Module 2: Critical Thinking

Redesigning the Waterfall Model

Nolan Byrnes

CSC505 – Principles of Software Development

Colorado State University – Global Campus

Professor Steven Evans

July 31st, 2022

Module 2: Critical Thinking

The waterfall model is the oldest paradigm that is used in software engineering, which follows a sequential model through the software development life cycle (Pressman, 2020, pg. 25-26). This paradigm is great for projects where the requirements are well understood but comes with some challenges. Customers are unable to see a working version of the program until the end of the project and feedback is given at the deployment phase. To follow the simplicity of having a linear model and combat some of the issues with the waterfall model, I have come up with the Byrnes Model, which is my own adaptation of the waterfall model. This paper explains how the Byrnes Model differs from the Waterfall model, how it overcomes some of the issues that arise when strictly following the waterfall model, and the Byrnes model class that I developed in python.

**Byrnes Model**

The waterfall model includes the communication, planning, modeling, construction, and deployment phases in order, as displayed in Figure 1 below.

Figure 1.

Waterfall Model

Diagram

Description automatically generated

Note. Diagram of the Waterfall model. From Software engineering: A practitioner's approach (pg. 26), by R. S. Pressman et. al., 2019, McGraw-Hill Higher Education.

In the Byrnes Model, I have included the “Modeling Prototypes”, “Prototyping”, and “Stakeholder Feedback” phases between the Communication and Planning phases of the waterfall model as shown in Figure 2 below.

Figure 2

Byrnes Model

A picture containing diagram

Description automatically generated

Note. Diagram of the Byrnes Model. Grey boxes indicate the new steps that are added to the Waterfall model. Orange boxes represent the phases from the original Waterfall model.

The first phase of the Byrnes model is the same as in the Waterfall model, where the project begins, and requirements gathering is performed. In the “Modeling Prototypes” phase, we have the development team perform analysis on the initial requirements gathering and design a couple prototypes based on the requirements that were given. After developing a couple quick prototypes in the Prototyping phase, we then demonstrate the prototypes to the stakeholders and receive their feedback in the “Stakeholder Feedback” phase.

The Modeling Prototypes and Prototyping phases of the Byrnes Model is where the developers can take the requirements that were given and develop a couple prototypes of how they envision the final solution will be.

The prototyping that is performed takes elements from the Rapid Throwaway Prototype model, “which is based on the preliminary requirement” and “quickly developed to show how the requirement will look visually” (Martin, 2022, para. 9). This helps to identify what approaches will work and could help identify any issues with some of the requirements that were given.

After the prototyping is completed, the Byrnes Model proceeds to the Stakeholder Feedback phase. This is where the prototypes are demonstrated to the customer, and the development team receives their feedback on the prototypes. This phase allows the stakeholders to see how the final product could look, identify what they like and not like about each prototype, and discuss what the final solution should be with the development team. After the Stakeholder Feedback phase, the Byrnes Model follows the rest of the Waterfall model, where the developer’s model, construct, and deploy the final solution.

By including the modeling prototypes, prototyping, and stakeholder feedback phase in the Byrnes model, some of the issues of the Waterfall model are addressed. Stakeholders can see a couple prototypes of the solution early in the process and provide their feedback before the final solution is developed. Also, any issues with the design or requirements can be addressed early in the process, whereas with the standard Waterfall model, they may not be detected until feedback is given in the deployment phase. I think the Byrnes model would work best for projects where the requirements are clearly defined, but unsure on how the final product would look.

**Byrnes Model Implementation in Python**

I implemented the Byrnes Model as a class in python that can prompt an end user to the phases of the Byrnes model. In the constructor of the class, I have the names of the phases in a list, the descriptions of the phases in a dictionary, and the current phase number starting at zero.

The public method “nav\_through\_phases()” is the main method that will navigate the user through all the phases of the Byrnes model. It begins by printing a welcome message and calculating the total “phase\_count” by taking the length of the “phases” property of the object. It then begins a loop where it will call the private method “\_\_print\_phase()”, and then ask the user to input “n” to continue to the next phase. After the end user inputs “n”, the “current\_phase\_num” increases by one, and it will continue the next iteration of the loop, until the “current\_phase\_num” property matches the “phase\_count”.

The “\_\_print\_phase” is a private method that uses the current phase number of the object as the index of the “phases” property, which gives the name of the current phase that it is on. It will print the current phase that it is on, and then determine the tasks that are included in that phase by finding the key value pair of the “phase\_description” property, where the key would be the current phase name, and the value would be a list of tasks to be performed within that phase. A loop is performed on the list of tasks and prints out each task that needs to be performed for the phase it is currently on.

Since “\_\_print\_phase()” is a private method, it “cannot be accessed outside the class that it is declared in” (Esplanada, 2021, para. 4). I made the decision to make this method private because the method was not made for the user of the class to call directly, because when using the Byrnes model, all the phases should be performed in order, and not just with one of the phases. By doing so, it also helps simplify the code for the public method “nav\_through\_phases()”, making it easier for the developer to read and edit the output of the class.

**Conclusion**

The Byrnes model is an adaptation of the Waterfall model by including phases that require prototyping of the solution and receiving feedback. It addresses some of the challenges that come by strictly following the Waterfall model by allowing the stakeholders to see how the final solution could look like, allows the developers to receive feedback on the progress they have made earlier on in the software development life cycle, and helps to highlight any issues with the requirements or design of the solution can be highlighted by creating a couple prototypes early on. The Byrnes class I developed using Python uses a list to keep all the phases in order, and a dictionary with a list of action items for each of the phases. I used a public method called “nav\_through\_phases()” to help the user navigate through the phases of the Byrnes model, and the private method “\_\_print\_phase()” to print out the details of the phases.

**REFERENCES**

Esplanada, R. (2021, May 18). Private methods in Python. Delft Stack. <https://www.delftstack.com/howto/python/python-private-method/>

Martin, M. (2022, July 9). Prototyping model in software engineering: Methodology, process, approach. Guru99. <https://www.guru99.com/software-engineering-prototyping-model.html>

Pressman, R. S., & Maxim, B. R. (2019). Software engineering: A practitioner's approach. McGraw-Hill Higher Education.