CS CAPSTONE TECHNOLOGY REVIEW

NOVEMBER 21, 2017

SOLAR CAR SIMULATION

PREPARED FOR

PHOENIX SOLAR RACING

CAILIN MOORE

PREPARED BY

DAKOTA ZAENGLE

INFERNO

Abstract

In this document I will discuss three of the technologies we will be using for our project. For each of these technologies there will be three choices and at the end of each section I will discuss the differences and state which choice we will be using.

CONTENTS

1 GUI FRAMEWORKS

1.1 Overview

In this section I will discuss the top choices we have for our GUI framework. This is what we will be using to create the interface that Phoenix Solar Racing will interact with. These choices will determine how the program looks, some will look more natural on the system we run it on because they use the operating system's features for their own. These frameworks will also influence how we display data, some come with APIs for data visualization while others simply work well with certain visualization frameworks.

1.2 Criteria

1) The framework should be free or inexpensive.
2) The framework should be usable with a visualization framework or support its' own.
3) The framework should be well documented. We shouldn't spend much time looking for resources to learn it.

1.3 Potential Choices

1.3.1 Qt

"Qt is a comprehensive cross-platform framework and toolkit that helps you create and build native applications and user interfaces for all the screens of your end user. With Qt, you can reach all your target platforms with one technology and one code base, minimizing your time-to-market and maintenance burden. Qt for Application Development is available under a dual-licensing model you choose whats right for your needs." [3] This means several things for us. First it will be easier to make our software run on different operating systems if the need arises. Second we wont need to pay for it because it is offered open source. This does mean however that we will have to follow some open source rules if we choose to use it.

From their website these are the rules we must follow to use the open source version [3]:

1) Using parts of Qt that are only available under GPL requires open sourcing of your application when distributing
2) Must provide a re-linking mechanism for Qt libraries
3) Must provide a license copy & explicitly acknowledge Qt usage
4) Must make a Qt source code copy available for customers
5) Qt source code modifications arent proprietary
6) Must make open consumer devices
7) For Digital Rights Management see GPL FAQ
8) Special consideration should be taken when attempting to enforce software patents

All of these should be agreeable however because we are building this software for PSR at the lowest cost possible and we will be making our work available to them so they may make changes in the future. Both the capstone team and PSR do not intend to re-market or profit off the software so as long as we follow the requirements we should have no trouble using this version.

So does Qt meet our criteria? It is offered free for open source use. It works well with a visualization framework that we will discuss in a later section. And because of its age and popularity there are many tutorials and examples available online. Therefore it does meet all of our criteria.

1.3.2 wxWidgets

"wxWidgets is a C++ library that lets developers create applications for Windows, Mac OS X, Linux and other platforms with a single code base. It has popular language bindings for Python, Perl, Ruby and many other languages, and unlike other cross-platform toolkits, wxWidgets gives applications a truly native look and feel because it uses the platform's native API rather than emulating the GUI. It's also extensive, free, open-source and mature." [4] For our use this GUI would be a very solid choice. Because of its use of the native Windows API it will, as they describe, have a native look. Additionally, as we will see in the *Visualization Framework* section wxWidgets comes with libraries for data display so we would not need to locate our own and hope it works easily with our GUI framework. Like the version of Qt we would use it is open source and free. So we will need to read and obey their license terms if we choose to use this library.

So does wxWidgets meet our criteria? It is offered free and open source. It comes with its own visualization libraries. And because it is also fairly popular it has a number of tutorials along with its' own documentation.

1.3.3 WPF

"WPF, which stands for Windows Presentation Foundation, is Microsoft's latest approach to a GUI framework, used with the .NET framework." [6] This framework is what Microsoft Visual Studio is currently written to use. As long as we are sure PSR will only run this software on Windows machines this makes WPF a good option. Because WPF works directly with Visual Studio we would be guaranteed to find many tutorials and examples to help us use it. However, WPF is written with C# instead of C++, while we could still use C++ for most of the program we would still need to know and use C# for at least some of the GUI. Using C++/CLI would allow us to mostly use C++ but it would not completely get rid of C#. This requirement makes WPF a bit of a weaker choice as it will make our lives, and the lives of anyone who modifies the code after us, a bit harder.

Regardless does WPF still meet our criteria? It is offered with Visual Studio, which we have access to as engineering students. Again because it works through Visual Studio it has visualization frameworks available. And because it is Microsoft's official choice there are multitudes of tutorials and documentation available.

1.4 Discussion

For this section I have included a table that lists the basic pros and cons of the three frameworks we are looking at. WPF was much harder to find information on than the other two frameworks as can be seen by the lack of info in the included table. Because of this I believe it will also be more difficult to find helpful documentation. Both Qt and wxWidgets were well documented on their sites and had a wealth of tutorials and documentation on how to use them.

Framework	Pro	Con
Qt	Open Source Works with some visualization frameworks Easy to work with on multiple operating systems	Lots of License to follow
wxWidgets	Open Source Has its' own visualization library Uses the native API (looks pretty) Can be used on different operating systems	
WPF	Uses Microsoft Visual Studio	Requires us to use C#

1.5 Conclusion

From the pros and cons table the choice lies between wxWidgets and Qt. Because wxWidgets is so much more open source friendly and the documentation of both is comparable it seems wxWidgets is the better choice. Additionally it comes with its own visualization framework as an added bonus.

2 Software Testing

2.1 Overview

In this section we will look at the different forms of testing we may choose to use for this project. Unlike the first section we may not end up with just one answer. Instead I will look at the forms of testing that may apply to this project and decide which types of testing we will perform during this project and how important it is that we use them. I found the *Software Testing Dictionary* on tutorialspoint.com helpful in picking these potential choices [1]

2.2 Criteria

The criteria for testing are fairly simple:

1) The test should provide us with information to improve our software.
2) The test should require a reasonable amount of time for its' payoff.
3) The test should be repeatable.

To clarify the first item, when we review any of these tests we should be able to use the information we gathered to make changes to our software or to verify completion and software stability. The three tests I have picked out already meet these criteria as there were so many forms of testing I had to find a way to narrow down the options.

2.3 Potential Choices

2.3.1 Acceptance Testing

Acceptance testing is a form of software testing where the program is tested by users to decide if it complies with the end requirements. This type of testing is central to verifying completion of our project and is included in our project requirements. We will use it when we feel we have completed our software in order to certify the software is usable by PSR and has satisfactorily met our project requirements. We will be focusing on External acceptance testing which means we will not be the 'user' in this test. Instead we will ask members of PSR to attempt to use the software for our final tests.

2.3.2 GUI Testing

GUI testing is a method of testing in which the user interface is tested to ensure it functions according to specifications. This testing would be performed before acceptance testing and would not necessarily need the program to output final results. Unlike the acceptance testing, we the designers would perform this testing in between GUI iterations to identify necessary changes.

2.3.3 Compatibility Testing

Compatibility testing can be used to make sure our software can run on different systems with varying operating systems and hardware. There are many different levels but we would focus on hardware compatibility and operating system compatibility. This will be done somewhat automatically in development however it would be beneficial to test the software during development on the systems PSR will be using and not just our own. Once the software is complete we will certainly test on their systems however it may also be helpful to test while we are building the program to identify any problems that arise as quickly as possible.

2.4 Discussion

Testing	Benefit	Time Commitment
Acceptance Testing	Can be used to determine project completion We can receive helpful feedback from our client Can be used to introduce the final software to PSR	Two hours with PSR, may be repeated
GUI Testing	We can do this testing ourselves 'in house' This test can help us to improve GUI features Can be automated	Maximum 20 minutes each iteration
Compatibility Testing	Uses Microsoft Visual Studio Ensure PSR won't encounter critical failures	Performed with other tests Testing on PSR systems adds 30 minutes

2.5 Conclusion

To conclude, we must use acceptance testing to complete our project requirements. GUI testing is somewhat optional. It will be performed simply through use of the software but the structured form of testing could be helpful in early iterations. Finally compatibility testing is guaranteed to happen at the end of the project. In addition we will hopefully have opportunities to test it on PSR's hardware during development.

3 Visualization Framework

3.1 Overview

These visualization frameworks are the libraries we may be using in order to display the final output of our software. Some of these are built into the GUI frameworks and others are made with certain GUI frameworks in mind. Therefore the choice of visualization framework will be mostly dependant on the GUI framework we choose.

3.2 Criteria

The criteria for visualization framework are as follows:

1) The framework must be compatible with whichever GUI framework we choose.
2) Tutorials and documentation must be available.
3) The framework must provide tools to output any graphs or other forms of data display we may use.

3.3 Potential Choices

3.3.1 VTK - Visualization Toolkit

"The Visualization Toolkit (VTK) is an open-source, freely available software system for 3D computer graphics, image processing, and visualization. It consists of a C++ class library and several interpreted interface layers including Tcl/Tk, Java, and Python." [2] This toolkit was chosen because it is compatible with Qt, one of the GUI choices so it passes our first criteria. This, like the other visualization frameworks, is very powerful and capable of 3D rendering so it will surely cover our needs and pass our third criteria. The framework is well documented, passing our second criteria as well.

3.3.2 OpenDX

"OpenDX is a uniquely powerful, full-featured software package for the visualization of scientific, engineering and analytical data: Its open system design is built on familiar standard interface environments." [5] Like their description says, OpenDX is powerful, however it could be more than we need for this project. The documentation for this framework is thorough but dense, meaning it is questionable if it meets our second criteria or not. Additionally I was unable to find reference to GUI frameworks that OpenDX would work with although it does seem to have GUI libraries.

3.3.3 wxWidgets

This framework is the same as what we discussed in the GUI section. Because it comes with its own visualization libraries it is also included here for discussion. [4] Like I talked about in the GUI section this framework provides both GUI and visualization frameworks which is very convenient as we will not need to locate both and confirm they are compatible. On the wxWidgets website they have screenshots and links to many programs created with their framework including many 3D applications so this framework will be sufficient for our visualization needs and meets all 3 of our criteria for this section.

3.4 Discussion

In Section 1 I concluded that we would be using wxWidgets and by extension its' visualization framework. However I am also including this section even though the choice has been made because these frameworks were considered when I made the choice of GUI framework. I felt it was important to include what research I did on visualization frameworks that lead to my choice of GUI framework as well.

3.5 Conclusion

Again because wxWidgets was the final choice of GUI framework it is what we are choosing here too. It will be more convenient to simply use one API for both GUI and visualization so this choice benefits us for both as well.

REFERENCES

[1] T. Point, Software Testing Dictionary, www.tutorialspoint.com, 15-Aug-2017. [Online]. Available: https://www.tutorialspoint.com/software_testing_dictionary/index.htm.

[2] "VTK", https://www.vtk.org/, [Online].

[3] "QT", www.qt.io, [Online].

[4] "wxWidgets", www.wxwidgets.org, [Online].

[5] "OpenDX", http://www.opendx.org/index2.php, [Online].

[6] "WPF", www.wpf-tutorial.com/about-wpf/, [Online].