Using the Peppers Ghost Pyramid technique to create real-time holograms for a charades style game

Final Report for CS39440 Major Project

Author: Elliot Oram (elo9@aber.ac.uk)
Supervisor: Dr. Helen Miles (hem23@aber.ac.uk)

28th February 2017
Version: 1.0 (Draft)

This report was submitted as partial fulfilment of a BSc degree in Computer Science (G401)

Department of Computer Science
Aberystwyth University
Aberystwyth
Ceredigion
SY23 3DB
Wales, UK

Declaration of originality

I confirm that:

- This submission is my own work, except where clearly indicated.
- I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
- I have read the regulations on Unacceptable Academic Practice from the University's Academic Quality and Records Office (AQRO) and the relevant sections of the current Student Handbook of the Department of Computer Science.
- In submitting this work I understand and agree to abide by the University's regulations governing these issues.

Name .....

Date .....

Consent to share this work

By including my name below, I hereby agree to this dissertation being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Name .....

Date .....

Acknowledgements

I'd like to thank Dr. Helen Miles for her continued support throughout the duration of this project which included advice and guidance on the project context and implementation. In addition I would like to thank the members of the Aberystwyth University computer science department for their assistance in setting up the stall for the prototype demonstration at the 2017 Aberystwyth Science week event.

Abstract

This report describes the process followed to develop a system to create real-time holograms using the Pepper's Ghost pyramid technique and an accompanying charades game. The system was first proposed to be used at Aberystwyth University's Science week in 2018, but is also suitable to be displayed at any appropriate outreach event. The Pepper's Ghost pyramid technique is an excellent tool to use for outreach as it is simple to understand, and in addition creates an impactful display. In order to improve the experience for participants, the charades game is designed to provide a way to interact with the holographic system.

This report will present the purpose, technologies and processes surrounding this system. The development process followed an adapted single person Feature Driven Development (FDD) plan driven methodology which will be discussed in detail below. The report will aim to focus on the software quality and testing performed throughout the development of the system.

Finally, the report will conclude with a critical evaluation of the system itself, the implementation choices and the development processes. This will include, but not be limited to, the programming languages and frameworks used, continuous integration, source control and testing tools, and prototyping and implementation testing.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Background & Objectives

1.1 The scenario

The project is to create a system to produce real time holograms from a camera feed, using the Pepper's Ghost pyramid technique. The system is intended for use at the Aberystwyth Science Week next year, however, there was the potential to display a prototype at the 2017 event held in mid-March. The project is intended to show case a technique, originally used in stage and theatre productions, and give insight into how it can be adapted, with the aid of computer science, to make an impactful visual display.

The system will take real-time data captured from a web cam in the staging area. The staging area will consist of a black background and appropriate lighting to illuminate the subject. The video feed will then be processed and displayed on a large monitor to work with a ghost pyramid of appropriate size.

To make the final demonstration more interactive, the display will have an accompanying system that allows users to play a charades style game. This would display a topic for the user to act out in the staging area and then allow those viewing the hologram to guess the activity being performed. This would require a system capable of taking multiple different string in puts (guesses) from users simultaneously, and feeding back to all users if a guess is correct.

1.2 Motivation and Justification

This project is appealing due to the uniqueness of the technology being used to produce it. Whilst the charades game can be considered as a generic system, the creation of real-time holograms is not heavily developed. Many examples of video footage which can be used with the Pepper's Ghost Pyramid are readily available online, however there are far fewer implementations that do real time manipulation for this purpose.

In addition to the project being interesting, it is designed with outreach events in mind therefore giving it value as a potential teaching aide. Pepper's Ghost is an excellent example for children to learn how basic physics and computer science can be used to create an interesting and engaging visual display. In addition, the charades game and real-time hologram system will help to further engage the audience with technique and visualise an impactful application of the technique.

1.3 Background

1.3.1 Pepper's Ghost Pyramid

The Pepper's Ghost technique was originally used for stage and theatre productions in the Victorian era to display holographic illusions to the audience. The technique, discovered by Dr. Henry Pepper, was first used in theatre in the 1860s [1] and was used primarily to create a ghost-like illusion.

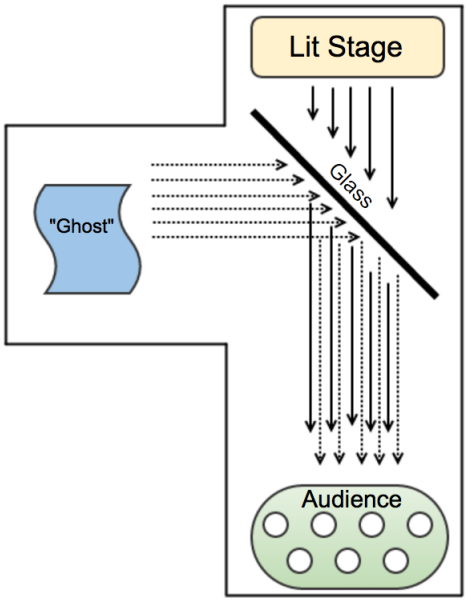


Figure 1.1: A basic diagram to display the Pepper's Ghost technique's use in theatre. In the diagram solid lines represent light from the lit stage and dashed lines as light from the "Ghost". The image has been taken from Brianne Costa's Blog on the website Comsol [1].

Figure 1.1 shows a basic example of how the technique produces holographic illusions. The lit stage that is parallel to the viewers line of sight will have actors and scenery as per a normal stage production. The "Ghost" is located off stage and is not visible by the audience. When illuminated, the ghost is displayed to the audience as the light rays travel radially from the "Ghost" and those that reach the glass will be refracted. The glass is positioned at 45° from both the audience and the "Ghost" to account for the 90° offset between the two.

This technique has been modernised since the 1860s, and has been used in different settings proving particularly popular in museums and exhibitions such as *An Audience with Sir Alex Ferguson* at the Manchester United Football Club museum [2] and *Shane Warne - 'Cricket Found Me'* at the national sports museum in Melbourne [3]. Within the last few years, there has been a resurgence in the use of the technique with consumers creating there own Pepper's Ghost illusions on mobile devices.

Whilst the technique still uses Henry Pepper's original concept, it has been modified to display the hologram from multiple angles using a pyramid - rather than a single glass plane. This structure is commonly referred to as the Pepper's Ghost pyramid.

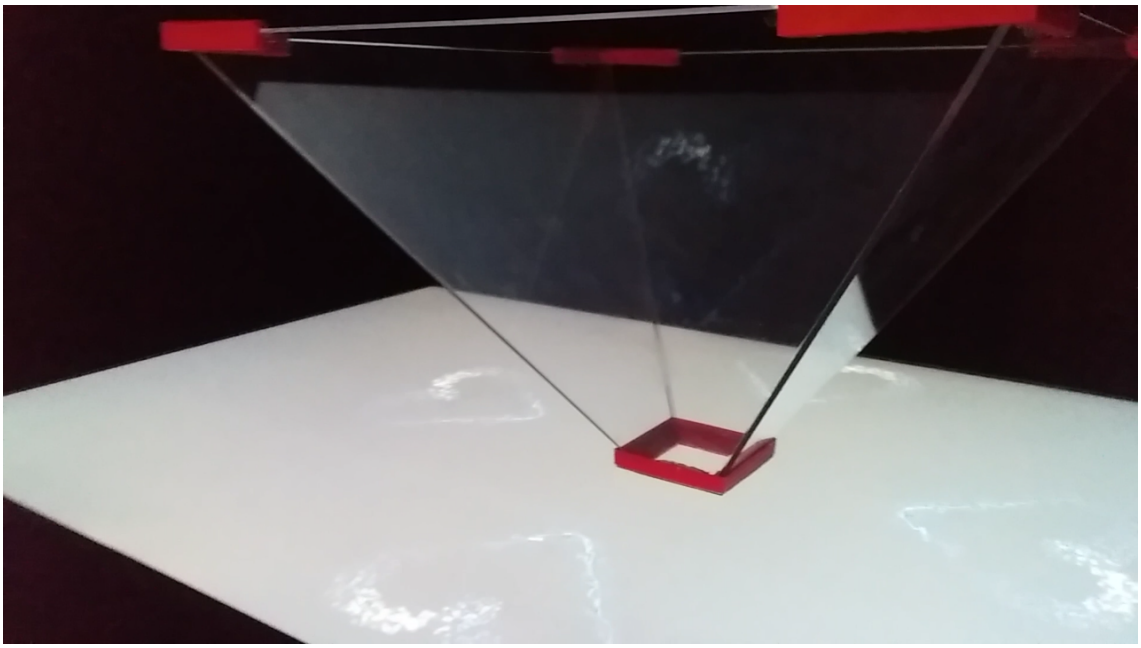


Figure 1.2: An image of modern day Pepper's Ghost Pyramid made from clear acrylic owned by the Aberystwyth University Computer Science Department.

Figure 1.2 shows an example of a Pepper's Ghost pyramid. The pyramid is square based, made from a transparent material (such as perspex, clear acrylic or glass) and is open at both the top and bottom. The technique for displaying holograms differs with the "Ghost" now being an image or video displayed on a screen. The pyramid is located on a large horizontal monitor that is being used to display the hologram input images. Furthermore, to create an illusion from all sides of the pyramid, four images are required (one for each side of the pyramid). This means that the 2D projection of the image can be seen from any side of the pyramid. This Pyramid structure will be used for the hologram creation part of this project.

1.4 Analysis

Whilst there have been numerous examples of applications using the Pepper's Ghost Pyramid as a computer visualisation technique, none were found that detailed using this as part of a real-time system. Furthermore, whilst many have used the technique in an educational setting, it is more often the case that the content of the hologram is used as an educational aid, rather than focusing on the technique itself. Due to its simplicity and visual impact, the Pepper's Ghost technique is ideal to help younger children understand light and refraction.

1.4.1 Objectives

Initially, the project scope covered the creation of a system to produce real-time holograms using a video feed. However, considering the time-line of the project and the work involved, the addition of an accompanying system for the Charades game was added to the objectives. The objectives that were agreed upon are as follows:

- To produce a system that can be used for outreach events to show case the creation of holograms using the Pepper's Ghost Pyramid.
- The system should use real-time video to capture a member of the audience and then project these images as holograms using the Pepper's Ghost Technique.
- An accompanying system should be produced to allow for a charades style game to be played using the hologram system to display the actor to the viewers.
- Given the target audience, the system should be interactive and engaging whilst also enabling the viewers to learn about the technique and how it works.
- The system should be easy to use and for the most part self explanatory.
- As it is designed for use at outreach events, the system should be robust and thoroughly tested.
- The system should be easy to run from the perspective of the client. Initial set up should ideally be straight forward and self explanatory.

1.4.2 Problem decomposition

Following the agreement of the project objectives, it was possible to begin to decompose the requirements of the project into smaller sections. As the project is comprised of two systems (the real-time Pepper's Ghost creation system and the charades game), this seemed a natural divide for the project at the highest abstraction layer.

1.4.2.1 Real-time Pepper's Ghost

Following research into the Pepper's Ghost technique, it was decided that it would be feasible to construct a prototype of the hologram system for the 2017 Aberystwyth University Science Week event. The main tasks to produce the prototype system were:

- Analyse and produce a design for the display area (where the pyramid and monitor are used to produce a hologram).
- Analyse and produce a design for the staging area (where the actor is filmed).
- Design and implement a program capable of obtaining a live video feed from a web cam.
- Correctly orientate and position multiple copies of the video feed in a single display window.
- Scale the video feed is of the correct size for the output device.

The main consideration throughout the development process of the Pepper's Ghost application is the efficiency of the video feed manipulation. To have an effective display, the video should be running at no less than 30 frames per second. An ideal target would be 60 frames a second but this may not be achievable... depending on the capture speed of the camera.

1.4.2.2 Charades game

The charades game forms the more interactive side of the project. The concept is to have a single actor (member of the audience) outside the viewing area with a mobile device. He or she selects a phrase from a choice of phrases and then act out the selected phrase. The acting is captured by the web cam from Pepper's Ghost system and displayed as a hologram in the viewing area. Whilst the phrase is being acted, the viewers (who will also have mobile devices) are given information about the phrase (such as the number of words and the genre of the phrase) and attempt to guess what is being acted. Once the phrase has been successfully acted, the actor is be prompted to select a new phrase and the viewers are allocated points. After three phrases, the winner is the viewer with the most points. The main tasks of the charades game are:

- Create a user interface for both the actor and viewer.
- Allow the actor to select a phrase and, if the phrase is comprised of multiple words, a current word to act.
- Allow the viewer to attempt to guess the current word or phrase.
- Inform both users when the word or phrase has been correctly guessed.
- Prompt the actor for a new word or phrase for them to act.
- Establish and implement how and when the game ends.
- Inform users of the winner.
- Create a session system to only allow those who have logged on to the website with the correct details to access the information.

Whilst a mobile application seems ideal for use at localised events, a web app implementation would allow users to access the system from their own mobile devices regardless of platform. However, using a web app raises additional security issues that should be taken into consideration. To stop malicious interactions from those outside the event, the web app must ensure that only those at the event can access the system. To ensure this, the main task list included adding a session key to the system. This session key will be distributed at the event, and only those who know the session key are able to access the system. The system also provides the ability to change the session key easily when required.

1.4.3 Alternative implementations

Implementing an android application over a web app boasts the increased security as devices could potentially connect via Bluetooth or a peer-to-peer connection. The shorter connection range offered by these technologies, would mean that those outside the event would not be able to join session and corrupt the system. Furthermore, it would stop users from attempting to change the information in the URL for each page and therefore reduce the number of checks required to ensure that the system data has not been tainted.

Despite this, a web application done correctly is more accessible to users as it is platform independent. In addition, it would be more likely that users can access the system from their own devices, such as smart phones, meaning more users can play the game at the same time.

1.5 Process

After consideration of different methodologies, the decision was made that this project would follow the Feature Driven Development (FDD) plan driven methodology. FDD is normally considered for larger projects as it provides a framework for distributed development. By dividing developers into smaller teams, FDD allows those teams to tackle features one at a time in parallel. Furthermore, the up front planning stage is generally more suited to projects that are more stable as, whilst it can be adapted throughout the process, the overall model is normally only added to, and the core architecture remains static. FDD follows five steps throughout the development of a software system. The first three steps regarding planning and designing an overall model, and developing a list of all the features that must be considered for the system to meet the criteria of the end users. The remaining two step form an iterative process of designing and implementing features one by one.

The steps required to complete this project are well defined and therefore would be well suited to having an up front design. Furthermore, FDD encourages continuous integration (CI) which offered a way to produce a functional prototype at various stages of the project. CI was a significant aid for producing a function prototype for both the mid project review and the 2017 Aberystwyth University science week event. FDDs **Design by feature** stage offers an ideal step to perform spike work for technologies that developers are unfamiliar with. Moreover, it was still possible to follow the five step process of FDD development in this project.

1.5.1 Single Person FDD adaptation

To successfully use FDD for this project, several adaptations to the normal processes of the methodology were made. Many of these changes mirror those discussed in S. Khramthchenko's thesis, *"A Project Management Application for Feature Driven Development (FDDPMA)"* [8]. The most notable being the abolition of developer teams in favour of a single developer. This required the developer to assume multiple roles throughout the development process and at each different stage as specified below.

1.5.1.1 Develop an Overall Model

Initially, a Domain Expert (Customer) was required to aid in the development of the overall model and feature creation. In this context, the project supervisor (Dr. Helen Miles) fulfilled this role and the developer acted as both the Chief Programmer and Chief Architect. For this project, the Domain specific language was shared by both the Domain Expert and the developer.

1.5.1.2 Build a Feature list

The feature list was produced with the objectives of the system and the main tasks for the project in mind. This was finalised with a discussion between the developer and the Domain Expert which verified all the features. In addition to a description, each feature also had a complexity and priority rating, as well as a list of dependent features and whether the feature was require for the prototype. The final feature list can be found in Appendix C.

1.5.1.3 Plan by Feature

Steps such as establishing developer teams and scheduling developer teams' time throughout the project were no longer required. In their place, the features were given priorities to aid time scheduling and development order. Once the order was created, the features were assigned to separate week long iterations.

1.5.1.4 Design by Feature

Features were selected in dependency order. Once selected, features were exhaustively designed taking into consideration the functions required to fulfil the feature as well as how these functions should be tested. The project used GitHub for version control and issues were created which corresponded to a feature. The first action in an issue was to complete a design of the feature and update the overall design as required.

1.5.1.5 Implement by Feature

Features were implemented in the same GitHub issue as the design work. The project was developed using Test Driven Development (TDD) and, therefore, the test suite was updated before any code was implemented. Once the tests were created, an implementation was added which had to pass the tests to be acceptable. Whilst the tests could be run locally, there was also a Jenkins service for continuous integration to run the full test suite before it was merged with the master branch.

Chapter 2

Design

You should concentrate on the more important aspects of the design. It is essential that an overview is presented before going into detail. As well as describing the design adopted it must also explain what other designs were considered and why they were rejected.
The design should describe what you expected to do, and might also explain areas that you had to revise after some investigation.
Typically, for an object-oriented design, the discussion will focus on the choice of objects and classes and the allocation of methods to classes. The use made of reusable components should be described and their source referenced. Particularly important decisions concerning data structures usually affect the architecture of a system and so should be described here.
How much material you include on detailed design and implementation will depend very much on the nature of the project. It should not be padded out. Think about the significant aspects of your system. For example, describe the design of the user interface if it is a critical aspect of your system, or provide detail about methods and data structures that are not trivial. Do not spend time on long lists of trivial items and repetitive descriptions. If in doubt about what is appropriate, speak to your supervisor.
You should also identify any support tools that you used. You should discuss your choice of implementation tools - programming language, compilers, database management system, program development environment, etc.
Some example sub-sections may be as follows, but the specific sections are for you to define.

The design for the system was completed in two major steps. First, the initial design was created before any development and following this, incremental changes were made to add more detail to the design with every relevant feature. The overall initial design of the system remained mostly static and was only added to. The UML class diagrams were designed to start as very basic and evolved as the features were developed

2.1 Overall Architecture

2.1.1 Use case

To best identify the requirements of the users, the use case diagram (Figure 2.1) was produced. In addition to the functionality required by each type of user (Actors and Viewers) the diagram

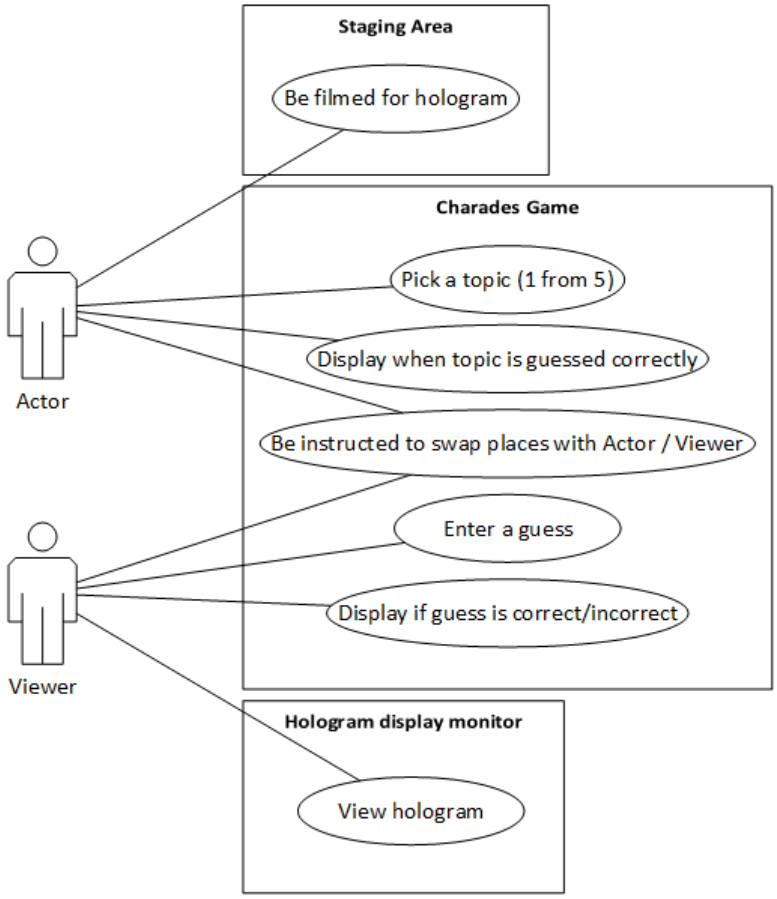


Figure 2.1: A use case diagram produced to show how the users will interact with the system and the functionality they require.

also divides the system into basic subsections. This diagram was a good first step in the early development of the system as it gave an overview of the system as a whole before any major implementation choices were required. Whilst very basic, it also showed interaction with the system that the users would require and acted as the base for the input and output requirements to be address in the user interface design. In addition, the use case diagram showed that the Charades game functionality was very dependant on the type of user which promoted an model-view-controller design pattern.

2.1.2 Activity diagram

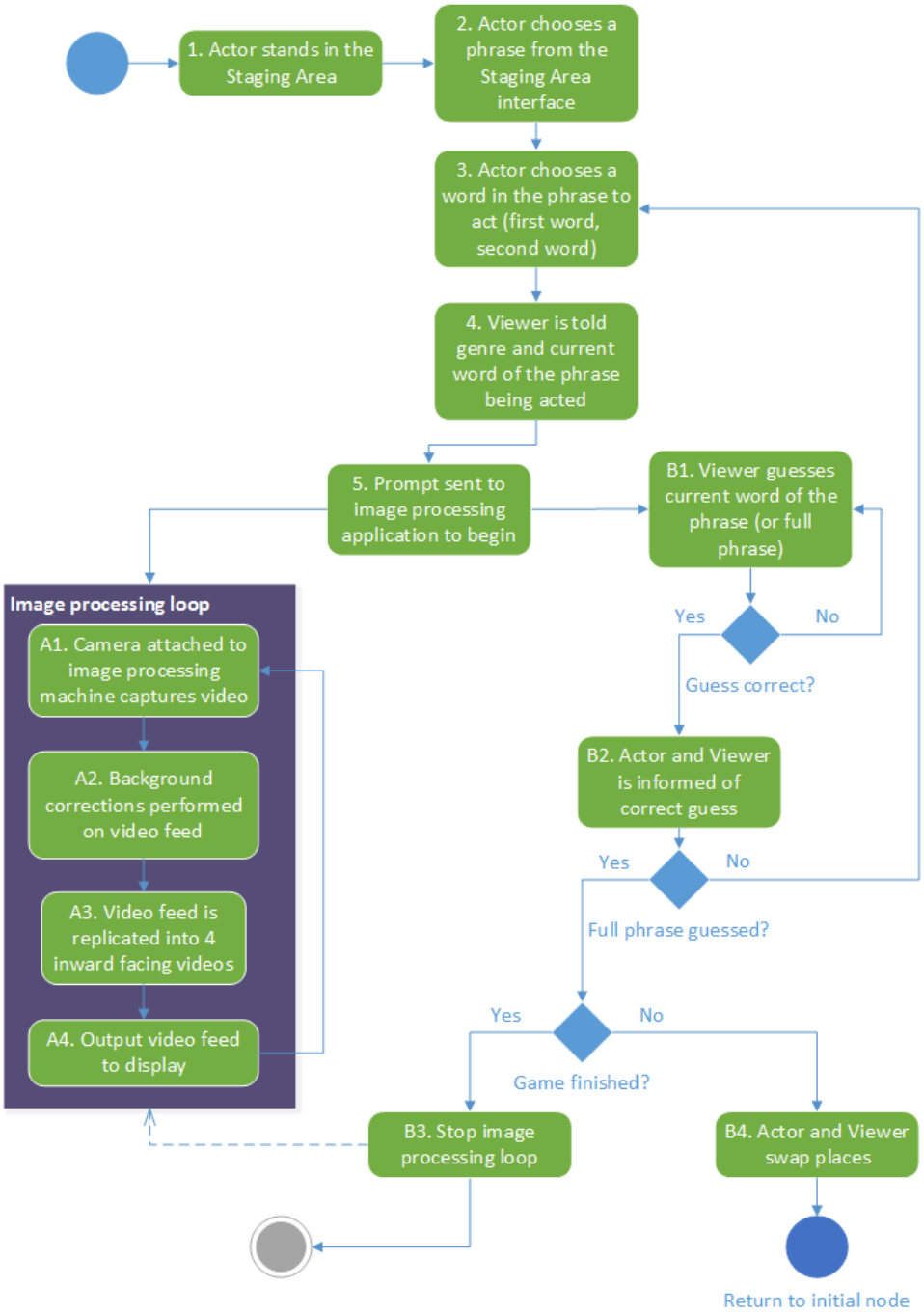


Figure 2.2: An activity diagram that describes the flow of information in the system. Includes the data flow for both the Hologram and Charades systems.

Given the system was deigned to be a game of charades, it follows a linear set of instructions that mirror the original rules of the charades game. An activity diagram was chosen to model the flow of activities in the system due to the well defined path of game play. Figure 2.2 shows an activity diagram that maps the basic flow of a single round of the game. The *Image processing*

loop, shown in purple, describes the flow of the hologram creation. This process (prefixed with the letter A) run continuously in parallel with the game loop (prefixed with the letter B).

2.1.3 Component diagram

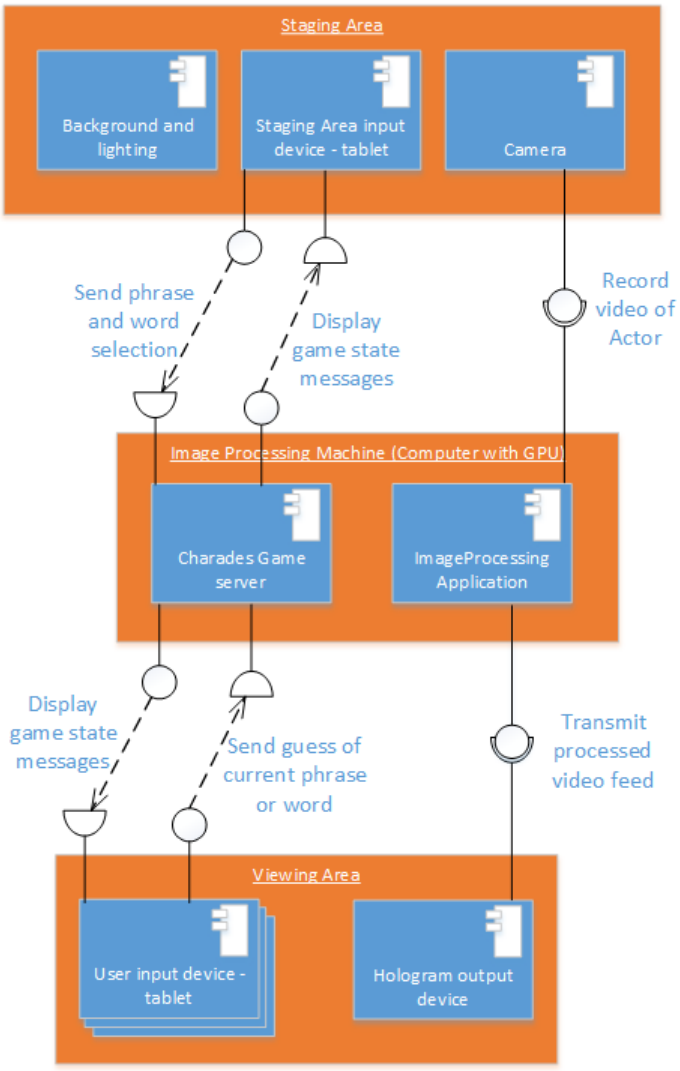


Figure 2.3: A component diagram that describes both the software are hardware components of the system and how they interact with each other.

The system, in its totality, had to include multiple pieces of hardware to operate. The component diagram, shown in Figure 2.3, helped to combine the different hardware considerations in tandem with the software. Furthermore, the diagram confirmed the presence of three separate areas of the project.

The **Staging Area** would be where the actor would be filmed. The camera directly connects to the ImageProcessing machine via a usb cable for data transfer. The Staging Area input device will use a wireless connection to send and receive messages from the Charades game server that is hosted on the image processing machine.

The **Image Processing machine** is a computer that acts as the server for the website as well as running the image processing application for the holograms. Due to the nature of the video feed processing in real-time, the computer will require a Graphical processing unit (GPU). The Image Processing application has input of the raw video feed from the camera in the Staging Area and will process the video and output the result. The Charades Game server is hosted on the Image processing machine. This component will handle the flow between the users in the Viewing Area and the Actor in the Staging Area.

2.2 Some detailed design

2.2.1 Even more detail

2.3 User Interface

2.4 Other relevant sections

Chapter 3

Implementation

The implementation should look at any issues you encountered as you tried to implement your design. During the work, you might have found that elements of your design were unnecessary or overly complex; perhaps third party libraries were available that simplified some of the functions that you intended to implement. If things were easier in some areas, then how did you adapt your project to take account of your findings?

It is more likely that things were more complex than you first thought. In particular, were there any problems or difficulties that you found during implementation that you had to address? Did such problems simply delay you or were they more significant?

You can conclude this section by reviewing the end of the implementation stage against the planned requirements.

Chapter 4

Testing

Detailed descriptions of every test case are definitely not what is required here. What is important is to show that you adopted a sensible strategy that was, in principle, capable of testing the system adequately even if you did not have the time to test the system fully.

Provide information in the body of your report and the appendix to explain the testing that has been performed. How does this testing address the requirements and design for the project?

How comprehensive is the testing within the constraints of the project? Are you testing the normal working behaviour? Are you testing the exceptional behaviour, e.g. error conditions? Are you testing security issues if they are relevant for your project?

Have you tested your system on "real users"? For example, if your system is supposed to solve a problem for a business, then it would be appropriate to present your approach to involve the users in the testing process and to record the results that you obtained. Depending on the level of detail, it is likely that you would put any detailed results in an appendix.

The following sections indicate some areas you might include. Other sections may be more appropriate to your project.

4.1 Overall Approach to Testing

The project used TDD as part of the process for software implementation. As well as designing features, tests were also designed for the required functions before functional code was written. There were two main types of tests for the website: unit tests that test the functionality of the model and the controllers and system tests that use selenium to check the flow of the system and simulate the actions of a user. Where appropriate, tests for functions were written to test three different cases:

- Success case: This would simulate either an expected input parameter for the function or an expected state that the system should be in in order to run it.
- Edge case: This would check the maximum functioning range of the function. For example if the function was only designed to allow 5 viewers to participate in a game, then it would check that the when the viewers total 5 the function still passes.

- Failure case: This would ensure that given incorrect parameters or system state then the function would fail in the expected way.

For unit tests, tests were organised into classes were each test class would test a class or module of code. Similarly, system tests were designed to test a single Django template. The requirement for adding code to the master branch is that all unit and system tests (past and present) must pass and the static analysis tools must produce no errors.

4.2 Automated Testing

4.2.1 Jenkins

To improve the consistency of the code base, the GitHub repository was linked to a local Jenkins server. This server was set up at the beginning of the project and was used as part of the development life cycle of each feature. The Jenkins server ran as a background process on the development machine. The set up for the Jenkins server followed two guides: The official Jenkins Windows Service installation guide *"Installing Jenkins as a Windows Service"* [4] from the Jenkins.io wiki and Steve's Blog *"Automated python unit testing, code coverage and code quality analysis with Jenkins"* [5] a self hosted technical blog from a software developing consultant specialising in Python.

Whilst Jenkins offers the ability to run tests periodically, this feature was not used and tests were triggered manually by the developer. The reasoning behind this is that incremental time based builds were not required. Had the project been developed by a team, the need for timed builds is more justifiable as many developers will be pushing code changes simultaneously. In a single person developer team, code is pushed one feature at a time and the work flow is linear hence there is no justifiable requirement for periodical builds. Despite this, the use of Jenkins is still admissible in a single developer project as it allows for centralised building of the project in a controlled, repeatable, environment. Additionally, The formatting for results produced by Jenkins offers an easy way to quickly visualise problems with the code base.

4.2.2 Static analysis

4.2.2.1 Pylint

Pylint is a linting tool that follows the PEP8 standards [6] for python coding. The linter runs static analysis on the code base and ensures that all code meets with the standards specified by PEP8 (and produces warning and errors were this is not the case. The PEP8 standards include guidelines style and syntax as well as documentation (in the form of doc strings). Whilst several standards for the Python coding language are available, PEP8 was chosen as it is promoted by the developers of the Python language and is the most popular. An example of the Pylint output rendered by Jenkins is displayed below in Figure 4.1.

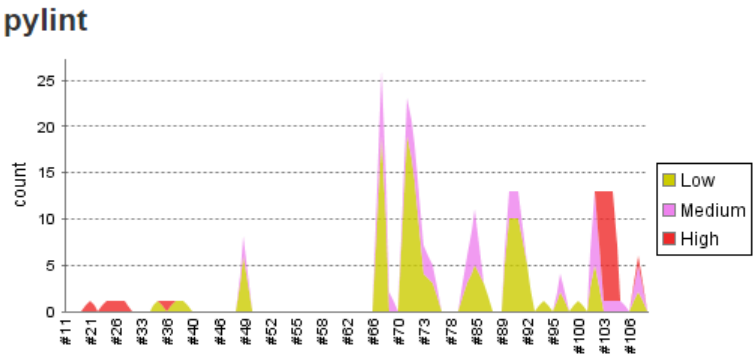


Figure 4.1: The output of the Pylint test run on the full code base rendered by Jenkins. Here, the number of warnings, and there severity, is logged by each build number. The peaks correspond to the first time a feature is pushed, these problems are then normally resolved in the troughs. In order to be able to merge a feature, the Pylint graph must be at zero warnings.

4.2.2.2 Test coverage

An additional testing package, nosetest [7], was used to run python tests for this project. This package, amongst other advantages, produces reports that are interpretable by Jenkins for both unit tests and test coverage. As the web based charades game was written in Django (which has its own testing framework), noetest could only be used for the hologram creation software. The test coverage displays graphically the proportion of the code that is run in tests. Although basic, this helps to highlight potential conditional statements, function and even lines that are not being tested. An example of the test coverage output rendered by Jenkins is displayed below in Figure 4.2. Figure 4.2 shows that whilst the majority of the code base is covered by testing, several lines in the python.vpa package are excludes. These lines had to be omitted from tests as their functionality was to the perform the video processing loop. Whilst the functions that are called within that loop are sufficiently tested, the loop itself can not be tested easily as it creates an output window. The output window is only successfully closed via mouse click on the window close button (an operation not easily replicated in tests).

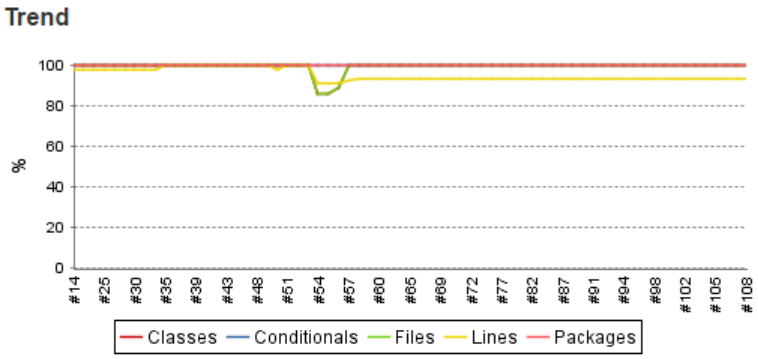
4.2.3 Unit Tests

To ensure the functionality of the system, unit tests were created to test functions.

4.2.4 User Interface Testing

Selenium was used for all the testing of the user interface. The selenium library allows for interaction with a web browser in a testing environment. All user interface tests have several generic tests which are as follows:

- Page elements: The generic view of a page is tested to ensure that the elements on the page are what the test expects. This is mainly used as a test of the Django template code, written in HTML, and the embedded Python.



Project Coverage summary

Name	Packages	Files	
Cobertura Coverage Report	100% 2/2	100% 8/8	100%

Coverage Breakdown by Package

Name	Files	Classes
python	N/A	N/A
python.tests	100% 4/4	100% 4/4
python.vpa	100% 4/4	100% 4/4

Figure 4.2: The rendered output of the test coverage report from nosetest in Jenkins. This report includes both a graphical and table representation of the data showing the amount of code that is being exercised by the unit tests. Note the incomplete test coverage and see explanation above.

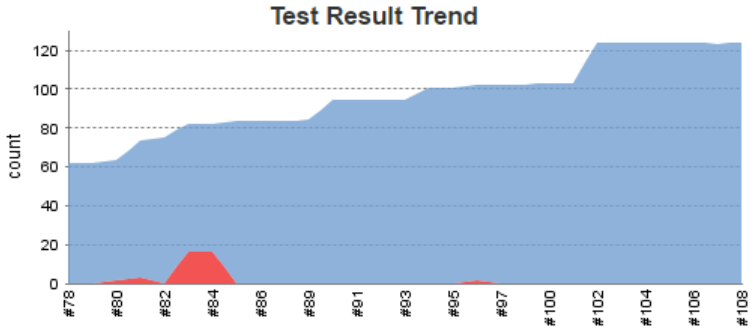


Figure 4.3: The rendered output of the unit test report in Jenkins. Blue indicates passing tests while red is those that failed.

- Navigation: Most pages of the website will either have a button to take the user to another page, or poll the api until the conditions are met to redirect the user. In both cases, a test case will exist to ensure the page transition is correct.

4.2.4.1 Selenium problems and solutions

Selenium gives the advantage of being able to test a web interface. However, when run locally, selenium took on average 6 to 8 seconds to create a new instance of the web browser. Many tutorials for using the Selenium testing framework

4.2.5 Stress Testing

4.2.6 Other types of testing

4.3 Integration Testing

4.4 User Testing

User testing took place at the Aberystwyth Science week event in March 2017. The event lasted for three days and gave the opportunity to test the prototype of the hologram system on the target audience. A black tent was set up in the event hall and this housed the touch screen table being used as an external monitor for the holographic display. The web cam for the video feed input was set up outside the tent and used to record viewers.

Chapter 5

Evaluation

Examiners expect to find in your dissertation a section addressing such questions as:

- Were the requirements correctly identified?
- Were the design decisions correct?
- Could a more suitable set of tools have been chosen?
- How well did the software meet the needs of those who were expecting to use it?
- How well were any other project aims achieved?
- If you were starting again, what would you do differently?

Other questions can be addressed as appropriate for a project.

Such material is regarded as an important part of the dissertation; it should demonstrate that you are capable not only of carrying out a piece of work but also of thinking critically about how you did it and how you might have done it better. This is seen as an important part of an honours degree.

There will be good things and room for improvement with any project. As you write this section, identify and discuss the parts of the work that went well and also consider ways in which the work could be improved.

In the latter stages of the module, we will discuss the evaluation. That will probably be around week 9, although that differs each year.

Appendices

The appendices are for additional content that is useful to support the discussion in the report. It is material that is not necessarily needed in the body of the report, but its inclusion in the appendices makes it easy to access.

For example, if you have developed a Design Specification document as part of a plan-driven approach for the project, then it would be appropriate to include that document as an appendix. In the body of your report you would highlight the most interesting aspects of the design, referring your reader to the full specification for further detail.

If you have taken an agile approach to developing the project, then you may be less likely to have developed a full requirements specification. Perhaps you use stories to keep track of the functionality and the 'future conversations'. It might not be relevant to include all of those in the body of your report. Instead, you might include those in an appendix.

There is a balance to be struck between what is relevant to include in the body of your report and whether additional supporting evidence is appropriate in the appendices. Speak to your supervisor or the module coordinator if you have questions about this.

Appendix A

Third-Party Code and Libraries

If you have made use of any third party code or software libraries, i.e. any code that you have not designed and written yourself, then you must include this appendix.

As has been said in lectures, it is acceptable and likely that you will make use of third-party code and software libraries. If third party code or libraries are used, your work will build on that to produce notable new work. The key requirement is that we understand what is your original work and what work is based on that of other people.

Therefore, you need to clearly state what you have used and where the original material can be found. Also, if you have made any changes to the original versions, you must explain what you have changed.

As an example, you might include a definition such as:

Apache POI library - The project has been used to read and write Microsoft Excel files (XLS) as part of the interaction with the client's existing system for processing data. Version 3.10-FINAL was used. The library is open source and it is available from the Apache Software Foundation [**?**]. The library is released using the Apache License [**?**]. This library was used without modification.

Appendix B

Ethics Submission

This appendix includes a copy of the ethics submission for the project. After you have completed your Ethics submission, you will receive a PDF with a summary of the comments. That document should be embedded in this report, either as images, an embedded PDF or as copied text. The content should also include the Ethics Application Number that you receive.

Appendix C

Code Examples

For some projects, it might be relevant to include some code extracts in an appendix. You are not expected to put all of your code here - the correct place for all of your code is in the technical submission that is made in addition to the Final Report. However, if there are some notable aspects of the code that you discuss, including that in an appendix might be useful to make it easier for your readers to access.

As a general guide, if you are discussing short extracts of code then you are advised to include such code in the body of the report. If there is a longer extract that is relevant, then you might include it as shown in the following section.

Only include code in the appendix if that code is discussed and referred to in the body of the report.

3.1 Random Number Generator

The Bayes Durham Shuffle ensures that the psuedo random numbers used in the simulation are further shuffled, ensuring minimal correlation between subsequent random outputs [**?**].

```
#define IM1 2147483563
#define IM2 2147483399
#define AM (1.0/IM1)
#define IMM1 (IM1-1)
#define IA1 40014
#define IA2 40692
#define IQ1 53668
#define IQ2 52774
#define IR1 12211
#define IR2 3791
#define NTAB 32
#define NDIV (1+IMM1/NTAB)
#define EPS 1.2e-7
#define RNMX (1.0 - EPS)
```

```
double ran2(long *idum)
{
  /*-------------------------------------------------*/
  /* Minimum Standard Random Number Generator        */
  /* Taken from Numerical recipies in C              */
  /* Based on Park and Miller with Bays Durham Shuffle */
  /* Coupled Schrage methods for extra periodicity   */
  /* Always call with negative number to initialise  */
  /*-------------------------------------------------*/

  int j;
  long k;
  static long idum2=123456789;
  static long iy=0;
  static long iv[NTAB];
  double temp;

  if (*idum <=0)
  {
    if (-(*idum) < 1)
    {
      *idum = 1;
    }else
    {
      *idum = -(*idum);
    }
    idum2=(*idum);
    for (j=NTAB+7;j>=0;j--)
    {
      k = (*idum)/IQ1;
      *idum = IA1 *(*idum-k*IQ1) - IR1*k;
      if (*idum < 0)
      {
        *idum += IM1;
      }
      if (j < NTAB)
      {
        iv[j] = *idum;
      }
    }
    iy = iv[0];
  }
  k = (*idum)/IQ1;
  *idum = IA1*(*idum-k*IQ1) - IR1*k;
  if (*idum < 0)
  {
    *idum += IM1;
  }
```

```
  k = (idum2)/IQ2;
  idum2 = IA2*(idum2-k*IQ2) - IR2*k;
  if (idum2 < 0)
  {
    idum2 += IM2;
  }
  j = iy/NDIV;
  iy=iv[j] - idum2;
  iv[j] = *idum;
  if (iy < 1)
  {
    iy += IMM1;
  }
  if ((temp=AM*iy) > RNMX)
  {
    return RNMX;
  }else
  {
    return temp;
  }
}
```

Annotated Bibliography

[1] Brianne Costa, "Explaining the Peppers Ghost Illusion with Ray Optics," https://www.comsol.nl/blogs/explaining-the-peppers-ghost-illusion-with-ray-optics/, 2016.

Comsol blog that describes the Peppers Ghost Pyramid implementation used for this project.The blog details in brief how the technique works and explains this with ray tracing. Furthermore, the blog mentions the history of the technique.

[2] Manchester United Ltd., "Meet Sir Alex - the hologram," http://www.manutd.com/en/News-And-Features/Club-News/2007/Dec/Meet-Sir-Alex--the-hologram.aspx, 2007.

News bulletin announcing new hologram of Sir Alex Furgeson in Machester United Museum.

[3] The National Museum, Melbourne, "Shane Warne - 'Cricket Found Me'," http://www.nsm.org.au/Exhibitions/Shane%20Warne%20Hologram.aspx.

Exhibition details of a hologram of Shane Warne discussing his career in Cricket.

[4] e. a. Erik Ramfelt, "Installing Jenkins as a Windows service," https://wiki.jenkins-ci.org/display/JENKINS/Installing+Jenkins+as+a+Windows+service, 2013.

The official installion guide for installing Jenkins as a service on a Windows machine. The guide is hosted and written by the developers of the jenkins continuous integration platform.

[5] Steve, "Automated python unit testing, code coverage and code quality analysis with Jenkins," http://bhfsteve.blogspot.co.uk/2012/04/automated-python-unit-testing-code_27.html, 2012.

A technical blog produced by a software development consultant who specialises in Python. This particular blog includes details of setting up static analysis tests for Python and Jenkins. The blog is written for Linux (Debian) but has been adapted to aide in the set up of a Jenkins service for Windows.

[6] ——, "PEP 8 – Style Guide for Python Code," https://www.python.org/dev/peps/pep-0008/, 2012.

The contents page for the PEP 8 style guide for python. This is used by the Pylint linting tool to ensure code conforms to standards.

[7] J. Pellerin, "PEP 8 – Style Guide for Python Code," http://nose.readthedocs.io/en/latest/, 2015.

A Python unit test exension that provides additional functionality and reporting to Python unit tests.

[8] S. Khramthchenko, "A project management application for feature driven development (fddpma)," http://fddpma.sourceforge.net/help/fddpma_thesis.pdf, 2005, accessed August 2011.

Details the authors adapted FDD methodology used when creating a project management application for FDD for a thesis at Harvard university. Not only does the author discuss the methodology he followed, (Chapter 2) raise good points for consideration this projects methodology.