Improving Capstone Courses with Content Management Systems and Virtualization

Alex Radermacher, Adam Helsene, and Dean Knudson
North Dakota State University

This paper discusses implementing content management systems and virtual servers in a university capstone course and how these tools can improve a capstone course. In our capstone course, students work on real-world projects sponsored by local and regional businesses. Student teams follow a well-defined process which specifies the actions they must perform (e.g. requirements gathering, software design, etc.) and the work products they must create (e.g. software test plan, weekly reports, project schedules, etc.). This creates a need for a content management system and project management tools which store the work products produced by the teams and allow team members to easily collaborate on the project. During the last two years, we have worked to migrate from using existing solutions (Blackboard E-portfolios, TWiki sites, and other tools.) to using Subversion, Trac, and VMware ESXi. This paper discusses our experiences implementing these applications and the benefits that using these applications provide over other solutions which we have used previously.

Corresponding Author: Alex Radermacher, alex.radermacher@ndsu.edu

Introduction

At North Dakota State University (NDSU) we have been performing real-world projects as part of our capstone course since 2005. During that time, we have worked with many local and regional companies such as IBM, Microsoft, ATK, 3M, and Thomson Reuters and have completed over 70 projects. Over time, we have worked to implement a Capability Maturity Model Integration (CCMI) Maturity Level 2 process. A more exhaustive description of our process can be found in our previous work[1]. In order to be considered operating at Maturity Level 2, the Software Engineering Institute (SEI) requires that a content management system be used to store documents, code, and baseline releases[2]. We regarded our previous approaches[3] as an unsatisfactory solution to Maturity Level 2 requirements. Even if a capstone course does not need to follow a specific process, using content management systems, project management tools, and virtualization is a good way to improve the quality of the capstone course.

Over the past two years we have used Subversion (SVN), a content management system, Trac, a project management bug-tracking tool, and virtualization for our capstone projects. This has allowed us to comply with CMMI Maturity Level 2 requirements, but has other benefits as well. For example, it allows us to expose students to tools that they can expect to use once they enter industry. Implementing these solutions also improves the ability for students and their sponsors (An experienced software developer or manager from an industry company who works with a student team.) to collaborate and to share information more easily.

The following sections provide a detailed description of our current solution and the advantages when compared to our previous methods. We also describe the hardware used and our experiences with installing and maintaining the software. A future work section describes our plans to further improve our implementation.

Content Management Using Subversion

Subversion is an open source content management systems released under the Apache license. Student teams use SVN to store and manage the code for their projects. CMMI guidelines require that content management systems be used to store work products, but the use of SVN also improves the ability of teams to share code and work together. Prior to implementing Subversion for code management, project teams were responsible for implementing their own content management system or using other methods for collaborating. The usual results of this approach were that students would not store their code in a content management system at all.

Each team is provided with a separate repository which is initially set up by the course instructors. A default structure is provided, but students are free to implement their own system. Students receive training on the basic functionalities of SVN (e.g. checking code out of a repository, locking files, and committing changes back to the repository.) and an SVN client is installed on all machines in the university computer labs. We use TortoiseSVN due to its ease of use, but students are free to select their own SVN client.

Subversion was chosen over other alternatives mostly due to cost factors, both in terms of initial cost and support cost. The software can be installed and administered effectively without requiring advanced Linux knowledge and there is a large amount of documentation and tutorials available online. SVN is also designed for managing code, which makes it ideal for use in computer science courses. Other universities have also found that SVN is a useful tool for computer science courses[4,5].

When we first began using SVN we used a spare Pentium 4 machine with 512 MB of RAM running CentOS 5.3. This machine had sufficient resources to provide an adequate level of usability, but would begin to run slow under heavy load. Basic access authentication and SVN path-based authorization were used to provide secure sites with limited access to non-team members. This required creating unique user names and generating passwords for every person who needed access to a repository. The authorization configuration files were also configured manually. Although this was time consuming, it was possible to set up without advanced Linux knowledge.

This year we have set up a virtual machine (More information regarding the hardware and software specifications of the virtual server can be found in the Virtualization section of this paper.) dedicated for use by Subversion and Trac servers, with one server instance (Fedora 11 is used as the operating system.) shared by all teams. Apache is used to serve the repository's contents via HTTPS. We use Apache as it allows for secure connections and allows us authenticate users at Apache's level rather than relying on SVN's methods, which are not as secure or convenient. Apache authentication is done against a local LDAP server, allowing us to manage users and groups at a higher level than using a text file on the server, which is the default for Subversion. This additionally permits us to avoid dealing with student password management. We have created a Perl script which initializes repositories and establishes authentication information for each SVN instance.

Project Management Using Trac

Trac is an open source, web-based project management tool released under a modified BSD license. Trac interfaces with Subversion to provide browser-based access to a repository and issue tracking features. Student teams primarily use Trac to track and fix software defects, store work products (e.g. design documents, test plans, etc.), and facilitate communication between project stakeholders.

Previously, we used Blackboard E-portfolios and TWiki sites to provide support for project management tasks, but we felt these solutions were inadequate. For example, Blackboard E-portfolios had several limitations: the inability to store documents in a tree structure, only allowing for one team member to upload documents, and limited storage capacity for files. TWiki sites address many of the shortcomings of Blackboard E-portfolios, but lack many of the features of Trac. TWiki does not provide web access to an SVN repository to easily view changes between versions of code and does not contain a built-in ticket system like Trac does.

Trac must be installed on the same server as Subversion as it not possible to use Trac with a remote SVN repository. Currently, Apache is used as it provides a convenient front end for Trac and allows us to use a standard authentication mechanism, LDAP. The virtual machine which runs our Trac and Subversion instances has been allocated 2 GB of RAM which has improved performance compared with our previous server.

Virtualization Using VMware ESXi

Virtualization is a computing technique used to allow for multiple virtual computers to run on a single physical computer. A single computer can 'host' many virtual computers. Virtualization is commonly used to allow for many servers to take up a small amount of space as well as share a pool of computing resources. This allows for more efficient usage of computing resources, especially for low-use and low-power virtual servers. We use virtualization to efficiently use resources and allow us to provide individual computing environments for each team.

One of the major advantages of using virtualization is the ability to quickly deploy virtual machines as either development or testing environments. This more closely mimics the business environment and allows the students to focus on their projects rather than trying to find and configure extra computer hardware necessary for their projects. This has also allowed us to accommodate projects which require special software as the sponsoring company can create a Virtual Machine (VM) which we can easily install.

Our virtualization server is a dual quad-core Xeon processor (i.e. eight total cores.) server with 2.0 Ghz per core and 16 GB of RAM. We are use VMWare ESXi version 4 for our virtualization platform. ESXi is a free version of VMWare's ESX hypervisor, which is a minimalistic operating system designed exclusively to host virtual machines. This, when combined with the free VSphere tool for managing virtual servers, allows for a free virtualization system. The tools provided by VMWare for free are rather basic, but sufficient for our purposes. VMWare also has a number of tools available for other high-level virtual machine management functions, but the cost of such tools may be prohibitive and the usefulness of such tools for this purpose limited.

Previously, we have used multiple physical servers to provide services that teams required for their individual projects. This caused a significant amount of administrative overhead as well as complications with multiple teams using the same servers. Occasionally, students were also forced to use their own hardware to run special software if the university was unable to provide sufficient hardware resources. Two years ago, we began to use virtualization to ease administrative overhead and to allow for teams to work in their own distinct environments. Initially we used the free and open-source virtualization program VirtualBox. VirtualBox functioned as expected, but we experienced problems with the management of multiple virtual machines, and VirtualBox lacks a bare-metal hypervisor. This means that VirtualBox requires that a full operating system be installed for VirtualBox to function.

Future Work

Migrating to SVN and Trac has made it easier to keep track of student work and assess student progress. Utilizing virtualization has also reduced the amount of work required for our IT support staff. In the future we are planning to conduct empirical studies to measure improvement in student learning through exposure to these tools. Our experiences with these tools suggest that they are beneficial for students and will help them in their future careers, but we have not yet performed a formal assessment of their impact.

Because most SVN and Trac environments for student projects are similar, their deployment can be scripted. We have worked towards this but have not yet completed our work on a full-featured SVN/Trac management tool. The completion of such a tool would provide assistance for other universities interested in implementing Trac and SVN. Another long term goal is to create a customized Linux distribution with all the scripts and packages required for easy deployment of these tools and services.

Conclusion

Through the use of Trac, Subversion, and virtualization we have been able to improve our capstone course. Our primary motivation for using these tools is to comply with CMMI Maturity Level 2 requirements, but these tools can be used even when not following a thorough process. Additionally, these tools closely resemble those used in industry and provide a great learning experience for students.

Using these tools requires a moderate knowledge of the Linux operating system and some additional administrative overhead when compared to ad hoc solutions; however, they do allow students to focus on their capstone project. Compared to other solutions that we have used in the past, Trac, SVN, and VMware ESXi provide a better experience for our students and sponsoring companies.

References

[1] Dean Knudson, Alex Radermacher, “Software Engineering and Project Management in CS Projects vs. ‘Real-world’ Projects: A Case Study”, *IASTED-Software Engineering Applications 2009*, Cambridge, MA, November 2-4, 2009.

[2] Capability Maturity Model Integration (CMMI), Version 1.1, Staged Representation, CMU/SEI-2002-TR-029, ESC-TR-2002-029, Software Engineering Institute, 2002.

[3] Dean Knudson, Kenneth Magel, “Comments on the Use of TWiki, Blackboard Portfolios, and Trac to Share Proprietary Information in Student Projects”, *SITE 2008 – Society for Information Technology & Teacher Education Information Conference*, Las Vegas, NV, March 3-7, 2008.

[4] Curt Jones, “Using Subversion as an Aid in Evaluating Individuals Working on a Group Coding Project”, *Journal of Computing Sciences in Colleges*, 25, 3 (Jan. 2010), 18-23.

[5] Louis Glassy, “Using Version Control to Observe Student Software Development Processes”, *Journal of Computing Sciences in Colleges*, 25, 3 (Feb. 2006), 99-106.

[6] The Trac Project Homepage http://trac.edgewall.org/

[7] The Subversion Project Hompage http://subversion.tigris.org/

[8] Peng Li, “Selecting and Using Virtualization Solutions – Our Experiences with VMware and VirtualBox”, *Journal of Computing Sciences in Colleges*, 25, 3 (Jan. 2010), 11-17.

[9] VMWare Homepage http://www.vmware.com/

[10] VirtualBox Homepage http://www.virtualbox.org/