Cooperative Learning and Object-Orientated Development Methods

Theda Thomas

School of Business and Informatics
Australian Catholic University
115 Victoria Parade, Fitzroy
Victoria 3065
Australia

t.thomas@patrick.acu.edu.au

Keywords: cooperative learning, object orientation, pedagogical theory and practice.

Introduction

Cooperative learning has been shown to promote higher achievement, enhance critical thinking, improve motivation, deepen students' understanding, foster interpersonal relationship skills and improve students' self esteem (Phipps, Phipps, Kask and Higgins 2001). These skills are obviously ones that we would like to develop in our students at whatever level in order to prepare them for the workforce of the future.

The use of Object Orientation and component-based software is on the increase. Object-Orientated techniques are said to improve productivity, allow reuse of components or objects, improve reliability, make for easier maintenance and offer a disciplined method of design (Johnson 2002). Major players in the software development environment field are moving to a more Object-Orientated / component-based approach. Aschbacher (2001) states that "You would have to be the programming equivalent of a cave dweller to believe components will not become a dominant technology in the very near future." Our teaching in the future will need to encompass these technologies and design methods.

This paper will briefly describe some of the principles of cooperative learning and object orientation and will then show the synergy between these principles. It will close by showing how a particular type of cooperative learning technique could be used to enhance the learning of object orientation while showing how the object orientation applies the principles of cooperative learning.

Cooperative Learning

Hilke (1990) defines cooperative learning as "an organisational structure in which a group of students pursue academic goals through collaborative efforts. Students work together in small groups, draw on each other's strengths, and assist each other in completing the task. The method encourages supportive relationships, good communication skills and higher-level thinking abilities."

Johnson, Johnson and Smith (1991) suggest that there are five principles in cooperative learning. These are individual accountability, face-to-face promotive interaction, positive interdependence, social skills and group processing. Students need to be held individually accountable for the part of the work for which they have been allocated responsibility. They should also be individually accountable for learning everything that the group learns. Face-to-promotive interaction occurs when students in the group support and interact with one another. Positive interdependence is achieved when each member of the group realises that he or she cannot achieve without interacting with the rest of the group. Social skills need to be taught to the students before expecting them to cooperate. The teacher should also monitor these skills during the interaction. Group processing involves the students being aware of how they are achieving as a group and monitoring their own interaction and progress. This is a reflective process.

Object Orientation

Object-Orientated development is based on four major principles according to Kafura (2000). These are the principles of abstraction, separation, composition and generalisation.

Abstraction involves taking an entity and focussing on the essential aspects of that entity while ignoring or hiding the non-essential elements. This allows the simplification of complex phenomena. Abstraction is implemented in Object-Orientation via classes and objects.

The principle of separation refers to separating the class or object's observable behaviour from the methods that it uses to achieve this behaviour. Abstraction in Object-Orientation includes separating the interface to the rest of the "world" (public part of the class or object) from the private data and code thus protecting the private code and data. This reduces complexity. Within an object or class, the data can only be manipulated by its methods. This is called encapsulation.

Composition is the third principle of object orientation according to Kafura (2000). It involves building a complex whole system by assembling the parts into a coherent unit. There are two forms of composition in Object-Orientation, namely association and aggregation. Association involves interacting parts that are externaly visible and can be shared among compositions, whereas in aggregation only the whole is visible and accessible with the components being inside the whole.

The last principle of Object-Oriented development is generalisation. Generalisation involves the identification and possible organization of common properties of abstraction. This is implemented by inheritance and polymorphism among other methods.

The Synergy Between the Principles of Cooperative Learning and Object-Orientation

The first principle of cooperative learning is individual accountability. In Object-Orientation each object can be thought of as individually accountable for the methods and data that are private to it. This is the aspect that enables software reuse. Individual accountability of the separate objects is also evident in the separation principle of Object-Orientation, where objects have both private and public parts. The objects need to interact with other objects or classes as team members need to do in a cooperative learning environment. There is thus interaction through the defined interfaces. These interactions can be via aggregation or association. Interaction within cooperative learning groups can be within the group itself, within subgroups or between group members or subgroups. This face-to-face interaction is the second principle of cooperative learning.

Positive interdependence is involved in both the composition and generalisation principles of Object-Orientation. When one object interacts within another it is dependent on the other object to give valid results. This is especially true if the object inherits the properties of another object. As the components are put together to form a system, each component is positively interdependent on the other components.

The social skills aspects and group processing principles are not evident in Object-Orientation itself, but are evident in Object-Orientated development. It is essential for developers of Object-Orientated systems to be able to work in a team, develop good interpersonal skills and resolve conflicts. As each object is independent and yet needs to interact with other objects this is especially important throughout the project life cycle. Planning the system involves defining objects and their relationships to one another. When putting the system together there must be a lot of cooperation and evaluation to make sure that all the parts work well together.

Examples of Using Cooperative Learning Methods for Learning Object-Orientated Development

Because Object-Orientated development involves writing components that can stand alone and then combining those components into a system, it is easy to create cooperative learning projects that adhere to the principles described above. In order to avoid using the terms component / object / class or class library, the term component has been used in the discussion that follows. The general cooperative-learning techniques are in standard text form with the Object-Orientated /component-based applications of those in italics.

The first example of cooperative learning methods is the Jigsaw method (Johnson, Johnson and Smith 1991), where the teacher is involved in putting students into situations where they must depend on one another. This would be suitable for a lower-level class.

- The teams are heterogeneous with regard to ability level, sex and race. Students are given training in team building and communications as well as group leadership. *This would need to be done for Object-Orientated teams.*
- The study material is designed in such a way that each student can be given a piece without having to understand the rest of the material. Although each student is given only part of the material, each is evaluated on the whole unit. *This could be done by dividing the components between the students with different aspects that need to be applied being expected in each component.*
- The student learns his or her own material and is then responsible for teaching that material to the rest of the students. Each team member is expected to be expert in his or her particular "part of the puzzle." In order to become expert the team members become members of a second group made up of the members from the other teams who have been assigned the same part of the material (piece of the puzzle). *Team members from different groups who have the same component, then get together to determine how to write that component with its data and its methods. They write the component as a group and then go back to their original groups and teach them how they have done this.*
- The group then puts together all its parts of the "jig saw." *The group then puts together all the components and does integration testing.*
- The students are evaluated individually, not as a group*. Further evaluation of the students would be done through tests and assignments.*

Group Investigation is another example of a specific method of cooperative learning that would work well in Object-Orientated development with a higher-level student group as the students themselves have to do the design and divide up the work. It is similar to the Jigsaw method, except that the students plan the learning task and divide the task into subtopics for investigation. They

do not join up with other members of other groups when becoming "experts" in their part of the investigation.

In Object-Orientation, the group will then design their solution using UML models and determining the different components to be programmed, the interfaces that will be needed and the order in which the components must be developed. These components are then assigned to the various team members. This promotes positive interdependence and the use of group skills.

The project itself is also evaluated in the group investigation method with the students preparing a final integrated report and presenting it to the class.

The components need to be integrated into the system or program. Integration testing is also involved. The students will evaluate their group processing and test the interaction of their components with those of the rest of the group. The final program can then be presented to the class or to the teacher. Each student can be responsible for discussing his or her own part of the code. The evaluation can be based on the individual component/s that the student has created as well as the integrated system or program. This enhances the individual accountability and the positive interdependence of the group.

Conclusion

As can be seen from the above examples, as Object-Orientated development is based on principles of abstraction and separation (individual accountability and interaction between objects) as well as composition and generalisation (positive interdependence) it lends itself to a cooperative learning approach. There are many different types of cooperative learning methods, most of which would adapt easily to Object-Orientated development methods. By using the cooperative learning methods, students will gain the benefits of this type of learning but are also "forced" into creating well-designed, truly reusable, stand-alone objects with well-defined interfaces. These methods are not only applicable to component-based development but can be of benefit across the Information Technology curriculum where educators strive for improved teaching and learning.

References

ASCHBACHER, C. (2001): UML components: A simple process for specifying component-based software, *Journal of Object-oriented Programming* **14**(3): 21.

GREY, N. (1996): *Teaching object orientation: Patterns and reuse.* Australian Software Engineering Conference, Melbourne July 1996.

HILKE, E.V. (1990): *Cooperative Learning*. Phi Delta Kappa International.

JOHNSON, R.A. (2002): Object-oriented analysis and design – what does the research say? *The Journal of Computer Information System*, **42**(3): 11–15.

JOHNSON, R.T., JOHNSON, D.W. and SMITH, K.A. (1991): *Co-operative learning: Increasing college faculty instructional productivity*, ASHE-ERIC Higher Education Report no 4, Washington D.C.: The George Washington University School of Education and Human Development.

KAFURA, D. (2000): *Object-oriented software design and construction with Java*. Upper Saddle River, New Jersey, Prentice Hall.

PHIPPS, M., PHIPPS, C., KASK, S. and HIGGINS, S. (2001): University students' perceptions of cooperative learning: Implications for administrators and instructors. *Journal of Experiential Educatio*n **24**(1): 14–21.