**CS 461 / 462 / 463**

**Client Requirements Document Outline**

**Project Name**

**Team Name:** Walnut Crusade

**Team Members:**

Matt Wilson – wilsomat@onid.oregonstate.edu

Colin Bradford – bradfoco@onid.orst.edu

Nicholas Smith – smitnich@onid.oregonstate.edu

Matt Vanderhey – [vandehma@onid.orst.edu](mailto:vandehma@onid.orst.edu)

**Mentor:**

Jan Iversen - [jani@apache.org](mailto:jani@apache.org), Apache Software Foundation

**Additional Assistance:**

Steve Hathaway – shathaway@apache.org, Apache Software Foundation

**Introduction to the problem:** Building the OpenOffice software through Visual Studio as a Windows developer is difficult and has to be done through CygWin with makefiles. The idea is to move from a makefile based platform to a Visual Studio based platform. If possible, we would also allow the easy conversion from this platform to other platforms.

**Project Description:** Make it easy to build the system in Visual Studio and then convert between different build systems.

**Requirements:**

- Create a Visual Studio vcxproj file for every makefile in the AOO suite

jan: 1 for every module (dirs in main), each vcxproj file contains all makefiles in that module.

- Create a Visual Studio solution file encapsulating all converted modules

- Ensure AOO solution can be built with the old build system using some modules built using Visual Studio to facilitate proper integration

- Allow AOO suite to be built from within Visual Studio without any additional tools beyond what is already required

- Allow for conversion from vcxproj files into makefiles using XML parsing/stylesheet applications, or other build platform formats

jan: I think you mix. requirements and solution ?

**Versions:**

- Version 0.1 is manually converting one module from the current build system into a vcxproj file, and testing it to be sure it builds when placed within the current build system

- Version 0.2 is converting 10 modules into vcxproj files and testing their results within the current build system

- Version 1.0 is writing a script that will help streamline the conversion process between modules and vcxproj files. Note that this version is optional, and may not be feasible. If it isn’t feasible, future module conversion will be done manually like in versions 0.1-0.2.

- Version 1.1 is converting 35 modules into vcxproj files.

- Version 1.2 is converting 60 modules into vcxproj files.

- Version 1.3 is converting 85 modules into vcxproj files.

- Version 2.0 is placing all converted vcxproj files into a Visual Studio solution file and cleaning up the solution for future work

jan: you have the dependencies, so you should be able to name the modules, at the very least the first 10.

jan: I wonder how you got 180 modules ? I count 191.

**Design:**

- One-to-one relationship between modules in the current AOO build system to (XML) vcxproj files that can be built within Visual Studio

jan: the current makefiles are written over 10 years by many people, so 1-1 relation does not exist. when the vc solution is finished new makefiles will be generated.

- Periodically the projects should be built and then inserted into the old AOO build system, whereby we can verify that the entire suite builds using some modules built with makefiles and other modules built with vcxproj files

- One Visual Studio solution file containing all converted vcxproj files

- Create a script using Apache Xerces to convert makefiles, possibly only parts of them or certain types, into equivalent vcxproj files to help streamline the conversion process

- Apache Xerces parsing of Visual Studio (XML) vcxproj file into other build formats, such as makefiles

**Specific tasks to be undertaken:**

- Build AOO system through CygWin, as is the current procedure

- Conduct research on vcxproj file format, makefile format, similarities & differences, Visual Studio environment variables, build rules, and so on

- Select a module with no dependencies from the AOO solution and convert its makefile into an equivalent vcxproj file, then build this through Visual Studio. Once built, insert this into the current AOO build system and verify that it builds properly.

- Select several more modules, possibly some with dependencies on already converted modules, and repeat the previous task with them.

- Research Apache Xerces functionality, use it to try to write conversion protocol between makefiles & vcxproj files. Don’t spend too much time on this task.

- Convert a lot of makefiles into vcxproj files. Break this up into milestones (25 at a time) to keep it manageable.

- Compose a Visual Studio solution file encapsulating all converted modules.

- Write up documentation on what we’ve learned, how to include new modules in the Visual Studio solution, how to use Xerces, etc.

**Risk Assessment:**

1. If there is no feasible way to automate certain aspects of converting between modules & vcxproj files, then we’ll have to do some conversion manually. This will take slightly more time, but should not prevent us from converting a majority of modules.

2. If it takes longer than expected to convert modules, then we won’t be able to hit some of our milestones, but we’ll at least have a theoretical base that can be utilized in the future.

**Testing:**

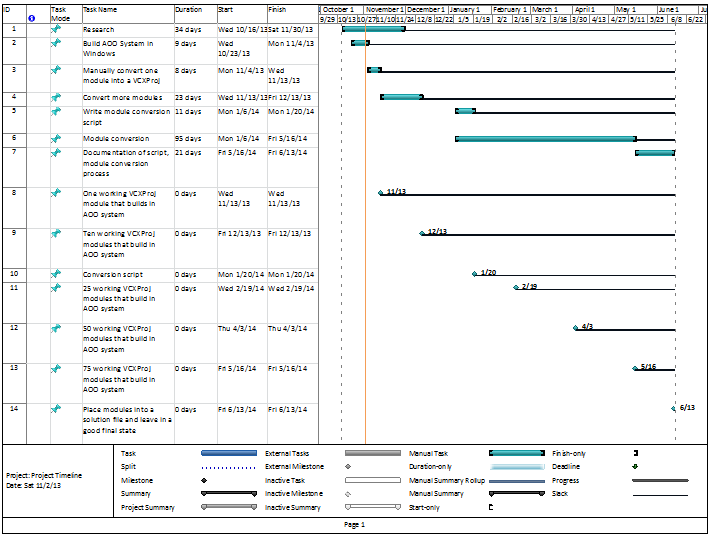
- The existing build system should be able to build using modules that were built with Visual Studio. This will ensure that the old build system will still work while constructing the new one.

- Possibly try building the Visual Studio solution using modules built with the current build system.

- Make sure that the compiled AOO .exe runs, and that its functionality works as expected

- There are a few modules whose functionality will need to be verified within the built application itself, but most we will assume are fine if the solution is built

**Preliminary Timetable:**



**Roles of the different team members:**

Matt Wilson - Scheduling, P3, Module Conversion

Nick Smith - Module Conversion

Colin Bradford - Secretary, Module Conversion

Matt Vandehey - Module Conversion

**Integration Plan:**

- Make sure current build system works when using modules created with Visual Studio projects

- Make sure Visual Studio solution works when using modules created with current build system

- Make sure converted modules do not conflict with each other

**Dataflow sequence diagram:** not sure what to do with this - any suggestions? This isn’t required, but if it’s useful

**User interface requirements:** None

**References:**

1. Current build system description: <http://wiki.openoffice.org/wiki/Documentation/Building_Guide_AOO/Step_by_step>
2. Apache Open Office website: <http://www.openoffice.org/>
3. First attempt at remaking the build system:
   1. Zen of gbuild: <http://wiki.openoffice.org/wiki/Build_Environment_Effort/Zen_of_gbuild>
   2. Gbuild Bootstrapping: <http://wiki.openoffice.org/wiki/Build_Environment_Effort/Gbuild_Bootstrapping>
   3. Module Migration: <http://wiki.openoffice.org/wiki/Build_Environment_Effort/Module_Migration>
4. Time Converter: <http://www.worldtimebuddy.com/?pl=1&lid=5720727,100&h=5720727>
5. Wiki for project: <http://wiki.openoffice.org/wiki/Build_System_Analysis:capstone2013_windows_build>
6. Build System Analysis: <http://wiki.openoffice.org/wiki/Build_System_Analysis>
7. Apache Xerces website: <http://xerces.apache.org/>
8. W3 XML tutorials: <http://www.w3schools.com/xml/default.asp>
9. Visual Studio changes: <http://msdn.microsoft.com/en-us/library/ee862524.aspx>

**Glossary:**

1. vcxproj: similar to a makefile in Unix/Linux systems. Determines how the project is compiled. Used to be vcproj files until more recent versions of Visual Studio. vcxproj files are written in XML.
2. AOO: Apache Open Office
3. Xerces: An XML parser developed by Apache.
4. CygWin: an emulation of Linux tools in Microsoft Windows.

**Signatures:**