (Person, Cls, etc.) Again, comments can be added as shown in column F. As with defining each section of data, you can subsequently re-execute a section is you have made modifications or additions.

Defining Students and Parents

Defining students and parents is performed in a similar way to teachers, except with some additional parameters. Figure 17, below, is a fragment of the example XML file showing the conclusion of defining teachers (up to row 65) followed by the beginning of student and parent definitions beginning on row 69. Again, the Username, Password, ExternalID and ParentPassword fields shown were used for test purposes. The same rules for Username and ID apply, as was explained above regarding teachers.

Figure 17. XML file – Students and Parents


Microsoft Excel - CSprovision-school2.xml
File Edit View Insert Format Tools Data Window Help
100%
Arial
10
Reply with Changes... End Review...

A22 | !Skip

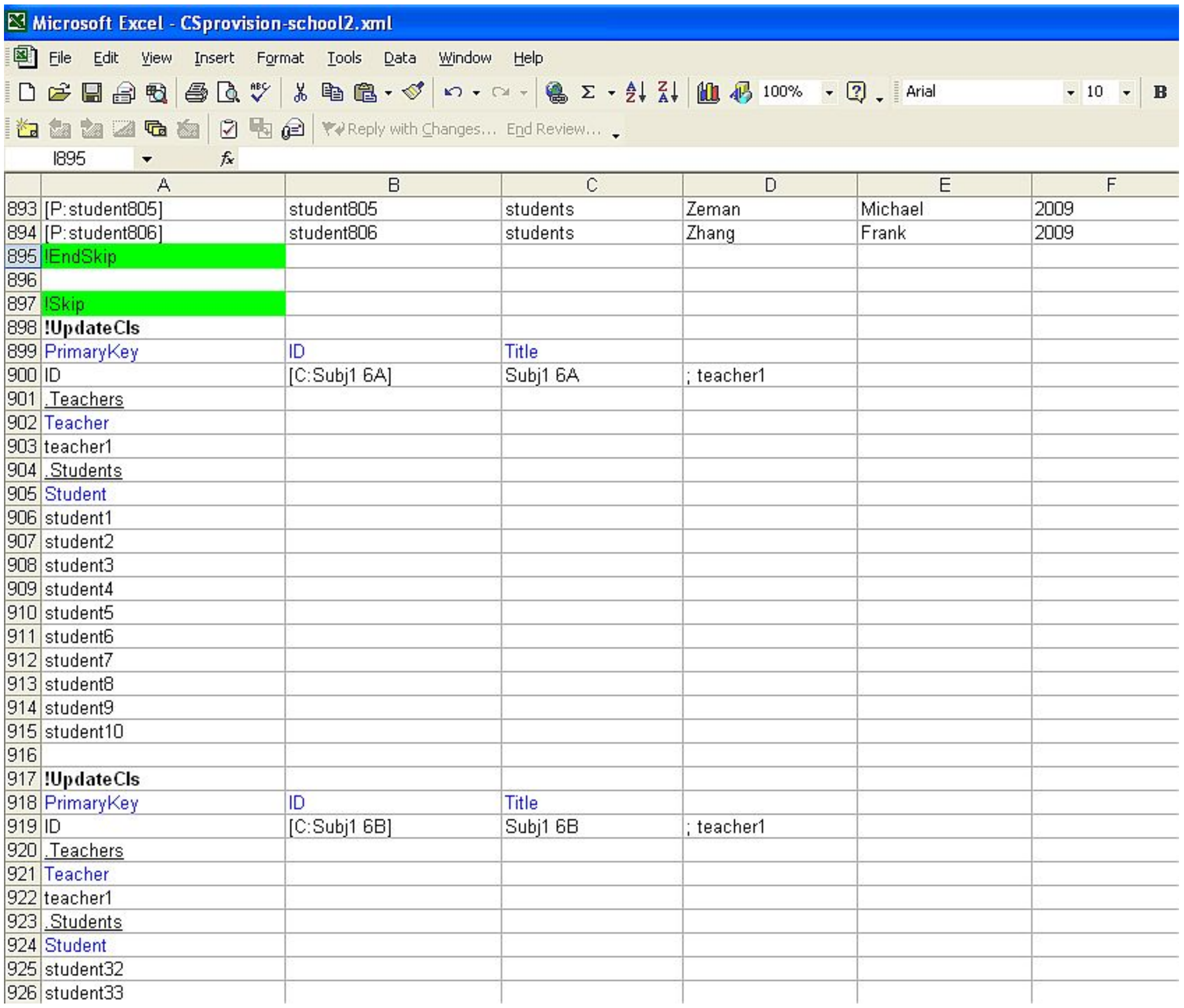
	A	B	C	D	E	F	G	H
64	[P:teacher19]	teacher19	teachers	Teacher	19	; S5 EF		
65	[P:teacher20]	teacher20	teachers	Teacher	20	; S5 GH		
66	!EndSkip							
67								
68	!Skip							
69	; ------------------- Grade 6 Students -------------------							
70	**!UpdateStudents**							
71	PrimaryKey	IsLockedOut	IsParentLockedOut					
72	ID	FALSE	FALSE					
73	.StudentInfo							
74	ID	Username	Password	LastName	FirstName	GraduatingYear	ExternalID	ParentPassword
75	[P:student1]	student1	students	Albrecht	Brian T.	2007	1	parentpwd
76	[P:student2]	student2	students	Alderson	Gregory F. (Greg)	2007	2	parentpwd
77	[P:student3]	student3	students	Allen	Michael	2007	3	parentpwd
78	[P:student4]	student4	students	Altman	Gary E. III	2007	4	parentpwd
79	[P:student5]	student5	students	Arifin	Zainal	2007	5	parentpwd
80	[P:student6]	student6	students	Baker	Bryan	2007	6	parentpwd
81	[P:student7]	student7	students	Baker	Mary	2007	7	parentpwd
82	[P:student8]	student8	students	Barr	Adam	2007	8	parentpwd
83	[P:student9]	student9	students	Barreto de Mattos	Paula	2007	9	parentpwd
84	[P:student10]	student10	students	Barzdukas	Gytis M.	2007	10	parentpwd
85	[P:student11]	student11	students	Bashary	Shay	2007	11	parentpwd
86	[P:student12]	student12	students	Bebbington	Mark	2007	12	parentpwd
87	[P:student13]	student13	students	Benson	Max	2007	13	parentpwd
88	[P:student14]	student14	students	Berge	Karen	2007	14	parentpwd
89	[P:student15]	student15	students	Berger	Katherine (Kate)	2007	15	parentpwd
90	[P:student16]	student16	students	Berglund	Andreas	2007	16	parentpwd
91	[P:student17]	student17	students	Berndt	Matthias	2007	17	parentpwd
92	[P:student18]	student18	students	Bischoff	Jimmy	2007	18	parentpwd

If a parent password is specified, then a student's parent can access the school site by singing-in with a username of "!<username>" (i.e. precede the student's user name with a "!" character), and using the defined parent (not student) password.

Defining Classes

A Class definition (beginning with **!UpdateCls**) comprises data relating to the Class (ID and Title), which is then followed by the teacher(s) responsible for that Class and a list of students who are members of the Class. The example XML file fragment in Figure 18, below, illustrates the conclusion of defining all the school's students (up to row 894) followed by the first Class definition starting on row 898. The Class titles shown were again for test purposes, however note that the Class ID is formed from the Class ID specifier using the Class Title (i.e. [C:<Title>]) and must be specified in that manner. Teacher definition is as shown in rows 901-903, with student definitions as shown in rows 904-915. Row 917 onward shows a second Class definition.

Figure 18. XML file – Defining Classes


Microsoft Excel - CSprovision-school2.xml
File Edit View Insert Format Tools Data Window Help
100%
Arial
10
Reply with Changes... End Review...
I895

	A	B	C	D	E	F
893	[P:student805]	student805	students	Zeman	Michael	2009
894	[P:student806]	student806	students	Zhang	Frank	2009
895	!EndSkip					
896						
897	!Skip					
898	!UpdateCls					
899	PrimaryKey	ID	Title			
900	ID	[C:Subj1 6A]	Subj1 6A	; teacher1		
901	.Teachers					
902	Teacher					
903	teacher1					
904	.Students					
905	Student					
906	student1					
907	student2					
908	student3					
909	student4					
910	student5					
911	student6					
912	student7					
913	student8					
914	student9					
915	student10					
916						
917	!UpdateCls					
918	PrimaryKey	ID	Title			
919	ID	[C:Subj1 6B]	Subj1 6B	; teacher1		
920	.Teachers					
921	Teacher					
922	teacher1					
923	.Students					
924	Student					
925	student32					
926	student33					

The above sections illustrate how schools and their related Classes and users can be easily defined in centrally-managed XML files, and be applied to the MLG solution front-end servers as needed. Note that only the Server and School definitions need be applied to every front-end server, as these actions result in various data being stored in the server's Windows Registry.

Adding or modifying school Classes or school users can be performed on any front-end server, as these actions result in data being written to the school SQL database.

The CSProvision.exe program, along with XML file commands and data, can perform a range of other administration functions. Refer to the CSPG document for further details.

Performance

The intent of the HP lab tests of the MLG solution was to determine and test best-practice deployment methodology, to determine performance, and to devise recommended solution sizing and configuration information utilizing HP ProLiant and BladeSystem servers. While it was not possible to test to very large user populations (e.g. millions of students), the results obtained demonstrated the expected throughput capacity of front-end servers, and of typical scalable web farm deployments.

In a USA School District scenario, it appears that a single web farm deployment will support most typical user populations. Up to 10 front-end servers were successfully lab tested in a large web farm and displayed excellent scalability and performance. As many as 15-20 such front-end servers could be supported by a single active SQL server, depending on the workload mix.

In an EMEA scenario, it would be typical to see a higher user population spread across a larger geographic area (country). Such scenarios could be supported by a single central administration configuration and multiple remote "satellite" configurations (hub and spoke model). Each such satellite would be sized to support the area's local schools and student/parent/teacher population.

Test results clearly showed that the MLG solution capitalizes on the WSS/SPS scalable architecture, and that the well-understood rules and guidelines for deploying WSS/SPS are applicable to MLG. However, given the higher resource consumption of some of the MLG custom Web Parts care must be taken to determine the intended workload as accurately as possible, so as to size the solution accurately. The later section covering Sizing and Configuration speaks to this topic further.

The following sections describe the HP performance lab testing, the performance results obtained and some analysis discussion.

Test methodology

HP has a significant historical investment in the performance characterization of SharePoint based solutions and has leveraged that to develop test scenarios for the MLG solution. HP ProLiant and BladeSystem servers are installed and configured normally, and are deployed in different configurations to determine performance and best practices. Emulated user scenarios are developed to test function performance and to investigate prototypical real world business and role-based user activities.

The workload user scenarios are created and applied using high-end workload emulation software (Segue SilkPerformer), which is an ideal tool suite to evaluate web-based solutions. The workload 'scripts' are based on recordings of actual user activity, and are then enhanced to provide such features as list item selection, randomization, logical flow, etc. The result is an emulated role-based user scenario that mimics real user activity. The scripts are applied using dedicated emulation servers. These act like client desktops, in that they interact with the front-end servers as in real life by transmitting the appropriate HTTP/HTML traffic and responding to the received content.

Accurate throughput rates (e.g. web pages per second) and per-function response times are also obtained. Additionally, a monitoring server is used to capture, collate and analyze the operating system monitor data for all servers in the tested solution. This allows correlation of user load, response times, throughput, and per-server resource utilization. These data are also used to create sizing models and tools.

The next section describes the workload scenarios employed for the emulated teachers, students and parents using the MLG solution.

Workload scenarios

It is un-necessary, and impractical, to create a workload scenario that employs every possible function that a user might perform. To create a reasonably representative workload scenario script, you should focus on solution functions that meet the following three criteria:

- They are important to the "business"
- They are performed reasonably frequently
- They are known, or believed, to have an impact on the servers

Functions meeting the above criteria will likely have an impact on the solution server(s) performance and thus on sizing. Some of the MLG functions are basic WSS or SPS functions (e.g. browse/open documents, discussions, etc.). The characteristics and performance of these are already well understood from prior performance characterization work. The workloads used, therefore, tended to focus on the new, customized functions (Web Parts) that provide the unique MLG functionality to support an Education-related portal. The following describes the role-based user scenarios and functions employed in the test workloads.

All users

All user roles (teachers, students and parents) performed some percentage of common (more simple) functions as part of their workload scenario, these being:

- Accessing a Class web site via the "My Classes" Web Part. This MLG custom Web Part lists Class sites of which the specific user is a member.
- Class site activity (browsing, reading, etc.) as per a typical WSS team site
- Participating in one or more threaded discussions (read and/or contribute)

The above functions are mostly benign and not overly resource consumptive although, as discussed in detail earlier, the My Classes Web Part is more complex and consumes more CPU than a simple Hyperlink List Web Part.

Teachers

In addition to the above, teacher activity also included functions related to Class assignments:

- Browse Curriculum Library and open/read learning resources (LRs), representing lesson preparation.
- Assign an LR to a Class (all students in Class). This involves selecting an LR from the Curriculum Library and then navigating multiple screens and making appropriate selections to invoke the assignment.
- Grade completed student assignments. While it is possible for a teacher to simply "return all" student assignments (auto grading), the workload instead emulated the teacher viewing/grading individual students' papers. While possibly more resource-intensive, it

was felt that in real life a teacher would gain more information as to the Class students' grasp of material and also spot common errors in test answers using this procedure. The typical logical flow used was:

1. Pick the first assignment from the "to be graded" list
2. Randomly choose a number of students' papers to grade (nominally 5-10)
3. Loop around, grading and returning each student's paper

This therefore emulated a "grading session" lasting a period of time.

Students

In addition to the common (all users) functions, students also performed assignment-related functions:

- Perform an assignment. This involved picking the first item from the "Due" list, opening it (invoking the LRM viewer), reading though the LRM material, and finally clicking to submit the completed assignment for grading.
- Viewing graded assignments. This comprised picking a random assignment from the "Graded" list, and opening it (via LRM viewer) to emulate viewing the grade and teacher's comments.

Parents

In addition to the common functions, parents also performed one assignment-related activity – this emulating them taking an interest in the child's progress.

- Pick a random assignment from the entire list (Due, Submitted or Graded) and open/read it via the LRM viewer.

The above workload scenarios were felt to be reasonably typical of user activity, and employed the key functions with respect to the solution goals and determining likely solution performance.

Results

The results measured, shown in Table 1 below, relate to an HP ProLiant BL20p 2-CPU BladeSystem server using 3.2GHz CPUs, running at a nominal average %CPU busy of 80%. While it is possible to load the server above this level, experience has shown that an average of 80% busy provides optimal response times and consistent performance while being cost-effective. Specific server role configurations (CPU, memory, disks, etc.) are discussed in detail in the HP authored *SharePoint Portal Server Performance* white paper noted in the section *For More Information*. The configurations HP recommends in that paper are also applicable to servers used in this MLG solution.

Simple functions yielded average response times that were generally sub-second, as has previously been measured for typical WSS/SPS functions. The simpler MLG functions took only a few seconds on average. Some of the more complex MLG functions (e.g. large sorted lists or assignment functions) took up to 5 seconds or so, depending on the quantity and type of the data presented, and the Web Part complexity (resource usage).

One of the key sizing metrics is to determine the throughput rate possible (at 80% average CPU busy) for each key workload scenario function. Table 1, below, presents summarized findings for the achievable per-function throughput rates in terms of web pages/second; and the related SQL Server %CPU busy and network traffic rates.

Table 1. Per-function performance data

User and Function	Throughput (pages/sec)	Weighting Factor	SQL Server %CPU	Client Net KB/sec	Server Net KB/sec
All – visit Class site	12	8.33	4.00	400	1000
All – Class site functions	100	1.00	13.00	200	400
All – View/add discussions	60	1.67	5.00	2300	1300
Teacher – Create assignment	6	16.67	3.50	350	2400
Teacher – Browse CurrLib	15	6.67	9.50	1600	3050
Teacher – Grade assignments	6	6.67	6.00	250	1500
Student – Perform assignment	10	10.00	3.00	300	650
Student – View returned assignment	5	20.00	3.50	150	800
Parent – View assignments	6	16.67	3.50	200	800

Analysis

The "Weighting Factor" column shown in Table 1 (above) represents the relative "intensity" of a function compared to the least intense (highest throughput) function. Thus, for example, the resource (CPU) cost of a student performing an assignment is about 10 times that of a typical simple WSS web site function (e.g. browse document list).

The SQL Server %CPU column represents the average %CPU busy for the SQL Server when supporting one (1) front-end server running at 80% CPU busy. Depending on the actual workload mix, a single active SQL Server can therefore support maybe 10-15 front-end servers. This is a higher number than has been measured for typical WSS team site deployments (nominally 6 front-end servers per SQL Server). The MLG front-end servers are more CPU-intense compared to basic WSS team site front-end servers; thus the related SQL Server CPU consumption per unit of work (transaction throughput) is lower.

The Client Net and Server Net columns show the client-to-front-end and front-end-to-SQL network traffic rates respectively, in Kilobytes per second (KB/sec). While some MLG Web Parts return a lot of data to the clients (e.g. assignment lists, CurrLib content), the function elapsed times are largely predicted on CPU consumption. Thus the network traffic per unit time is not as high as might be expected as the overall function takes longer.

As was discussed in an earlier section, the assignment workflow functions are quite resource (CPU) intense compared to other functions. However, the incidence (and thus total frequency) of these occurring per unit time is likely quite low. For example, each teacher may only create a few Class assignments per day.

Sizing solutions

Knowing the achievable per-function throughput per server, the maximum throughput (highest throughput function capacity) and the relative weighting, we can combine mixes of desired function rates arithmetically to determine expected server loading; and the number of such servers required to support a specific workload. The following presents a simple methodology to perform these calculations and arrive at an expected configuration solution.

Sizing methodology

We must first define the expected solution workload in terms of the number of users of each type. We can define users by absolute number, or derive the number of each user type by understanding expected ratios. Such ratios might typically include:

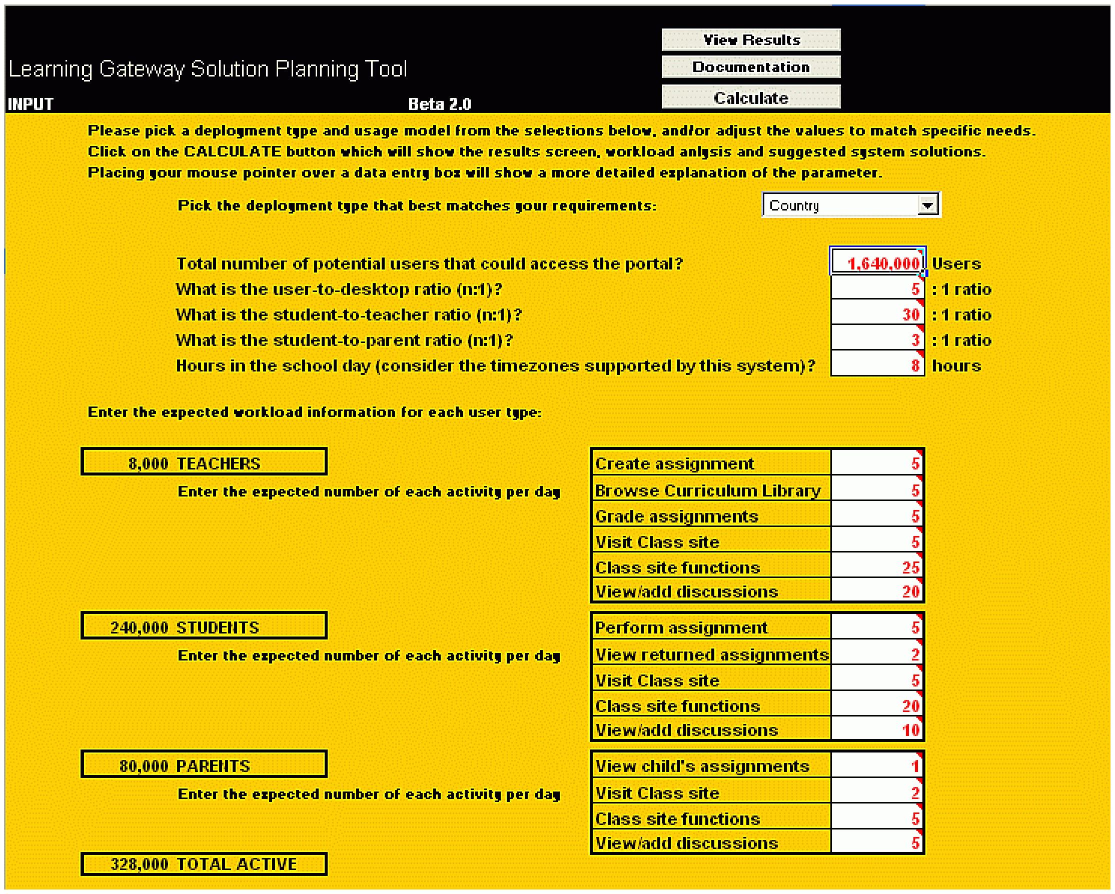
- User-to-desktop ratio (determines active user population)
- Student-to-teacher ratio (implies typical Class size)
- Student-to-parent ratio (typical family size)

We also need to know the expected frequency (how many per day) that user-related functions will be performed. Finally, we will need to know other parameters, possibly country or geography related, such as hours in a school day. From the above discussion, we also know (empirically) the weighting factor (relative intensity) of the user functions, so we can normalize to a common function and calculate "equivalent load".

Sizing data input

Figure 19, below, shows a screenshot of a prototype (Microsoft Excel) sizing sheet that is being used to develop and validate MLG sizing algorithms and guidelines.

Figure 19. Sizing example – workload input



Learning Gateway Solution Planning Tool

INPUT

Beta 2.0

Please pick a deployment type and usage model from the selections below, and/or adjust the values to match specific needs.
Click on the CALCULATE button which will show the results screen, workload anlysis and suggested system solutions.
Placing your mouse pointer over a data entry box will show a more detailed explanation of the parameter.

Pick the deployment type that best matches your requirements: Country

Total number of potential users that could access the portal?	1,640,000	Users
What is the user-to-desktop ratio (n:1)?	5	: 1 ratio
What is the student-to-teacher ratio (n:1)?	30	: 1 ratio
What is the student-to-parent ratio (n:1)?	3	: 1 ratio
Hours in the school day (consider the timezones supported by this system)?	8	hours

Enter the expected workload information for each user type:

8,000 TEACHERS Enter the expected number of each activity per day	Create assignment	5
	Browse Curriculum Library	5
	Grade assignments	5
	Visit Class site	5
	Class site functions	25
	View/add discussions	20
240,000 STUDENTS Enter the expected number of each activity per day	Perform assignment	5
	View returned assignments	2
	Visit Class site	5
	Class site functions	20
	View/add discussions	10
80,000 PARENTS Enter the expected number of each activity per day	View child's assignments	1
	Visit Class site	2
	Class site functions	5
	View/add discussions	5
328,000 TOTAL ACTIVE		

The screen illustrates fields soliciting the required sizing input. The values shown were part of sizing a 2,500,000 user solution for an EMEA country Ministry of Education, including 1,200,000 students.

Such a tool also allows performing "what if" analyses. For example, varying the number of assignment-related functions per day will show the impact on the total number of servers required. Similarly, changing (increasing) the number of users can show how a solution will need to grow over time.

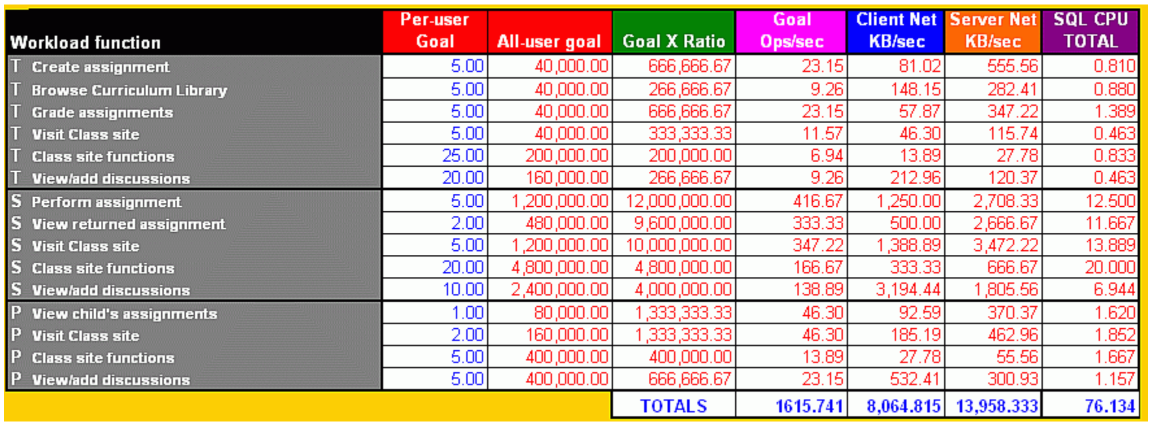
Sizing calculation

Figure 20, below, shows the calculation worksheet. Its operation is as follows:

- The Per-user Goal column simply repeats the desired throughput (operations/day) for each user function, obtained from the input worksheet.
- The All-user goal column multiplies the above by the number of active users of each type.
- The Goal X Ratio column multiplies the above by the relevant function weighing value, thus normalizing the desired throughput to the least-intense function.
- The Goal Ops/sec column converts the goal in operations per school day into operations per second.

- The Client Net, Server Net and SQL CPU columns are calculated relative to the desired goal, based on known value relationships from the above results table.

Figure 20. Sizing example – required throughput calculation



	Workload function	Per-user Goal	All-user goal	Goal X Ratio	Goal Ops/sec	Client Net KB/sec	Server Net KB/sec	SQL CPU TOTAL
T	Create assignment	5.00	40,000.00	666,666.67	23.15	81.02	555.56	0.810
T	Browse Curriculum Library	5.00	40,000.00	266,666.67	9.26	148.15	282.41	0.880
T	Grade assignments	5.00	40,000.00	666,666.67	23.15	57.87	347.22	1.389
T	Visit Class site	5.00	40,000.00	333,333.33	11.57	46.30	115.74	0.463
T	Class site functions	25.00	200,000.00	200,000.00	6.94	13.89	27.78	0.833
T	View/add discussions	20.00	160,000.00	266,666.67	9.26	212.96	120.37	0.463
S	Perform assignment	5.00	1,200,000.00	12,000,000.00	416.67	1,250.00	2,708.33	12.500
S	View returned assignment	2.00	480,000.00	9,600,000.00	333.33	500.00	2,666.67	11.667
S	Visit Class site	5.00	1,200,000.00	10,000,000.00	347.22	1,388.89	3,472.22	13.889
S	Class site functions	20.00	4,800,000.00	4,800,000.00	166.67	333.33	666.67	20.000
S	View/add discussions	10.00	2,400,000.00	4,000,000.00	138.89	3,194.44	1,805.56	6.944
P	View child's assignments	1.00	80,000.00	1,333,333.33	46.30	92.59	370.37	1.620
P	Visit Class site	2.00	160,000.00	1,333,333.33	46.30	185.19	462.96	1.852
P	Class site functions	5.00	400,000.00	400,000.00	13.89	27.78	55.56	1.667
P	View/add discussions	5.00	400,000.00	666,666.67	23.15	532.41	300.93	1.157
				TOTALS	1615.741	8,064.815	13,958.333	76.134

The summed value at the bottom of the Goal Ops/sec column thus reflects the total throughput required, in normalized terms of the least intense function. From empirical results, we can accomplish 100 such function operations/sec per front-end server. Thus, in this example, approximately 16 (1615.741/100) front-end servers are required to support the teacher/student/parent workload for this large country-wide solution. The SQL CPU column summation shows the expected %CPU busy for SQL Server. While this indicates in the above example that a single SQL Server might suffice, a practical deployment of this size will likely be different. This is discussed in the following *Examples* section.

The above Microsoft Excel sizing worksheet is being used to validate algorithms, guidelines and deployment best practices. HP intends to produce an executable MLG solution sizing tool, very similar to the SharePoint 2003 sizing tool currently available from HP ActiveAnswers.

Sizing examples

The MLG solution is designed with high scalability, thus it can be deployed to countries having different educational infrastructures.

School Districts

In the USA, schools are grouped into School Districts within each State. Such Districts tend therefore to be more modest in size, although a few large Districts do exist (e.g. New York State and Miami-Dade). Solutions for these modest sized Districts can therefore be supported by a single deployment comprising sufficient front-end servers to meet the workload needs. As an example, a specific School District in the San Francisco area has the following characteristics:

- ~ 16,000 students, 800 teachers, 600 administrators
- 22 school sites

- ~ 2,500 active students (~ 6:1 active ratio), ~ 1,200 active staff (~ 2:1 ratio)
- ~ 400 Class and other team sites

For these characteristics, a deployment including at least two front-end servers and an active/passive SQL cluster (both for HA needs) will be adequate for WSS and Class Server functionality. Additional servers could be deployed to support SPS sites and optionally SPS Search and Indexing needs. Growth is accomplished simply by adding extra front-end servers.

Country deployments

In contrast, in EMEA a solution will usually be deployed for a country's entire school system under the Ministry of Education. This involves a much higher number of schools and users, but will be spread geographically. An example of this was shown in the Sizing example above, the requirements for the total country school system including:

- ~ 2,500,000 users in total, including 800,000 centralized administrators
- 1,500 school sites in 8 geographic areas (1,200,000 students, 500,000 parents)
- ~ 10:1 user:desktop ratio

A proposed solution for these needs was a hub-and-spoke model, comprising a central system deployment to support the administration functions and 8 satellite deployments for each geographical area. The total to support all required functions and locations was over 50 servers.

Summary

The MLG solution capitalizes well on the highly scalable SharePoint foundation, and provides a robust Learning Management feature set by leveraging Class Server v4 functionality and WSS custom Web Parts. Optionally, SharePoint can enhance the overall solution by providing District-level portal sites and shared services – including content aggregation, search, notification and team collaboration. Installation is quick and simple, and provisioning of schools, Classes and users can be performed via automated procedures.

Some assignment workflow related functions will consume higher levels of server resources than the more simple WSS Web Parts. However, the incidence of use of these functions will likely be markedly less than the common portal and web site functions. Prescriptive configurations for SharePoint are also applicable for the MLG solution, as are the deployment guidelines and best practices detailed in related documents. Performance characterization again confirmed that SharePoint-base solutions such as the MLG that employ a scale-out server architecture are ideally suited to HP ProLiant or BladeSystem servers, providing excellent price/performance and high availability.

For more information

HP recommends the following as sources for more detailed information regarding the MLG solution and SharePoint technologies.

Related HP documents and information sources:

- SharePoint Portal Server 2003 Performance white paper: http://h71019.www7.hp.com/ActiveAnswers/cache/70672-0-0-0-121.html
- SharePoint Portal Server 2003 on HP BladeSystem servers white paper: http://h71019.www7.hp.com/ActiveAnswers/cache/106558-0-0-0-121.html
- SharePoint Portal Server 2003 sizing and configuration tool: http://h71019.www7.hp.com/ActiveAnswers/cache/79168-0-0-0-121.html
- HP ActiveAnswers for SharePoint Products & Technologies: http://www.hp.com/solutions/activeanswers/sharepoint
- HP Solutions for SharePoint Products & Technologies: http://www.hp.com/solutions/microsoft/sharepoint
- HP Solutions for Microsoft Office System: http://www.hp.com/solutions/microsoft/officesystem

Related Microsoft documents and information sources:

- The Microsoft Learning Gateway solution: http://www.microsoft.com/education/learninggateway.mspx
- Microsoft Class Server: http://www.microsoft.com/education/classserver.mspx
- The Solution Accelerator for Intranets: http://www.microsoft.com/downloads/details.aspx?FamilyID=7cdc1f2d-f550-49e0-9b74-318da11ba1b4&DisplayLang=en
- The SharePoint Portal Server online documentation: http://www.microsoft.com/sharepoint/server/techinfo/productdoc/default.asp
- The SharePoint Portal Server product overview: http://office.microsoft.com/en-us/FX010909721033.aspx
- The SharePoint Portal Server resource kit: http://www.microsoft.com/sharepoint/server/techinfo/reskit/default.asp

2/2006 -1