Deeksha Juneja

**Proposal for Developing a Website on Design Patterns**

1. **Introduction**
   1. **Purpose**

This document proposes the development of new study material on design patterns for software development. Understanding design patterns is necessary for students in computer science, software engineering, and computer engineering. These students will have plenty of class projects that would require knowledge of design patterns. Moreover, the knowledge of the design patterns will also be used when the students go for internships and jobs. It is evident from the discussions in software classes like COMS 309, COMS 319 and COMS 339 at Iowa State University that it is imperative for students to know how to design, implement and reuse the design patterns. The current material – classroom lectures, books, web articles, and youtube videos are not sufficient help for the students. These resources are not comprehensive enough to help students master design patterns. Each of them lacks in one area or the other. Some of them are very good at explaining what design patterns are, other documents relate them well to the object-oriented techniques, some give simple and easy-to-understand examples and some use friendly language. But none of them combines all these characteristics. Additionally, a problem with the currently available materials is that, either they provide very basic information about design patterns or get extremely technical. Hence, students struggle with moving from very easy material to very difficult material. They don’t have any transition material available for them.

The new teaching material proposed will help students understand design patterns, how to break down a problem into the templates of given design pattern and how to implement the design pattern. It will bridge the gap between beginner level material and advanced level material. It will have the best of all the currently available materials and improve it further, to help the students better understand the design patterns.

* 1. **Background**

The design patterns are templates or re-usable solutions for commonly occurring software problems. They are language independent and hence can be used in C++, Java, Python, PHP, etc. The design patterns are divided into three classes namely - structural, creational, and behavioral. Structural patterns deal with “relation between entities”, creational patterns concern themselves with “instantiation mechanisms” and behavioral patterns work with “communication between entities” (Bauisa, 2010, p.1). These sub-classes contain various design patterns. Each design pattern can further be divided into the intent, problem, structure, checklist, and discussion. These are the specifications of the design pattern and helps the user understand the use or the purpose of the design pattern. Each design pattern has a diagram to help explain it (Shvets, p. 1-2). The best feature of design patterns is that a piece of code could be easily removed or replaced without affecting the rest of the program. Hence, the programs can be easily changed without causing a catastrophe.

Most students are required to implement design patterns in their class projects and assignments as it is a part of good programming practice. Good programming practices are encouraged by professors so that it is easy for students to maintain those when they get hired. No company wants to employ a disorganized coder. Especially, if companies have had messy coders before, they more often than not require someone to organize their code and write better code for them. This is where design patterns and refactoring comes into play. Fowler’s book on refactoring is one of the most famous books to improve the haphazard code. The first part of COMS 339 focusses on refactoring of code. The professor goes in-depth on the examples given in the book, he especially focuses on Movie Rental example. He then goes on to discuss design patterns by the middle of the semester. It is very normal for students to make mistakes while applying design patterns and design patterns give disastrous results if executed in the wrong way. Hence, they seem hard for students to implement.

Generally, when students are studying design patterns in class, they seem very easy to them. Most students refer to online websites along with lectures. Some of the websites do a very good job of explaining java concepts of inheritance, polymorphism and encapsulation before they dive into design patterns. This helps the student relate earlier concepts they learned to begin coding with the design patterns as done in the youtube videos by Derek Banas. Some of these websites even do a great job of explaining design patterns to non-technical people. An excellent example of this is on codeproject.com where there is an article called “How I explained Design Patterns to my wife: Part 1” by Shubho. It is a conversation between Shubho and his wife, where he first related design patterns to object-oriented design principles. He also goes into the history of design patterns and explains how they were created. The software problems occurred; and then based on the requirement, some design solutions were evolved. Later, when it was noticed that these were some of the commonly occurring software problems, they were standardized into patterns. Each problem can potentially be broken down into a design pattern and then solved step by step with the help of design patterns (2010, p. 1-5). However, the problem with the youtube videos are that concentrate on the very basic stuff for a very long time. The students easily get distracted and find it unhelpful. On the other hand authors like Shubho try to help the beginners but fail to help them actually implement the design patterns. Their examples are so basic that the students don’t end up learning much. Additionally, many other websites across the internet lack in their approach by either becoming complicated, fast-paced, not directly addressing the problem in hand, being theoretical, not giving real life software problems, or not extending problems to bigger problems.

Furthermore, books like Head First Design Patterns by Freeman and Robson and Design Patterns Explained by Shalloway & Trott recommended by professors, take the students only so far because they leave a lot of loose ends. For instance, they start on examples and then leave them in between. Additionally in the process of problem solving, they skip a lot of steps which confuse the students and sometimes complicate the easiest of the things. Even university lectures with handouts don’t do a perfect job of explaining design patterns. They focus on the practical implementation, but without the theoretical base they can only take the practical so far. The students are more often than not lost half way through the practical implementation because of the lack of theoretical knowledge on the subject. Besides, they are not writing the code with the professor most of the time and hence have difficulty keeping up with the professor.

One of the potential solutions would be to read as many articles and books as possible as each one will have some different examples or strategies to help you understand design patterns. However, this would be very time-consuming and will not guarantee results. The difficultly with most of these books, websites, or classroom lectures is that they focus on one particular example for a particular design pattern. So, mostly all the websites or books will have the same example to explain that particular design pattern. As most students must be aware that one of the most famous examples is the cake decoration with the decorator pattern or employee manager relation with observer pattern. It is easy to understand how these can be implemented. But extending the decorator example to planning a party would not be something that is easily visible to most students. They might think that template method could be used for the party planning problem. This would lead to disastrous results and hence, will result alienating them from design patterns. Most students try to avoid using design patterns due to their unpleasant experience with them.

It is important for students to know how to design a pattern, use it and then reuse it; and for that, it is vital for students to understand design patterns. If they don’t know and don’t understand the design patterns by heart then they will waste a lot of time in going over each and every design pattern every time they have a new project which is not feasible.

* 1. **Scope**

The proposal addresses the needs of the students to better understand design patterns. It is limited to covering only design patterns. It will not cover the basic concepts of object-oriented programming though it may touch on how these concepts extend to design patterns. This document will cover how to understand design patterns using examples and will explain how to extend those examples to different problems. It will also talk about how to make a template for the described problem and step by step solutions for the problem. The documents proposes a website to accomplish the above mentioned tasks.

1. **Discussion**
   1. **Approach**

The students today need a comprehensive solution of examples that extend design patterns into real software problems. To accomplish this, a website needs to be developed for design patterns only. It would first cover the theoretical knowledge required by students to understand the origins of design patterns. The theory will go over the technicalities of each design pattern and will cover some basic examples; which students will be encouraged to understand properly. The theory will explain each design pattern in simple language and clarify any technical terms used. Each section of theory will be followed by a small quiz on the theory material. When the students feel comfortable with the theoretical knowledge, the next step will be to have exercise problems on design patterns. Students will be given a software problem and will be required to breakdown the problem. They would then have to identify which design pattern will fit the problem. Thereafter, they should work on implementing the pattern step by step and successfully solving the problem. A step by step solution will also be provided to the students if they encounter trouble. The students will also have access to a help forum on the website. Students will work on multiple problems with increasing difficulty levels. The solution proposed here blends the practical implementation with theoretical knowledge.

* 1. **Result**

The website will reduce the gap between beginner level knowledge and high level implementation. It will also walk the students through beginner level learning; to medium level implementation; and finally to high level coding. Therefore, the website will overcome the flaws currently present in the material available on design patterns. Also, computer and software Engineers are more prone to learning from the web than from textbooks. Hence, this will provide them with a platform which does not have intensive reading but intensive implementation. The reading will be just enough to inform them; but implementation of problems will glue to them to the website. It will be a one stop destination for all information on design patterns. The aim of the website will be to make the students not only learn design patterns but also like them. This is exactly what will set the website apart from the currently available material.

* 1. **Statement of Work**

To accomplish the task of producing better learning material for students, the following tasks need to be accomplished:

Task 1: A detailed list of topics need to be created (1 Week)

Task 2: Setting up a website and server (10 Days)

Task 3: Collection of material from different sources (2 Weeks)

Task 4: Organization of the collected material (1 Week)

Task 5: Recognizing the faults with the collected material (1 Week)

Task 6: Creating material to fill in the gaps between the collected materials (2 Weeks)

Task 7: Creating additional material and coding examples for students (2 Weeks)

Task 8: Updating the website with the material (1 Week)

Task 9: Organizing coding examples and exercises by difficulty level (2 Weeks)

Task 10: Building short quizzes for students (2 Weeks)

Task 11: Putting theoretical material on the website (2 Weeks)

Task 12: Launching the website (5 Days)

1. **Resources**
   1. **Personnel**

People required to accomplish this project would be:

1. A web developer to make a website and handle the server side coding
2. Two content developers who are proficient in design patterns and can research about the material available on design patterns and create new learning content on design patterns.
3. Contact person with professors and technical team at companies to confirm the design requirements and promote the website.
4. Technical executive for customer grievances.
   1. **Facilities/Equipment**

The equipment and facilities required will be:

1. High speed internet connection to do the research.
2. A server for the website.
3. A laptop for the web developer, two content developers, technical intern and customer support representative.
4. Telephone line for customer support representative and technical intern.
5. **Costs** 
   1. **Fiscal**

The fiscal cost will include the salaries of a web developer, web content developer and an intern in computer engineering to contact the companies and professors (Computerworld.com, 2015). It will also include the cost of the website domain.

Web developer at $40/hour = $ 76800/year

Two Web content developer at $35/hour = $134400/year

Computer engineering intern at $20/hour = $ 19200/year

Customer service representative at $30/hour = $ 57600/year

Domain Name = $ 10/year

Hosting (will go up with the popularity of the website) = $ 40/year

Website Maintenance (Parr, 2014, p. 2) = $ 500/year

Five laptops = $ 2500

Internet connection = $ 444/year

Telephone lines = $ 600/year

**Total Cost = $292094/year**

After the first four months of establishing the content, one of the content developers can be relieved from his/her duties.

* 1. **Time**

It will take about four months to set up the website and establish the content. A timeline can be found in the section 2.3. Statement of work. A web developer will be required permanently to maintain the website and a content developer will be required as a part time employee after the first four months to update the contents of the website.

1. **Conclusion**
   1. **Summary**

The proposed website for design patterns will be a one stop destination for students to learn design patterns. This unique website will be a better alternative to the currently available material. The content will be comprehensive in learning and will help students build their foundation in design patterns. The website will be an amalgamation of theoretical knowledge with practical implementation ingrained with small quizzes, and exercise problems. The students will be encouraged to practice the problems by fragmenting them, finding a solution for them and then implementing the code with the help of the website or on their own.

* 1. **Contact**

For more information on the proposal, please contact Deeksha Juneja on email [deeksha@iastate.edu](mailto:deeksha@iastate.edu). Comments, suggestions, feedback and questions are also welcome and much appreciated.

1. **References**

Bautista, N. (2010, July 7). *A Beginner’s Guide to Design Patterns*. Retrieved August 6, 2015,

from <http://code.tutsplus.com/articles/a-beginners-guide-to-design-patterns--net-12752>

[Derek Banas](https://www.youtube.com/channel/UCwRXb5dUK4cvsHbx-rGzSgw). (2012, August 19). *Design Pattern Video Tutorial* [Video File]. Retrieved from

<https://www.youtube.com/watch?v=vNHpsC5ng_E&list=PLF206E906175C7E07&index=1>

*Computerworld's Smart Salary Tool 2015*. (2015). Retrieved September 20, 2015, from

http://www.computerworld.com/salarysurvey/tool/2015

Fowler, M., & Beck, K. (1999). *Refactoring: Improving the design of existing code*. Reading,

MA: Addison-Wesley.

Freeman, E., & Robson, E. (2004). *Head First design patterns* (1st ed.). Sebastopol, CA:

O'Reilly.

Mitra, S. (2015, August 31). *Design Patters*. Lecture presented at Coms 319 in Lagomarcino

Hall, Ames.

Parr, R. (2014). *How Much Does A Small Business Website Cost in 2014?* Retrieved September

20, 2015, from http://www.executionists.com/blog/cost-to-build-websites-2014/

Shalloway, A., & Trott, J. (2010). *Design patterns explained: A new perspective on object*

*oriented design* (2.nd ed.). Boston, Mass. [u.a.: Addison-Wesley].

Shubho, A. (2010, August 9). *How I explained Design Patterns to my wife: Part 1*. Retrieved

September 6, 2015 from <http://www.codeproject.com/Articles/98598/How-I-explained-Design-Patterns-to-my-wife-Part>

Shvets, A. (n.d.). *Design Patterns* (G. Frey, Ed.). Retrieved from

<https://sourcemaking.com/design_patterns>