M12 Individual Case Study

General
This is an INDIVIDUAL assignment. All work and analysis should be your own and reflect your personal conclusions.

Choose a system
Look on our syllabus page for the electronic reading called: "The Architecture of Open-Sourced Applications." There are several systems described there in two volumes. Pick one that might interest you. If you have no other preference, the discussion on Eclipse (from volume 1) is good. Otherwise, there are examples from almost every domain from games to sound processing to networking.

Read the author's description of the system
Now for your chosen system, read and think about their comments. This is a rationale for how their system is put together. Since these are open source systems, you can obtain the source code if you wish to do more detailed analysis of the system. If you are interested in distributed, scalable web systems, then look at the book: Exploring CQRS and Event Sourcing; A journey into high scalability, availability, and maintainability with Windows Azure. There is a link to this on the web page also.

Perform a case study analysis of the design.

1. Start by identifying the problem the software was supposed to solve.
2. What is their user demographic (who is the software targeted towards: students, developers, musicians, etc).
3. What architectural style do they seem to be using? What would you have used given their constraints?
4. What seems to be their criteria for packaging (grouping classes or code together)? Is it maintenance or reuse?
5. Look at their description and then apply the SOLID design principles to discuss application of these principles to the approach taken by the software.
6. Identify design patterns that would be especially helpful in solving their particular problem.
7. What quality factors (like FURPS, or maintainability) are addressed in their design?

Length
Please limit your writeup to 3 single-spaced pages of text. Supporting figures may be of any length. Quality is more important than volume.

Format (Paragraph lengths are suggestions, not requirements)

Introduction (1-2 paragraphs):
First, open with some background information about the program. Give the reader some information on why the program was written and who it was started by. Try to give a general idea of the current state of computing at the time, especially if it is relevant to the reason behind the project being started. Try to give some supporting examples of how it is addressing whatever problem it was meant to address. End your beginning segment by explaining who the project was initially written for, and explain if that has changed at all over the course of time. List the uses that this demographic uses for (Audacity was written for musicians to edit their music before publishing).

Description(1-3 paragraphs):
Now that you've introduced the program and its origin and uses, begin the next section by describing the implementation of the project. Make a point to talk about its current architecture style, packaging and structural design patterns. Try to be as descriptive as possible of the project, so the reader can be fully aware of the project's state during your analysis of it. Make sure to fully describe all the aspects of the project that you will need to analyze below

Analysis(3-4 paragraphs):
First off, in each section of your analysis, make sure you are clear about what aspect you are currently analyzing. It is usually easiest to start by analyzing its architecture. Does it seem to meet the needs of the project? Should they look at changing it? If not, why is the current implementation correct? Next, apply SOLID and give some supporting evidence or examples behind your application of each. Many projects may not be object-oriented styles, so explain how they are/aren't using SOLID and whether that is a good/bad thing. Are there currently any major issues with the project that should be addressed? Try to

find a current area the program is lacking in addressing and explain a design pattern that will help them address it. If there isn't a specific issue, explain a design pattern that will help them continue to achieve addressing the problem the program is mean to solve. Again, don't forget to support any arguments you give. Now that you've analyzed the development of the code, we need to critique the quality of the program. One of the best means to do this is FURPS. Address each part of the acronym critically and give solid experience examples when possible. Eclipse is a good example of a program that exemplifies the S(supportability) by the large amount of plugins available to a program developer of almost any language.

Conclusion(1 paragraph):

Take this moment to do a quick recap of the more important observations you feel you've made. Then, give an overall conclusion as to whether you feel the design is a good, average or bad one.

Grading

1. A - Paper shows student has a mastery of both high-level (architecture) and low-level (class/code) design principles. Appropriate styles and design patterns are identified or suggested and show a well-grounded knowledge of design. Opinions and suggestions are supported if necessary by UML or concrete figures illustrating proposed solution.
2. B - Paper shows student has general understanding of the high and low level design concepts. Appropriate styles and design patterns are identified or suggested and show a basic understanding of design. Supporting figures are provided if needed, although there may be minor issues with the diagrams.
3. C - Paper shows student has some problems with understanding the architecture or low-level design concepts. Some styles and patterns may not be appropriate for the problem. Some needed supporting figures are provided, and there may some errors in the diagrams.
4. D - Paper shows student has major problems with understanding the architecture and design of a system. Poor analysis of situation and almost no use of design patterns. Required supporting figures have major problems, missing information or are not understandable.
5. F - No real analysis of the problem. No supporting figures or documentation when they are needed to understand the analysis.