2009 Sixth International Conference on Information Technology: New Generations

Design Patterns Go To Hollywood: Teaching Patterns with Multimedia

Adam Dukovich, David S. Janzen
Computer Science Department
California Polytechnic State University
San Luis Obispo, CA 93407
adam.dukovich@gmail.com, djanzen@calpoly.edu

Abstract

Studies indicate that understanding the contexts in which design patterns are to be used is one of the most (if not the most) difficult challenge in applying design patterns, yet little research on the topic attempts to solve the problem of better teaching the contexts. This paper discusses a new paradigm through which the teaching of design patterns can be viewed, one which focuses on conceptual examples and contexts as the key elements in teaching design patterns. We created several multimedia learning modules that use this approach and we evaluated the modules by comparing them to other methods of instruction in junior-level software engineering courses. The context-oriented modules performed better (or at least not significantly worse) than traditional lectures on virtually all metrics, and the videos are easily deployable, making them ideal for uses like distance learning, and they can save valuable instruction hours for professors.

Keywords

design patterns; software engineering education; multimedia; learning modules; context-oriented

1. Introduction

Design patterns were widely popularized with the publication of *Design Patterns: Elements of Reusable Object-Oriented Software* [5]. To say that the book was influential would an understatement, in light of the prolific industrial popularity of the patterns [6, 7] and the 13,000+ scholarly citations of *Design Patterns*, commonly known as the Gang of Four (GoF) book. The GoF book is neither the be-all nor the end-all of design patterns, but it was important in popularizing them.

Since 1994, there has been some investigation into how best to introduce these patterns to students. After all, in light of their popularity in industry and academe, teaching design patterns might seem like a natural topic of inquiry. However, most attempts to introduce design patterns into the classroom have focused on using better examples, simpler models, and better explanations. While we acknowledge that efforts along this line are *very* important, we felt that a more radical approach to design patterns pedagogy was needed. To this end, we created several multimedia learning modules (i.e. instructional videos) that focused on teaching students first and foremost about the contexts in which the patterns are to be used. Instead of starting by answering the "how" of these patterns, we endeavored to answer the "why" first, and we accomplished this by starting the modules with sketches that introduced the main idea of the pattern in a non-computer science context, without getting bogged down in terminology or UML diagrams (at first). After this, and after tying the pattern into the CS context, the videos begin to go into the nuts and bolts of the patterns and how they are put together. The idea was that starting with the big picture and drilling down to more specifics afterward was a better approach than giving the students a tool that they did not know how to use, all the while telling them how the tool worked rather than what it did. Such an approach would not cut mustard in wood shop, nor should it in computer science.

This paper presents related work, a project statement, a section that discusses the creation of learning modules, evaluation and the final results.

2. Related Work

A design pattern is, fundamentally, a pairing between a common problem in software development and a proven solution for that problem [11]. The context of a design pattern can be defined simply the circumstances under which a pattern is to be used.

Research on design pattern pedagogy focuses predominantly on teaching design patterns in CS1, as a complement to an objects-early approach [1, 10, 13].

Pecinovsky [9] makes this explicit connection, and his approach follows this desire: in essence, he would like to simplify the patterns and make them more user-friendly so that introductory-level students will be better able to understand the patterns. His approach can be classified as being a structural method of teaching design patterns, which emphasizes better examples and simplification as the key to teaching design patterns.

Pecinovsky's structural method is necessitated by his audience—CS1 students will have less aptitude for the level of technical detail that is exhibited, for one, in the Gang of Four book. However, the teaching of design patterns cannot simply be about finding a better structure. Dewan [3] notes that the biggest stumbling blocks to teaching design patterns were as follows:

1. Students have trouble identifying the contexts in which design patterns are applicable.
2. The examples presented are excessively complex for less mature students.

Many researchers (like Pecinovksy) have opted to tackle the second problem, but ideas on solving the former one are still lacking. *Head First Design Patterns* [4] makes an attempt to introduce the contexts in which design patterns appear on a conceptual level, however, their approach has never been empirically tested. Efforts by Weiss and Nevison [8, 12] also try what might be considered a context-oriented approach to teaching design patterns. Both introduce design patterns in the context of an already-completed, familiar project that has utilized design patterns without the overt knowledge of the students. Such efforts, though, would inevitably tend to be difficult to implement.

One might wonder why there has been such a glut of attempts to try to teach design patterns to intro-level students while there have been substantially fewer attempts to teach patterns to students with more experience. One might reasonably attribute this to a matter of ideology. Pecinovsky (and others) see little difference between OO design and design patterns, and conflate the two as a necessary part of a CS1 curriculum (as in [9]), rather than seeing design patterns as more of a sui generis phenomenon. The ability (or lack thereof) of an introductory student to understand the sophisticated concepts represented by design patterns ought to concern educators, as should the relative lack of attention paid to illuminating design patterns' respective contexts in the current literature. What is needed is a new approach to teaching design patterns.

3. Problem Statement

The goal of this project was to create an easily-deployable (i.e. multimedia) set of design pattern learning modules that would be at least as effective, and hopefully more so, than teaching design patterns with a method primarily focused on teaching structure.

Ideally, these modules will prove to be an effective way of teaching design patterns, and could be useful to an instructor looking to teach design patterns to his or her students, or to a manager in industry looking for some quick training for employees.

4. Creation of Learning Modules

The fundamental goal of these modules was to introduce the patterns' contexts as the key to being able to eventually understand and use the patterns. Only after grasping the “why?” of these patterns could students be reasonably expected to know how to apply them in the future. This led naturally to a certain structure for the modules, which will be shortly discussed in section 4.1.

These modules were meant to be targeted at intermediate-level undergraduate students with some understanding of OO design practices. This was assumed to be superior to efforts targeted at introductory CS students, as students who have a more comprehensive background in concepts such as the object-oriented paradigm would likely be more comfortable with the level of sophistication associated with design patterns than students who have only recently learned the function of a for-loop. This base of knowledge would allow more focus on teaching the pattern contexts, which as Dewan [3] noted is among the hardest (if not the hardest) elements of teaching design patterns. Nevertheless, the examples that the module provides were deliberately meant to be as simple and abstract as possible, so that detail could be subsequently added onto the student's understanding.

Figure 1: Still from the Strategy video

Deployability was another one of the most important factors in creating these modules. We wanted to create learning modules that could easily be used by instructors as part of an in-class lesson, a lab, or homework. This approach was dictated because of the deficiencies of some of the other approaches that sought to introduce patterns in the context of a complicated, multi-stage project that would leave little freedom for instructors with respect to how they approached the material, which we regarded as crucial.

We created three short instructional videos which covered the Adapter, Observer, and Strategy design patterns, as defined in [5]. These patterns were chosen because of their utility to students, their varying levels of complexity, and because each one lent itself fairly naturally to a conceptual example. These videos contain a combination of live-action segments and static slides, which we believed would make the videos dynamic, enjoyable, and informative. We chose not to use professional actors in the videos, but rather upper-division CS students who would already "speak the language."

Figure 1 shows an image from the Strategy video. Each video was tied to one short and one longer exercise that were intended to reinforce the material presented. The viewer is prompted to pause the video in order to complete the exercises which can be completed with paper and pen. The purpose of including these exercises was to allow for us to assess the effectiveness of our videos, which is further explained in section 4.2.

4.1. Structure of the Videos

Each video contains four main sections (acts):

1. A skit that introduces the concept of the pattern in a context entirely unrelated to computer science. For example, the iPod was used as a non-CS example of the strategy pattern, as it allows dynamic selection of songs, videos, etc., in comparison to the static ordering of only songs on a tape player.
2. A section that explains the pattern's context specifically within computer science, and where it might be used in a program that they might write. For example, with the Adapter pattern, the video mentions the pattern's utility in code reuse.
3. A section that looks at the structure of the design pattern and how the parts interact with one another. This section introduces the short problem for the pattern. For the Strategy video, students have to match up classes from a code example with classes from the Strategy pattern.
4. A section that introduces the longer problem. Students are given a piece of code and then refactor it such that it implements the pattern being taught.

This structure stresses the contextual elements of each pattern first, before moving on to the structure and internal relationships. This represents a conscious reversal of most of the research on this subject, in hopes of finding out which approach works better at teaching the patterns. Each video is about ten minutes long.

4.2. Assessment

As indicated earlier, each video included two associated exercises. One was a short exercise (either multiple choice, true/false, or matching) to test students' basic comprehension of the pattern, and the other was a longer exercise to test students' ability to apply the pattern by refactoring an existing piece of code to utilize the design pattern in question. The short and long exercises are intended to take students approximately one and 10 minutes, respectively.

Our three primary objectives with the videos were that the students be able to comprehend the patterns, that they be able to apply them, and that they be able to retain the basic knowledge of what patterns accomplish. We assessed the first two via the exercises previously discussed: comprehension is tested by the student's performance on the shorter question, and application by the longer question. Retention was tested after the fact, with a final exam question that tested how well students retained the concepts of the design patterns they were taught.

5. Experiment Design

In order to evaluate the efficacy of the learning modules, we conducted a controlled experiment in two undergraduate software engineering courses (CSC 307 and 309) at Cal Poly, San Luis Obispo. The courses are intended for third-year computer science and software engineering majors. There is some danger that the students in these courses already know the design patterns presented, but we control for this eventuality by having students state on a questionnaire whether or not they have already used the patterns. The experiment involved two parallel sections of CSC 309 taught by the same instructor, with 40 students altogether in both sections. The experiment was performed once again in a slightly different setting with 20 CSC 307 students, who had comparable levels of experience. The data reflects all of these sections. CSC 309 was the second in a two-course software engineering sequence. CSC 307 is a one-course software engineering alternative.

The experiment proceeds as follows: one section is shown a full video on one of the patterns (e.g. Strategy). After this, the same section will observe an in-person lecture on another pattern (Adapter) and then will be shown a video on the third pattern (Observer) but without an initial skit. The second lab section will receive parallel instruction on the patterns in the same order, but the methods will be different. In the second lab section, the strategy pattern will be taught with the video minus the skit, followed by the adapter pattern taught by the video with the skit, and concluded with the observer pattern taught by lecture. Table 1 summarizes the experiment organization.

	Pattern	**Section 1**	**Section 2**
Activity 1	Strategy	Video with skit (Long video)	Video w/o skit (Short video)
Activity 2	Adapter	Lecture	Video with skit
Activity 3	Observer	Video w/o skit	Lecture

Table 1: Experiment design for CSC 309

Why proceed in this manner? Before we compare our context-oriented approach to teaching design patterns to what other researchers have done in the past, it is important to make sure that the method we chose to use to create the videos does not handicap the material.

Earlier, we mentioned a section of CSC 307. The experiment for that class was similar to the one in CSC 309, except we used an even shorter version of the videos that included no information on contexts (here referred to as the shorter video). The graphs reflect both experiments.

Just to make sure that the variables in play here are understood: the order in which the patterns are presented is constant, as are the exercises used to evaluate students' understanding of the patterns. The independent variable for each pattern is the method of instruction—context-oriented videos, lecture, or video without the skit. The lecture material will be substantially the same as the video without the skit, and both will still have information on the context in which a pattern is to be used.

6. Expected Outcomes

We expected to find that our approach to teaching design patterns, which we have dubbed "context-oriented," is more effective than the prevailing model of teaching the structure of the patterns as the primary aspect of design patterns. In addition, we hope that the learning modules we create will become widely used among educators and professionals in the field as a way of introducing these particular patterns.

In a greater sense, we hope that this work will incline instructors away from the practice of "design patterns-early," which we feel is problematic, and focus more toward teaching the patterns at a later time in the undergraduate curriculum with a context-oriented approach.

7. Results

We present the results in three parts. The first part will look at how well students responded on the questions that they answered immediately upon viewing the videos. The second part will look at how students did on the test question as a test of retention, and the third part will look at how the students graded the videos based on subjective measures.

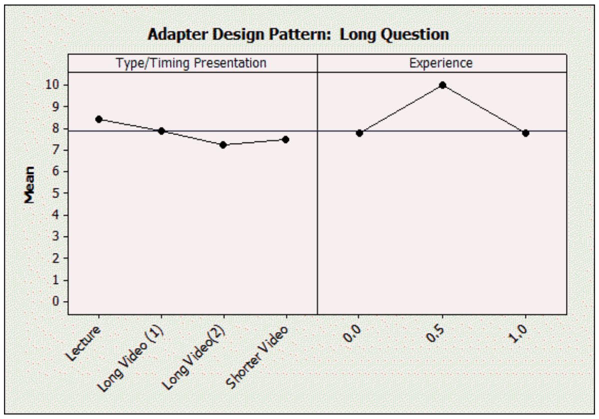

Figure 2: Adapter Exercise Results

7.1. In-class Exercises

The results from these exercises can be partly found in Figures 2, 3, and 4. These figures report the average student score on the longer exercise that tested students' ability to apply the pattern by refactoring an existing piece of code to utilize the design pattern. (The parentheticals indicate on what class the experiment was performed.)

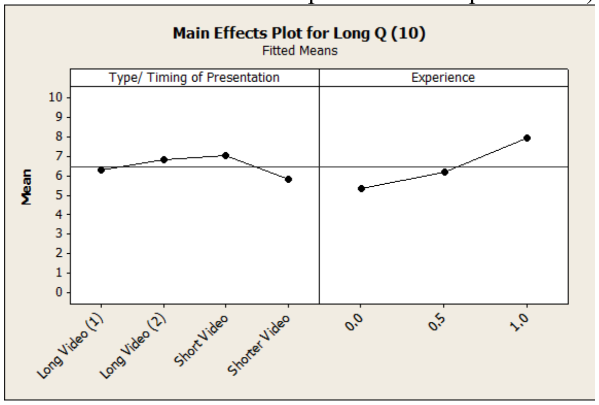

Figure 3: Strategy Exercise Results

In general, lecture was consistently found to be the best method of instruction, and the video completely devoid of contextual information was consistently found to be the worst. The comparisons between different methods of instruction are broken down in Figures 2, 3 and 4 by the three different design patterns. As Table 1 showed, not all approaches were used with all patterns. The experience columns show how well students did on the exercise against their experience with the patterns. We included the data in the paper, but it was ultimately inconclusive.

These results do not really show one type of presentation method as vastly superior to the others. Lecture is best, and the shorter videos fare the worst. These trends are present but muted, suggesting that the effect of which method is used to teach the patterns—at least as measured by the ability to immediately apply the pattern in question—is less urgent. All in all, it is not possible to say that any other method of instruction is significantly better than the context-oriented videos.

7.2. Test Questions

Table 2 shows the performance of the students in CSC 309 on the test question. These questions were embedded in the course final exam, thus measuring student retention of the patterns over time. The numbers represent the students who successfully answered the question.

Problem 1: Strategy		
	Section 1 (Long/Context-Oriented Video)	Section 2 (Short/Non-CO Video)
Correct	14	5
Incorrect	6	15
No Answer	4	4
Problem 2: Adapter		
	Section 1 (Lecture)	Section 2 (CO Video)
Two parts correct	4	8
One part correct	11	11
No parts correct	6	3
No answer	3	2

Table 2: Final Exam Question Results

In both questions, it is clear from visual inspection that the students watching the context-oriented videos did better than their counterparts experiencing other types of instruction. Based on these results, it is safe to conclude that the context-oriented videos perform better in terms of retention than non context-oriented videos (p=0.0024), and since the video element is the only variable that is changed here, one can conclude that the context-oriented approach is better than the approach that does not emphasize context when it comes to getting students to remember the basic ideas of the pattern.

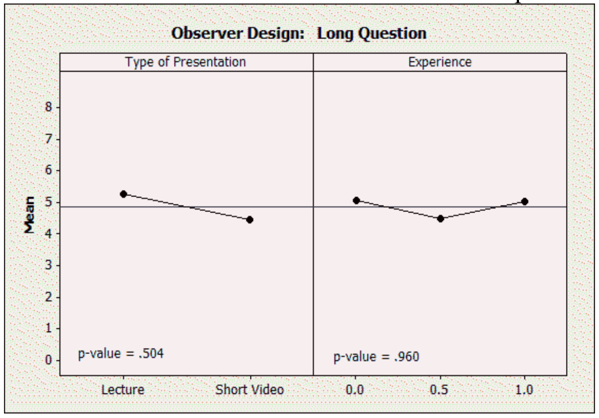

Figure 4: Observer Exercise Results

In comparison to lecture, though, which contains context material, the context-oriented video still does a better job, though the result is not statistically significant (p=0.13). It would appear that teaching with video is not significantly superior to teaching with lecture, but drawing such conclusions was never the purview of this paper.

7.3. Subjective Data

This section breaks down some of the self-reported subjective data that was supplied by students. These questions appeared on a postmortem survey that students answered after each experiment, and they were asked to rate the videos on a scale from one to five (five being very well) with respect to how well the videos taught the context, the structure of the patterns, and just how much they liked the videos. Table 3 reports the question results.

Teaching Method	**Conveys Context**	**Conveys Structure**	**Like**
Long Video	3.48	3.22	2.98
Lecture	3.31	3.30	2.52
Short Video	3.09	2.85	2.48
Shortest Video	3.29	2.93	2.64

Table 3: Subjective Data

According to Table 3, students felt the long video was superior in terms of conveying the context, and they liked it more than other methods of instruction. The lecture just narrowly beat out the long video in terms of conveying structure. These results indicate that, in the opinions of the students who participated in the experiments, that the long video succeeds at its main goal—focusing on the context —and it's a more satisfying experience overall. However, these results should perhaps be taken with a grain of salt, as statistical validation is once again hampered by the small sample size for the shorter video.

8. Conclusion

As can be seen in Section 7, students who viewed videos with a context-oriented approach not only did significantly better on the test questions (although there exist some significant caveats for the second experiment), but also preferred the context-oriented videos on two of the three criteria measured. The data on the in-class exercises is, to be sure, muddled, and larger sample sizes are necessary to determine statistical significance. However, it is safe to say that there is no reason, given the data, to believe that the context-oriented videos are significantly worse than the other methods of instruction that excluded contextual material. In fact, use of the videos could be preferable considering the advantages that the videos have in the other metrics, as well as some of the less quantifiable but nevertheless tangible advantages that they have over, say, lecturing. For instance, one can simply outsource lecture on Strategy to the videos and not need to worry that students are significantly worse off, based on the data and its corresponding analysis.

In addition to the information and analysis about the performance of the videos that included context-oriented content relative to the non-context alternatives, it is worth noting that there is some anecdotal evidence to further back up the case of the videos. Several students, after viewing the context-oriented modules, were immediately able to recall what the pattern did because of the conceptual examples provided. For example, a student reported that he was able to remember that Strategy allows a user to select an algorithm at runtime because, in his mind, as soon as he heard “strategy” he immediately made the connection to the iPod, as was made in the movies. This was reported in several surveys as well, and should not be surprising in light of context-oriented students' superior retention by the metric of test scores.

All in all, the videos that we created during the course of this project—videos that taught the Strategy, Adapter, and Observer patterns—were successful according to the metrics that we set up at the outset of the project. There is more work to be done, but it is our hope that the success of this experiment shows the feasibility of a context-oriented approach to teaching design patterns.

9. Availability

The videos discussed in the paper are available at http://users.csc.calpoly.edu/~adukovic/DesignPatterns.html

10. References

[1] F. Arcelli, S. Masiero, and C. Raibulet. Elemental design patterns recognition in Java. *13th IEEE International Workshop on Software Technology and Engineering Practice,* pages 196-205, 24-25 Sept. 2005.

[2] K. Beck, R. Crocker, G. Meszaros, J.O. Coplien, L. Dominick, F. Paulisch, and J. Vlissides. Industrial experience with design patterns. *Proceedings of the 18th International Conference on Software Engineering,* pages 103 - 114, 25-29 Mar 1996.

[3] Prasun Dewan. Teaching inter-object design patterns to freshmen. *SIGCSE Bull.*, 37(1):482-486, 2005.

[4] Elisabeth Freeman, Eric Freeman, Bert Bates, and Kathy Sierra. *Head First Design Patterns.* O'Reilly Media, Inc., 1st edition, 2004.

[5] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software.* Addison-Wesley Professional, 1994.

[6] Timothy C. Lethbridge. What knowledge is important to a software professional? *Computer*, 33(5):44-50, 2000.

[7] Tracy L. Lewis, Mary Beth Rosson, and nones Manuel A. Pérez-Qui. What do the experts say?: teaching introductory design from an expert's perspective. In *SIGCSE '04: Proceedings of the 35th SIGCSE technical symposium on Computer science education*, pages 296 - 300, New York, NY, USA, 2004. ACM.

[8] Chris Nevison and Barbara Wells. Teaching objects early and design patterns in java using case studies. In *ITiCSE '03: Proceedings of the 8th annual conference on Innovation and technology in computer science education*, pages 94-98, New York, NY, USA, 2003. ACM.

[9] Rudolf Pecinovský, Jarmila Pavlícková, and Lubos Pavlícek. Let's modify the objects-first approach into design-patterns-first. *SIGCSE Bull.*, 38(3):188-192, 2006.

[10] David Reed. Incorporating problem-solving patterns in cs1. In *SIGCSE '98: Proceedings of the twenty-ninth SIGCSE technical symposium on Computer science education*, pages 6-9, New York, NY, USA, 1998. ACM.

[11] Yonglei Tao. Teaching software tools via design patterns. In *ACSE '00: Proceedings of the Australasian conference on Computing education*, pages 248 - 252, New York, NY, USA, 2000. ACM.

[12] Stephen Weiss. Teaching design patterns by stealth. *SIGCSE Bull.*, 37(1):492-494, 2005.

[13] Michael R. Wick. Kaleidoscope: using design patterns in cs1. *SIGCSE Bull.*, 33(1):258-262, 2001.