

A Game Development-based strategy for Teaching Software Design Patterns through Challenge-Based Learning under a Flipped Classroom approach

Ismar Frango Silveira

Faculdade de Computação e Informática – Universidade Presbiteriana Mackenzie
Rua da Consolação, 920 – 01302-907 – São Paulo – SP – Brazil

ismar@mackenzie.br

***Abstract.** Software Design Patterns, namely those belonging to GoF catalog, are gradually being incorporated into undergraduate Computer Sciences-related curricula worldwide. However, this is a subject that often requires students to have some maturity on Software Development issues, which is a scenario that is barely found on actual classrooms. Besides, the comprehensive taught of many patterns' aspects depends upon professors' expertise on a large range of Software Engineering and Object Orientation-related topics. Both aspects are possibly caused by the complexity and abstract nature of the subject, whose concrete, real-world applications not always are successfully reached in an undergraduate course. In this sense, this paper presents a strategy for Teaching GoF Design Patterns through a Game Development Process, under a Flipped Classroom pedagogical context. The results of some case studies are also shown, taken from two classes where Challenge-based Learning techniques were applied.*

1. Introduction

Design Patterns are a better way to think about software. Since they encourage modularity and reuse, they would naturally support and be supported by the principles of Object-Oriented paradigm. In this sense, it sounds natural that such a subject would be a must have on undergraduate curricula for courses in Computer Sciences (CS) and similar ones. However, the teaching and learning of this subject in undergraduate levels has been theme for academic discussion by more than one decade. Since the effective popularity of some pattern proposals— more specifically the GoF catalog [Gamma et al, 1995], many teaching strategies have been presented in the literature to deal with the necessity of learning a highly complex subject like Software Design Patterns in an undergraduate course of Computer Science and correlated ones.

Few years ago, the teaching of Design Patterns still played a discrete role in some Software Engineering syllabi, being barely visited in an undergraduate course and led to be covered in elective courses or at a intermediate level, as already defended by Johnson and Barnes (2005). Recently, this subject has been absorbed by CS-related undergraduate careers, in a movement that appears to be a primary need of business contractors, which have

been pushing Universities to include Design Pattern in their curriculum as soon as possible, giving the high probability of students having to pursue such knowledge in their internships.

This discussion, in fact, is not a new one: Pecinovsky et al. (2006) already proposed a Design Pattern approach for teaching Object Orientation good principles as early as possible. Their arguments rely on the fact that letting this subject to be learned too late would “calcify bad programming habits in students”. Clearly, early adoption of Design Patterns on curricula would require more suitable approaches for younger publics.

Ludic elements have been used by some authors as strategic components for teaching Design Patterns along last years: Dukovich (2008) presents a ludic proposal of showing Design Patterns through multimedia examples and many other authors rely on games to implement their strategies. For instance, Mustaro et al. (2009) present an experience of reverse engineering of games to teach principles of Design Patterns in a undergraduate level; Wick (2005) proposes a sequence of laboratory activities using the Game of Life – an approach similar to Gómez-Martín et al. (2009), who used a family of games instead; on the other hand, Gestwicki and Fu (2008) proposed to teach Design Patterns through the development of a game, strategy that is being used in the context of this paper.

Besides the ludic application to motivate students, some specific teaching strategies were chosen for this experiment, namely the Flipped Classroom and Challenge-Based Learning (CBL). These strategies include cognitive mobilization activities that can be undertaken through the support of technological resources, which includes videos, digital games and simulations, among other elements [Davies et al., 2013

Often simplistically defined as “homework done at school and school tasks done at home” the set of techniques that make up what is now known widely as Flipped Classroom comprises a set of strategies and diverse approaches. Bishop and Verleger (2013) bring a comprehensive review of the literature, which will serve as a guiding principle for the experiment presented here.

Together with Flipped Classroom, it was adopted a CBL (Challenge-Based Learning) approach. O’Mahony et al. (2012) showed that this strategy has been providing greater learning performance gains over a lecture format As in Baloian et al. (2013), the CBL approach here presented consists on an extended form of PBL (Problem-Based Learning), in which the problems are of realistic, open-ended nature, leading to more experiential, project-based and decision-based learning perspectives.

Both belong to the context of Active Learning. Methodologies, whose fundamentals were mapped by an early paper of Prince (2004); a more recent, despite not up-to-date review can be found in Settles (2010). Felder and Brent (2009), define them under a broader approach, categorizing as part of Active learning as all that is related to a course that students are encouraged to do than just watch, listen and annotate. In this sense, such methodologies would embrace a wide spectrum of pedagogical techniques that are mainly based on the introduction of activities to be performed by students in the context of traditional education and the promotion of student engagement in the learning process, as opposed to teaching strategies that do not require students to play an active role in the process, considering them passive

receptors of pieces of information, not always connected among themselves or even related to students’ cognitive framework.

Under these perspectives, next section will show the pedagogical framework being proposed for teaching Design Patterns.

2. The SuperRoamer Game

The experiment consists on the following steps: a GDD (Game Design Document) about a game proposal, here named “The SuperRoamer Game” is to be presented to students in a gradual manner, along four weeks. Students are meant to understand the requirements of each week and construct a class diagram that gradually could model these requirements. Depending on the context, an implementation could also be considered.

All main principles of flipped classroom and CBL are to be applied in all classroom encounters during this experiment, which must never rely on explanatory, pre-expositive classroom moments.

2.1 Week 1: Creational GoF Patterns

In Week 1, the basic game mechanics are stated: SuperRoamer has to go through two jungles: the Amazon Rainforest in Brazil and the African Savannah. He must try to get as many diamonds and pots of gold, besides some jackfruits and *açaí* berries to maintain his energy. He should take care of some beasts scattered along the jungle, which can hurt or even kill him, depending on his accumulated energy.

Table 1 shows the main game requirements for the first week, together with an icon to represent the situation in GDD and the GoF Pattern that is expected to be discovered and applied by students to fulfill the requirement.

Table 1. Creational GoF Patterns



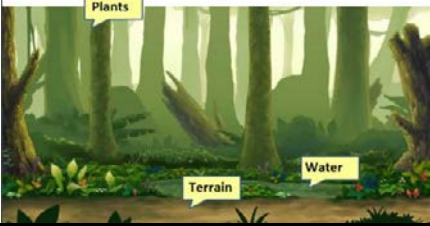



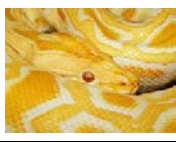
GDD Icons	Game requirement	Expected Pattern
	There is one and only one SuperRoamer along the entire game	Singleton
	There are four collectable items, and the pots of gold and diamonds represent wealth. The consumption of one jackfruit gives two minutes of energy to the player, whilst another item to be sought is the <i>açaí</i> , which should be consumed in great number. Each <i>açaí</i> berry gives 5 second of power.	Factory Method and Prototype.

Table 1, continued. Creational GoF Patterns

		<p>The levels themselves must be constructed following a pre-determined order and combination of elements like terrain, water and plants.</p>	<p>Builder</p>
<p><i>Amazon Rainforest</i></p>	<p><i>African Savannah</i></p>	<p>Depending on the jungle, there will be different animals from three types: snakes, raptors and felines. For the Amazon Forest, we have Anacondas, Caracaras and Jaguars. For the African Savannah, Pythons, Martial Eagles and Tigers. The attack of a raptor takes 30 seconds of power; the attack of a snake, 1 minute; the attack of a feline 5 minutes.</p>	<p>Abstract Factory</p>
			
			

Clearly, all requirements are about situations that required the creation some object. Singleton is the most straightforward pattern, followed by two factory patterns, namely, Factory Method and Abstract Factory. Students must identify two different situations for creating objects, being able to identify Abstract Factory as the most adequate for creating families of objects, instead of Factory Method. As an example, Figure 1 shows an extract of a class diagram made by students that correctly models the last requirement using the Abstract Factory pattern. All classes' names, attributes and methods are in Portuguese.

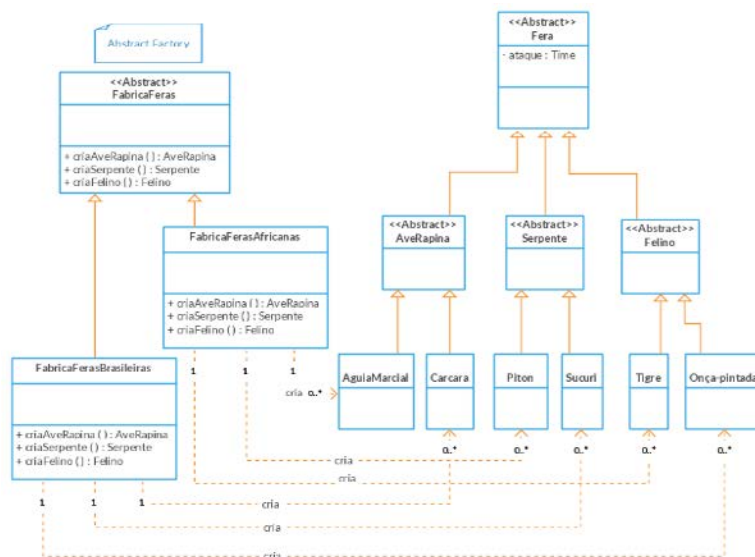


Figure 1. Abstract Pattern applied to Week 1's fourth requirement





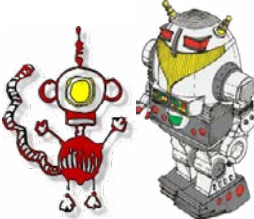
Considerations about Singleton being an anti-pattern and preoccupations about “class explosion” in the context of Factory Method could be explored here by the professor. The natural complexity of Builder pattern could also be explored in this application, as well as considering the use of Prototype to model *açaí* berries because of their expressive quantities (and not for their complexity) could lead to some interesting discussions about performance of dynamic instantiation versus cloning.

It must be noted that no suggestion of pattern should be previously given to students, since the reflection and discovery of which pattern would fit better in each situation is necessary to be taken by students.

2.2 Week 2: Structural GoF Patterns

Activities in Week 2 start by discussing the results of last week and reinforcing theoretically the main concepts behind the five creational patterns. Thus, new requirements could be presented, in which game mechanics slightly change in order to provoke some reflection about the structural aspects of game project. Table 2 shows these new requirements.

Table 2. Structural GoF Patterns

GDD Icons	Game requirement	Expected Pattern
	Users could choose between a male or female avatar.	Bridge
	Every avatar could have any combination of the following items (or not, depends on the user's choice): Cover, Mask, Gloves, Glasses and a magic Utility Belt	Decorator
	The Magic utility belt is an element apart: it has objects and countless pockets; each pocket may contain other objects or even other pockets within them and so on.	Composite
	Many objects in the belt are weapons or shields that can be used to defend or face the challenges of the jungle, as the beasts. There are, however, two types of objects: ordinary objects and magical objects. Common objects can be handled directly by the character to attack or defend. However, magical objects have to be handled by means of wearing magic gloves, which allow the use of objects making the magic affecting less the character.	Adapter
	When the character enters the game, he wins two "beaters", which are robots-emissaries who go ahead to check the operating conditions. There are Water Emissaries to see if the character can follow the jungle rivers and Land Emissaries, who walk certain dry part of the jungle to predict the presence of beasts. With the evolution of the game, the character can win or lose emissaries.	Proxy

As an example, Figure 2 shows a diagram (in Portuguese), made by students, using Decorator to model the requirement related to Cover, Mask, Gloves, Glasses and Belt.

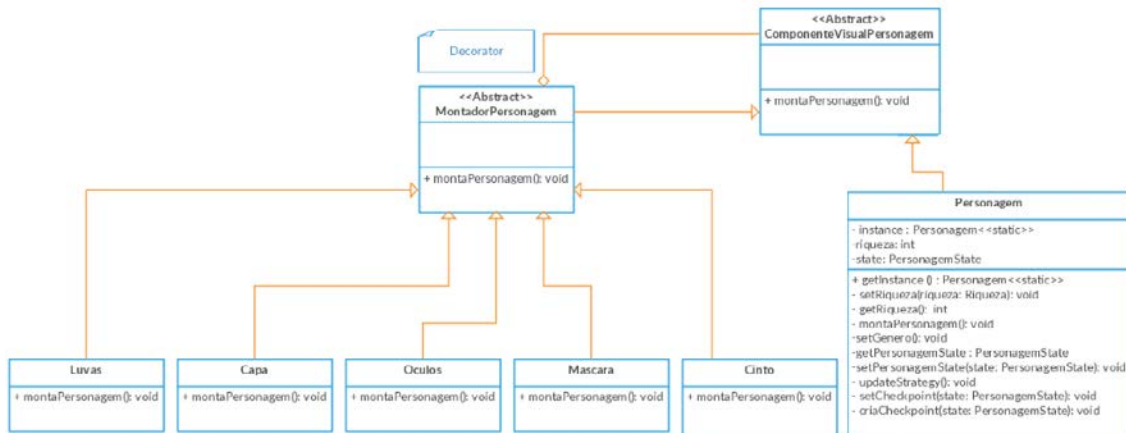


Figure 2. Decorator Pattern applied to Week 2's second requirement

The application of many of the structural patterns does not appear to be so straightforward as the creational ones. While the requirements for Decorator and Composite – and maybe Bridge – are less complex to understand and discover, Proxy and Adapter are two patterns that, in spite of being widely used, requires a higher degree of abstraction. In this point, professor could intervene and bring some “real-world” examples for these two patterns, like the use of stubs (as Proxies) and skeletons (as Adapters) in many distributed computing standards, like CORBA and RMI. Depending on the maturity of the class, more details could be given about different implementations of Adapter (based on multiple inheritance versus delegation) and variations of Proxy (Virtual, Remote, Protection, Smart) [Geary, 2002].

It must be noted that two structural patterns were not covered by these requirements: Flyweight and Façade. It could be asked to students for creating new requirements for showing the application of these patterns.

2.3 Week 3: Behavioral GoF Patterns

As in last week, the discussions about structural pattern must precede the activities of Week 3, which deals with the most complex part of the catalog, the behavioral patterns. Besides being more numerous – there are eleven of these patterns, their understanding relies on a higher capacity of abstraction. For this week, only 3 new requirements are to be presented, as shown in Table 3 and 4.

Table 3. Structural GoF Patterns


GDD Icons	Game requirement	Expected Pattern
	<p>The belt has a number of useful elements. Some of them are specific to defend the attacks of wild beasts:</p> <ul style="list-style-type: none"> • Magic Sword: it must be drawn when the character is being attacked by a snake; • Giant cage: it must be used to trap a raptor; • Hose: used to throw water on felines 	Strategy

Figure 3 shows a students' application of Strategy Pattern to meet this requirement (diagram in Portuguese).

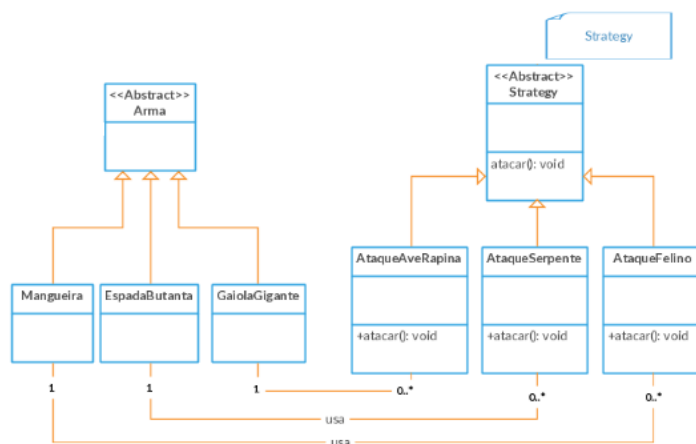


Figure 3. Strategy Pattern applied to Week 3's second requirement

Two additional requirements were presented to students, as shown in Table 4.

Table 4. Behavioral GoF Patterns (continued)

	<p>All items on the belt to fight the beasts have their magical versions. However, it is recommended to use these magic versions only if the character is in a healthy state, namely more than 30 seconds of remaining power. As the use of magic items, even with gloves, consumes some lifetime of the character, if the character is in a moribund state, (the remaining lifetime of the character is less than 30s), the magic weapon can destroy the character even before attacking the beast. In this case (moribund state), normal weapons should be used.</p>	<p>State</p>
	<p>When walking in a jungle, the character can spread three types of electronic organs (the "watchers" - which are somewhere in the belt) along the environment. These watchers are connected (by long-distance brain vibrations) to the character and warn him of the following events:</p> <p>Ear watchers: they warn when there is a suspicious noise in the woods, which can be a birdsong, or the sound snake approaching (felines use to be more silent);</p> <p>Eye watches: they warn when approaching any of the beasts of their field of vision (they are not very useful against snakes);</p> <p>Nose watchers: they warn when they feel a sharp odor in the jungle (felines and raptors usually exhale very strong smells).</p>	<p>Observer</p>

At this point, the instructor could suggest students to create up to eight situations where behavioral patterns could be applied. Some of these patterns would require extremely specific situations (like Interpreter, for instance), but depending on classroom's maturity, even

complex behavioral patterns like Visitor, could be covered (in this case, the combination State/Strategy could be replaced by a Visitor). Experience has shown that students could easily find applications for those more straightforward patterns, like Memento. Some other patterns, like Template Method, Iterator, Mediator and Chain of Responsibility would require a more direct instructor intervention.

2.4 Week 4: Final Pitch and evaluation

In the last week, it would be advisable to discuss the results each student have reached. A “pitch” – a brief presentation of the project – could be scheduled, which would motivate students to discuss their result with colleagues, which would allow them to learn from others in a collaborative way.

3. Experimentation and Analysis

An experimental application of the technique related in this paper was performed with two classes of fourth-semester undergraduate students of a Brazilian University. The study comprised 42 students that had previous contact with only some very basic GoF Design Patterns, as part of a previous course they have taken.

The methodological approach was based on the concept of Perception of Learning [Kuhn and Rundle-Thiele, 2009]. Since it follows qualitative basis, no procedure involving control groups could be taken, so that the same experiments were taken with both classes. Since they had theoretically the same formal pre-requisites, they were considered as a unique group for analysis.

The experiment took four weeks, as described before. A survey was performed in order to try to capture students’ self-awareness of Learning. Using a five-tiered, labeled Likert scale, it was asked to students to self-evaluate their learning of each one of the 23 patterns. Figure 4 shows the results for the creational ones (values in %).

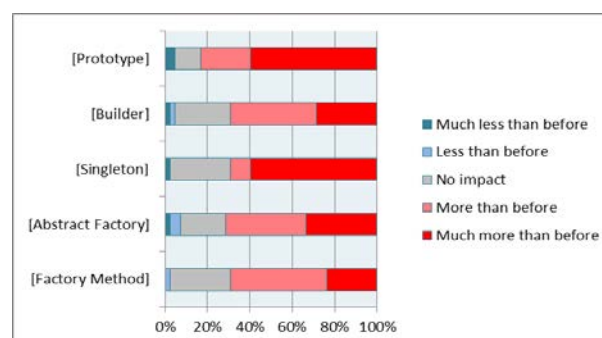


Figure 4. Students’ self-evaluation of Perception of Learning of Creational Patterns

It must be noted that all patterns showed a positive perception of learning of more than 60% (the sum of those who answered that now, they perceive that they know “more than before” and “much more than before”). The Prototype pattern was the best learned, according to students’ perception. The Singleton and Factory Method patterns presents a very good perception of learning, but also presents a considerable amount of neutral responses (those who answered “no impact”), which denotes that part of students already have built the

correspondent knowledge at the previous contact with these patterns, which was before mentioned.

Figure 5 shows the results for the structural and behavioral patterns.

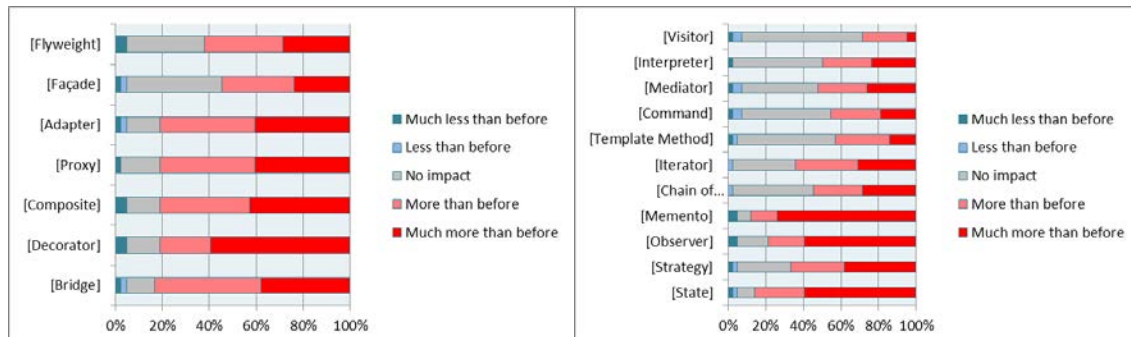


Figure 5. Perception of Learning of Structural (left) and Behavioral (right) Patterns

As expected, Flyweight and Façade were the worst learned structural patterns, since no game requirements were elaborated for them. All the others had excellent perceptions of learning (around 80%), being the Decorator pattern the most understood by students. For behavioral ones, the same scenario is perceived: the three patterns that were covered by game requirements (Strategy, State and Observer) had good perceptions of learning, and the same (or even better) with Memento, which was discovered by students themselves. Iterator and Chain of Responsibility (both covered by situations created by students) had also good values. All other patterns had no good perceptions, in special Visitor, which was considered by students as the worst one.

4. Conclusions and further work

This paper presented a teaching strategy based on the fundamentals of Flipped Classroom and CBL. At each encounter, students were motivated to discuss the diagram they prepared along the past week, with instructor presenting some theoretical discussions about that family of patterns. If an expositive, traditional approach were chosen, probably the contact of students with this abstract subject barely would be converted into meaningful learning. A game was used as the object of study, but eventually any kind of software project could be proposed, according to the audience.

Further works will deal with some improvements on this technique, in order to better cover the behavioral patterns, probably extending the numbers of weeks and augmenting the proposed requirements in order to cover more patterns. However, the stimulus for students to create new requirements is important for developing their creativity. New applications of this technique are currently being performed, together with a variation of it, applied to Architectural Patterns, whose results will be shown in other occasion.

References

- Baloian, N.; Breuer, H.; Hoeksema, K.; Hoppe, U.; Milrad, M. (2004) Implementing the Challenge Based Learning in Classroom Scenarios. Proceedings of the Symposium on Advanced Technologies in Education. Argostoli, Greece.

- Davies, R. S.; Dean, D. L.; Ball, N. (2013) Flipping the classroom and instructional technology integration in a college-level information systems spreadsheet course. In: *Educational Technology Research and Development*, v. 61, 4, p. 563-580.
- Dukovitch, A. (2008) *Design Patterns go to Hollywood: teaching Patterns with Multimedia*. MSc. Thesis, Faculty of California Polytechnic State University, San Luis Obispo, USA.
- Gamma, E. et al. (1995) *Design patterns: Elements of reusable object-oriented software*. Reading, Massachusetts, USA: Addison-Wesley, 1995.
- Geary, D. (2002) Take control with the Proxy design pattern. Available on the Internet: <http://www.javaworld.com/article/2074068/learn-java/take-control-with-the-proxy-design-pattern.html?page=2> retrieved March 19, 2016.
- Gestwicki, P; Sun, F. S. (2008) Teaching Design Patterns through Computer Game Development. *Journal on Educational Resources in Computing*, vol. 8 (1), article 2.
- Gómez-Martín, M. A.; Jiménez-Díaz, G.; Arroyo, J. (2008) Teaching design patterns using a family of games. *ACM SIGCSE Bulletin*, 08/25/2009, vol. 41(3), 268.
- Johnson, C. W.; Barnes, I. (2005) Resigning the intermediate course in software design. In *ACE '05: Proceedings of the 7th Australasian conference on Computing education*, pages 249-258, Darlinghurst, Australia.
- Kuhn, K. A.; Rundle-Thiele, S. R. (2009) Curriculum Alignment: Exploring Student Perception of Learning Achievement Measures. *International Journal of Teaching and Learning in Higher Education*, vol. 21 (3), 351-361.
- Mustaro, P. N.; Silva, L.; Silveira, I. F. (2009). Using Games to Teach Design Patterns and Computer Graphics. In R. Ferdig (Ed.), *Handbook of Research on Effective Electronic Gaming in Education*, 525-545. Hershey, PA: Information Science Reference. doi:10.4018/978-1-59904-808-6.ch030.
- O'Mahoney, T. K.; Vye, N. J.; Bransford, J.; Sanders, E. A.; Stevens, R.; Stephens, S. D.; Richey, M. C.; Lin, K. Y.; Soleiman, M. K. (2012) A Comparison of Lecture-Based and Challenge-Based Learning in a Workplace Setting: Course Designs, Patterns of Interactivity, and Learning Outcomes. *The Journal of the Learning Sciences*, vol. 21,1, p182- 206
- Pecinovský,R; Pavlíčková, J.; Pavlíček, L. (2006). Let's modify the objects-first approach into design-patterns-first. *SIGCSE Bull.*, 38(3):188-192, 2006
- Prince, M. (2004) Does Active Learning Work? A Review of the Research. (2004) In: *Journal of Engineering Education*. In: *Journal of Engineering Education*, v.93, 3, p. 223–231
- Settles, B. (2010) Active learning literature survey. Computer Sciences Technical Report, 1648. University of Wisconsin-Madison.
- Wick, M. R. (2005) Teaching design patterns in CS1: a closed laboratory sequence based on the game of life. *Proceedings of the 36th SIGCSE technical symposium on Computer science education*, 487-491.