Jordan Eisele, Sebastian Pappert

CSE481E

06/05/2007

Report for the Project "CSE481E: Capstone Software -- UrbanSim"

New Build System

Context for the project

The Center for Urban Simulation and Policy Analysis (Cuspa) [7] develops the UrbanSim software which provides a platform and models for an urban simulation system [1]. The Software consists of several sub projects, Open Platform for Urban Simulation (Opus) which is the framework to run models, sets of models for different areas or cities, e.g. Urbansim or for the Puget Sound area, and a sub project for documentation. Most of the developers work in the Cuspa lab but contributions are also received from abroad. The development process used by Cuspa applies several agile programming concepts [2]. The most important of which are a test first methodology and short iterations in which tasks determined. In order to always have a runnable system without any failing tests a build system is used. Build systems automatically check source repositories for modifications, run scripts for the source code, and can display the script results in a number of ways. Currently the source code for the projects is stored in a CVS repository and Cuspa is using a home-brewed build system called Fireman. In the case of Fireman, the scripts are used to primarily run unit tests and the results are displayed on a fireman website and traffic lights within the Cuspa lab [3]. Because of the outside contributions to UrbanSim, Cuspa has encountered some difficulties with using CVS. The accounts used by CVS to access the repositories are closely tied to the underlying system, effectively meaning outside contributors must be given UW CSE accounts. Switching to a different version control system, such as Subversion, would eliminate the need to give out UW CSE accounts, as the accounts are independent of the underlying system. However, Fireman only supports the use of a CVS repository. Since the creation of Fireman an open source alternative has emerged which does support SVN: CruiseControl [8]. Besides the support for SVN CruiseControl has several other advantages over Fireman. Fireman is a home-brewed system so its development takes time away from Urbansim projects. Also, Fireman is written in Perl with limited extendability and has very verbose log output making it difficult to locate specific errors. Conversely, CruiseControl is written in Java with the idea of easily extending CruiseControl in mind and also has the ability to neatly output XML log files in a manner that emphasizes any errors. The advantages CruiseControl has over Fireman has led to the idea of migrating from Fireman to CruiseControl being floating around Cuspa for several years. With the need to move away from CVS the move to CruiseControl then became a priority.

Installing and setting up the new build system

Thus, the objective of our project has been to facilitate the migration of the Urbansim projects from CVS to SVN and to set up CruiseControl as the new build-system managing the projects. The actual setup of CruiseControl itself is extremely easy only requiring it be unzipped. From there a build script can be hooked up to a quick project configuration [4] to achieve a basic build system. We were able to quickly get a very simple build system working from this and spent the majority of our time setting up a build parallel to Fireman, testing the new build-system, adding features and duplicating functionality provided by Fireman. Throughout working on the new build-system we have strived to achieve a system which incorporates all of the major features of the build-system under Fireman. These features are: the management of many different sub projects, able to manage distributed development, publishing of build results through the traffic lights in the Cuspa lab and on the Urbansim website, and automatic building and publishing of various installation files and manuals.

CruiseControl already has the ability to manage many different projects, each with it's own log and the ability to force a build of a specific project. However, we did implement some special setups. Each project has its own configuration file that is loaded into CruiseControl. This results in a configuration setup similar to that of Fireman and also make it very easy to add projects. Also, CruiseControl has the ability to produce a neatly organized output document from XML log files that are in the style of the output produced by the JUnit package. To take advantage of this we had to write an extension of the code used to run the unit tests of the Urbansim projects so that is was capable of producing XML output in the correct format. Previously the test runner was only capable of outputting a plain-text log, but we were able to seamlessly integrate the new XML output feature in a manner that is entirely invisible by having the test runner only use XML output if it is invoked with the -x option. With this we were able to use the build scripts originally written for Fireman with only minor changes, e.g. implementing a clean interface to the build file by creating an entry point target called cruisecontrol in each build script. This interface gave us the possibility to insert a thin layer between CruiseControl and the build scripts for executing tasks before and possibly after every build script. These tasks include collecting the overall status of the project and controlling the traffic light building states. Finally, we added the new feature of dependencies within project configuration files. This means that whenever a project is successfully built all depending projects are then built in turn. This is something that is not available in Fireman. Previously, Fireman had to rely on the nightly building of all projects to try an achieve a similar functionality.

With the new build-system we have set up, we are able to take the distributed development capabilities of Fireman and expand upon them. With SVN it will be easier to give access to Urbansim projects to people in other locations and their contributions will be integrated into the projects. Also, CruiseControl will be able to quickly catch errors in contributed code and make them known to the Cuspa lab. Most of this functionality we got for free from our setup, but it did require the setup of some access restrictions. The page which displays the results of project builds also directly contains the ability force builds of any of the projects and allows access to a CruiseControl settings console. We were able to set up account restrictions on both forcing builds and access to the console. Because of this, our new build-system is an ideal solution for distributed development.

Duplicating Fireman's publishing of build results for the Urbansim projects was one of the more time consuming parts of the project and presented some interesting challenges. Cuspa has a number of traffic lights in the lab that use color coded signals to indicate the status of the Urbansim projects as a whole. This functionality has been deeply integrated within Fireman and CruiseControl has nothing resembling this kind of functionality. Additionally, there is no means directly within CruiseControl of getting the status of all projects combined. To match the functionality that Fireman is providing we had to write Ant scripts to keep track of both the individual project statuses and the status of all projects combined. We did this using Ant properties, so it is not tied into CruiseControl or any Urbansim packages whatsoever. It's also very easy to use, only requiring a call to an ant target. Another problem with controlling the traffic lights was actually communicating with the traffic light server. We were able to accomplish this fairly simply by writing an Ant script that uses a telnet Ant task, however, using the telnet task requires an additional library file. Using extra libraries caused some difficulties for us because Ant checks a system wide configuration file, which we do not have access to, for the location of library files and the library location was being set to a different installation of Ant. Once we discovered the problem we were able to remedy it by creating a user Ant configuration file to override system wide configuration and set the library location to the version of Ant we use. As for publishing results to a website, CruiseControl has this type of functionality built in, although as previously stated, we had to restrict access to some of the features.

Fireman automatically builds and publishes an installer for Opus Core, a zip file for the Eugene tutorial, and a manual for Opus Core. The manual is handled in its own project and was very easy to migrate into CruiseControl. However, the Opus Core installer and Eugene zip file are handled in a nightly build. The nightly build in Fireman also builds every project to ensure nothing was broken due to something going wrong in a project that is depended on. We were able to accomplish this using dependencies, so building every project in a nightly build is unnecessary. Although, we did create a nightly build just for creating the installation files and publishing them. We did encounter some problems adapting the scripts to build the installers for CruiseControl, mainly due to path issues. To build the installation files we copy the projects needed for the installers into a sandbox folder. We invoke the script to build the installation files from within the sandbox folder to isolate it and prevent other projects being pulled into the installation. Once the building of the installation files has been done, a publisher task in CruiseControl publishes the file to a specified folder.

Results, problems and future additions

The installation and set up process for CruiseControl turned out to be very straight forward but we did encounter a major problem with CruiseControl. It spawns a new process to check the repository for updates and the main process waits for the result. Sometimes the repository checking process did not finish and CruiseControl got stuck. The only possible way to get CruiseControl back to work without changing the code of CruiseControl was to find and kill the spawned process [5]. We noticed that CruiseControl has the same problem with checking the subversion repository but it seems to happen less often. Therefor we have to monitor CruiseControl in the next few weeks before we decide whether this is still a major problem or not.

Other than the CruiseControl bug, the setup of CruiseControl parallel to Fireman was fairly straight forward and went smoothly. Once we had determined that the parallel build-system did not differ from Fireman in any significant manner, we decided to migrate the projects in CruiseControl to SVN. The migration process from CVS to SVN was very easy, although time consuming. The process required only exporting each project in CVS to SVN and changing the modificationset within each project configuration from CVS to SVN.

Additionally, the integration of new sub projects into the new build-system should be quite simple, although we have not tested this. To add a sub project should only require access to the source code in some repository, a build script for the project, and the set up of a very simple CruiseControl configuration file for the project. To integrate the new project with the traffic light would only require that the project's build script comply with our interface.

With this project we have managed to produce a build-system that closely matches the functionality of Fireman, but with key advantages. However, there are some things which could be added in the future to further improve the system. Currently, the testing framework for the Urbansim projects mixes true unit tests and functional tests. A nice improvement would be to separate these two types of tests so that whenever a modification is made only the unit tests are run, and the functional tests are run only in a nightly build. However, separating the tests is not simply a CruiseControl issue, it would also take modification of the Urbansim testing framework and is a major undertaking. But this change would not only affect CruiseControl but also the individual developing process on local machines. Test-driven development depends on repeated runs of unit tests and the speed of the tests affects the developing process directly and the willingness of the individual developers to adhere to the test-first philosophy [6].

References

[1] Alan Borning, Paul Waddell, and Ruth Förster, UrbanSim: Using Simulation to Inform Public Deliberation and Decision-Making. Preprint of paper to appear, *Digital Government: Advanced Research and Case Studies*, Hsinchun Chen et al. (eds.), Springer-Verlag, in press.

[2] K. Beck. Extreme Programming Explained: Embrace Change. Addison-Wesley, Reading, Mass., 2000.

[3] Bjorn Freeman-Benson and Alan Borning, YP and Urban Simulation: Applying an Agile Programming Methodology in a Politically Tempestuous Domain. *Proceedings of the 2003 Agile Development Conference*, Salt Lake City, Utah, June 2003.

[4] http://cruisecontrol.sourceforge.net/main/configxml.html

[5] http://thread.gmane.org/gmane.comp.java.cruise-control.user/16040/focus=16253

[6] K. Beck. Test-Driven Development – By Example. Addison-Wesley, Reading, Mass., 2003.

Links

[7] http://cuspa.washington.edu/

[8] http://cruisecontrol.sourceforge.net/