Teaching from Different Perspectives

Editors:

Joseph Bergin
berginf@pace.edu

Jutta Eckstein
jeckstein@acm.org

Mary Lynn Manns
manns@unca.edu

Helen Sharp
h.c.sharp@open.ac.uk

Marianna Sipos
sipos.marianna@nik.bmf.hu

This pattern language documents some successful techniques to assist teachers in helping learners look at course material from different perspectives.

Quick Access Table

The following table lists on the left hand side (Objective) your objective, which you might want to focus on in a feedback oriented teaching environment. And on the right hand side (Pattern) the table suggests one or several patterns that will help you to reach this objective.

Objective	**PATTERN**
You want to prepare the students for the world outside the course.	WIDER PERSPECTIVE, NOBODY IS PERFECT, FIXER UPPER
You want to make use of the different perspectives peers can provide.	TEAM TEACHING, INDUSTRY PARTNER
You want to include the different perspectives your students can provide.	ROUND AND DEEP
You want to cover as many topics as possible without losing the focus.	MULTIPLE PRONGED ATTACK, EARLY BIRD, RESTRUCTURE, FIXER UPPER
In addition to knowledge, you want your students to develop tools that they can use later on.	TOOL BOX

The first pattern (WIDER PERSPECTIVE) is the root for all the following patterns. It is a larger or more abstract pattern and sets the context for the different perspectives. You will find more detailed patterns in the categories of widening the perspectives by cooperating with other educators (TEAM TEACHING, INDUSTRY PARTNER, NOBODY IS PERFECT), by teaching multiple things simulateaously (MULTIPLE PRONGED ATTACK, RESTRUCTURE, EARLY BIRD), by using the students' experiences (ROUND AND DEEP).

WIDER PERSPECTIVE **

This pattern is based on ETHOS, written by Klaus Quibeldey-Cirkel [QC] and revised by Mary Lynn Manns.

You want to prepare students in technical-oriented area to be able to use their knowledge for tackling real world problems.

***

An academic concentration in a technical area, such as Engineering or Computer Science, requires a large amount of technical knowledge. Therefore, coursework often stresses the vital technical facts and applications in such areas as programming languages, hardware and software design, and mathematics.

However, the workplace rarely places workers in a position in which they work only with technical matters. Students soon realize that it takes much more than technology to do their jobs. The most influential and complicated matters to handle are often not encompassed in the technology itself but rather in the social and economical implications of the technology.

Even though such realities as economic conditions, human behavior, and organizational politics will affect how students in technical disciplines will be able to do their jobs, they are rarely given an appreciation for these issues. This can give students the impression that expertise in technical topics is all they need to be successful in the workplace. In addition, it will not adequately prepare students to take on a managerial role.

***

Therefore, teach a technical topic from wider perspectives.

Give students an interdisciplinary understanding of the implications of the technology, such as economical, ethical, social and organizational.

Some may argue that there isn't time to do this in an already overloaded course or curriculum. But the technical content can still be the emphasis of the topic. The instructor can favor "breadth over depth" to simply equip students with an appreciation for the other issues they will face when working with the technology.

But this brief overview can lead to underestimating the importance of these issues. You can confront the learner with MISSION IMPOSSIBLE to "shock" him or her into a deeper thinking about the subject and provoke further questioning, exploration, and self- study.

Instructors that feel they don't have the background and knowledge to include a wider perspective in their course can get help from other instructors (TEAM TEACHING) or from invited speakers (INDUSTRY PARTNER). When students ask questions you cannot answer, be honest about what you don't know and offer to get the answer later (NOBODY IS PERFECT) or, if time permits, lead a discussion among the students to explore the possible answers to the question (ROUND ROBIN).

***

Klaus Quibeldey-Cirkel suggests an "ETHOS" approach to remind engineers that a solution to a technical problem commonly comprises economic (E), technical (T), human (H), organizational (O), and social (S) aspects. These aspects should be integrated into the general structure of a course.

A wide range of topics could also be structured with a SWOT approach. The instructor would discuss the strengths (S), Weaknesses (W), Opportunities (O), and Threats (T) of the technology.

TEAM TEACHING *

This pattern is based on Team Teaching, written by Jeanine Meyer [JM] and on Teacher Teams, written by Astrid Fricke and Markus Voelter [VF] and revised by Jutta Eckstein.

You want to teach a subject FROM WIDER PERSPECTIVES by providing different views on it, but you are aware that NOBODY IS PERFECT, which is also true for you. This means that you have difficulties providing many different perspectives.

***

Some topics are very broad and complex. These topics provide a real challenge if you want to cover all their aspects. Furthermore time limitation might make it harder for you to get knowledgable in the whole topic. Students, on the other hand sometimes have difficulties to get a holistic understanding of the topic if they are introduced to only one opinion on the subject matter.

***

Therefore, team up with fellow educators and teach the course together. Your peer might know some aspects of the subject you don't know and vice versa. Working in teams is generally more effective than working alone, because the partners motivate each other and according to the idiom that one plus one is more than three, the team produces much more and better output than each member alone.

It is often helpful if your partner belongs to another domain than you do. This will emphasize different interdisciplinary views on the same subject. For example if you are an academic teacher, your partner might work in industry. Thus your peer doesn't necessarily have to be a teacher. Another possibility is that you are teaching the material from a technical perspective, whereas your partner teaches it from an economic perspective.

It is easier to focus on these different perspectives if you are co-present while teaching, i.e. both teachers are in the classroom at the same time. On the other hand you can also focus on the different perspectives, if the teacher team does not teach at the same time, but the teachers are in the classroom at different times. In all cases, it is important that the teachers and the students understand the distinct roles and yet also understand that it is still one course, with the teachers conferring with each other regularly.

***

The students seem to take over more responsibility for the learning when they see teachers learning from each other. Students also observe teachers (adults) grappling with issues and being truly involved in the subject matter. If there are problems with students, it is wonderful to have someone to talk with about the problems.

TEAM TEACHING when applied as co-teaching is much more vivid than teaching alone. The vividness is enabled, for example, by different voices and different body languages. Your partner will also help make sure that you do not lose yourself in details, but instead reinforce that the students get a holistic understanding of the topic. However, if you prefer to co-teach in presence you might encounter some financial or temporal constraints. However, the teaching partners can decide to team teach one of each of their courses and thus compensate for the financial constraints; however the temporal constraints remain (or might even be doubled).

Although you might imagine that team teaching will cut your workload in half, in reality, harmonizing the course topic and the regular consultations require a lot of time. Moreover it will get harder the less the co-teachers act as a team.

At Pace University, Jeanine Meyer and Dr. Martha Driver did team-teach a course on English literature with a side-focus on mulitmedia. Jeanine focused on the technical aspects whereas Martha took care of the contents (Beowulf to Lear: Text, Image and Hypertext: http://csis.pace.edu/grendel).

Jeanine Meyer and Dr. Sandra Flank did team-teach a summer intensive one-week course: Introduction to Multimedia for the Classroom. Jeanine took over the technologist role and worked with Dr. Karen Berger on Strategic Web Marketing.

INDUSTRY PARTNER *

You want to teach a topic from WIDER PERSPECTIVES but NOBODY IS PERFECT.

***

Even the most experienced teacher can't know everything. In addition, teachers that spend most of their time in the classroom aren't well versed on what is happening in industry. Yet, they want their students to have an appreciation for how the topic they are studying is relevant and useful in the work place.

At the same time, individuals who spend most of their time in industry aren't aware of how and what the students, their future employees, are being taught.

***

Therefore, partner with one or more people from industry. Include them in some classroom activity.

You can ask them to do a presentation, or join an in-class exercise, or provide feedback on student projects. Take the time to inquire about what they are most comfortable with and offer opportunities that match their interests and skills.

Inviting industry partners into your classroom means you must be willing to accept that the class will have some unknowns. For example, if you ask someone to do a presentation, you may find that this person does not have good presentation skills. Brief your students that even though the industry person is not a professional presenter, he still has valuable information to contribute.

ROUND AND DEEP **

This pattern was written and revised by Helen Sharp.

You want the whole class to benefit from the experiences of individual students in your class.

⁂

An experienced student is likely to gain a deep understanding of a complex concept by relating it to his or her own experience. But the same experience which results in a deep understanding may also limit it because a rounded understanding of a complex concept can only be achieved by considering different perspectives.

To gain a deep understanding of a complex concept, the student needs to consider it from many different perspectives, but your own experience is necessarily limited and a classroom exercise (in a University or Industrial setting) is necessarily too simple to cover adequately the deep issues surrounding the concept.

Experienced students will relate a new concept to their own real-world experiences, and will form a deep understanding of it, but if their experience does not validate the concept, then its significance may be lost, and if their experience does validate the concept then although understanding may appear deep, it may also be narrow.

⁂

Therefore exploit the variety of the students' own experiences to deepen their own understanding of the concept and to provide alternative perspectives for other students.

The key is to give students the opportunity to relate the concept to their own experience and to allow time for students to share that understanding with others in the class. The first part results in a deepening of the student's own understanding. It can be achieved through a group or individual activity that exercises the concept from a number of perspectives. Choosing suitable activities to encourage students to relate explicitly and deeply their experiences to the concept being taught is not easy. However a small group discussion around an example application can be enough to elicit different views. The second part of the pattern (allowing time to share understanding) results in a rounding of other students' views of the concept. This can be achieved in a group setting, where students can hear others' perspectives, exchange ideas, concerns, lessons learned and so on.

Widening the discussion to include the whole class, or at least a larger group in the class can be achieved by presentations followed by discussion. Asking students to present findings to you and to fellow students compels them to clarify their own thoughts. This in itself can deepen the student's own understanding, while the presentations and discussions deepens the collective understanding of the class by sharing other students' experiences, misconceptions and breakthroughs.

How well this pattern works depends on how many and how varied are the experiences in your class and how prepared the students are to expose their understanding of the concept. Some perspectives may conflict, but good learning experience will emerge from their resolution and compromise. You will also learn quite a lot and broaden your understanding!

A variant on this is to let members of a team who finish early to split off and join other teams to learn from their debates. This increases the level of cross-fertilization, and is also an alternative strategy when the student group is too big to allow each team to present their findings.

***

This pattern gives you a chance to exercise a concept or set of concepts within a collaborative environment and it encourages students to reflect on what they have learned. However the exercise must be chosen carefully to fit within limitations (time and space) of the classroom situation, and it must be deep enough to engage your students and not to simply re-state the key concepts given in lectures. To make the most of students' experiences, you need to keep a relatively low profile and allow students to learn from each other.

***

Helen Sharp has used this pattern to teach postgraduate software engineering professionals object modelling, where some very interesting discussions have emerged between telecommunications and business systems developers because they have quite different technical concerns in each of their domains. A similar pattern was proposed by Martin Barrett from East Tennessee State University.

NOBODY IS PERFECT **

This pattern was originally contributed by Astrid Fricke and Markus Voelter [VF] and revised by Jutta Eckstein.

You are teaching a topic and your students are asking questions, which are beyond your acquired knowledge.

***

Students expect the one and only right solution to a problem from the instructor. However, on the one hand there is often no single answer, but many equally correct answers. And on the other hand you might not know what the best answer is. You cannot know everything. If you paper over the cracks, the students will stop relying on you in areas where you are knowledgeable.

⁂

Therefore, admit your limitations with grace. Do not try to be perfect. In particular, if you cannot answer a question, admit it.

Be honest to the students, by telling them that you do not know the answer. Avoid searching through your material in front of the students. Unless you really know where to find the answer, it is unlikely you will find the correct answer in this short time frame.

Don't create the impression that the question is overly difficult just because you don't know the answer. This will hinder the students trying to speculate about the solution. Suggest that you will look up the solution, or explicitly ask the students whether one of them knows the answer to the question. You can also suggest to either work on the solution jointly with the students or start exploring with the students to find the answer.

⁂

It is much harder to implement this pattern in countries where the culture does not allow people to admit that they need help or where the students are regarded as impolite if they admit that they do not know the answer.

Use REFLECTION to foster the trust in the students' own competency.

MULTI PRONGED ATTACK *

This pattern was originally contributed by Joseph Bergin [JB2] and revised by Jutta Eckstein.

Despite the fact that your curriculum is packed with topics, you want to include additional material.

⁂

You are feeling overwhelmed by the amount of material you need to cover in a course. However, you feel the need to continually include new material, although you wonder what you can omit in order to have the time for teaching the new material. There never seems to be enough time to teach all that we want to. **Sometimes eliminating stuff to make room for new topics is the best choice, because some topics are becoming obsolete by the introduction of new ones. But often you will find that there are no topics to leave out and that the new ones are just coming on top of them.**

⁂

Therefore, choose your examples and exercises so that they cover more than one idea or topic at the same time.

As Kristen Nygaard used to say, one must start with *sufficiently complex examples* instead of *sufficiently simple examples* in order to teach the students the world view.

Students have to see some examples, which cover multiple aspects and they have to work on some appropriate exercises. These exercises should focus on the current topic, but allow students to learn additional topics along the way. These aspects can be covered in a way that the students don't really realize that they are

learning another topic additionally. The exercises can be either single faceted or multi faceted. The former seldom stretch the students imagination or skill. Teach a topic in such a way that it reinforces several ideas at the same time.

Often you can add new material to a course simply by changing the examples you use in class. When used well, this gives the students very rich environments in which to learn. It can also engender a questioning attitude in the students.

***

For example, when teaching novices how to program in an object-oriented way you still need to teach them if statements and loops. You can either do this in isolation with abstract examples, or you can do so in such a way that the broader goals of object orientation are also reinforced. Do this by choosing examples and exercises that illustrate while statements, for example, but in the context of doing something useful in a class. Even better, you can choose the example in such a way that it also teaches students something about the breadth of computer science as well. For example, a finite automaton has a control loop. It can be built as a class, and it shows some of the underpinning theory of CS.

Karel++ [BSRP] uses a gentle introduction to the art of object-oriented programming by requiring that the students build a new class and new methods to do anything, reinforcing object-orientation at every step, even while teaching if statements.

This pattern can also be combined with TEAM TEACHING, as Jeanine Meyer and Dr. Martha Driver did at Pace University in a course on English literature with a side-focus on mulit-media. While Martha focused on English literature, Jeanine emphasized the multimedia aspect.

FIXER UPPER **

This pattern was originally contributed by Joseph Bergin [JB4] and revised by Jutta Eckstein.

You want to challenge your students with sufficiently complex examples but do not want to overwhelm them.

***

We often need to introduce students to a new field requiring mastery of several topics. However, students often fail to see how the topics fit together when introducing them sequentially. **Too often students work on only "toy" problems because they may not have the experience or skill to build large artifacts from scratch and there is only just so much time. But all realistic problems are large and the day in which small problems were interesting is about past.**

On another front, students also have difficulty when unexpected errors arise in their own work. They also often fail to have a grasp of the means of locating and correcting errors.

***

Therefore, give your students a fairly large artifact that is generally sound but with carefully introduced flaws. Ask the students to repair and discuss the flaws.

The artifact proposes to be the solution to a problem, but while generally correct, the instructor has purposely introduced flaws into it. Most of the flaws should be simple and obvious to most readers. There should be one or two deeper flaws.

Ask the students to find and correct the flaws. Ask them to discuss the nature of the flaws found and the reasons for their changes. Finally, ask them to discuss the overall structure of the artifact and draw inferences from it.

Fixing a larger artifact than can be created by students is generally within their grasp. It gives them a better sense of scale of interesting problems and permits them to integrate a number of issues into the solution of a single problem.

***

This pattern allows students to actively work with larger artifacts, which can even be LARGER THAN LIFE, than they can develop completely themselves. They benefit since finding flaws in their own work is a valuable skill. It is important that the overall structure of the artifact be sound.

The best way to develop such an artifact is to start with an excellent solution to a problem and then doctor it by introducing flaws. There must be different kinds of flaws, but probably not structural flaws if you are dealing with novices. This latter rule can be broken if the artifact is introduced later in the course rather than at the beginning, at a point at which structure is the main issue. When used as an EARLY BIRD instance, however, concentrate on flaws of detail, rather than structure.

Most such exercises cause the students to generate questions that can be a fruitful source for classroom discussion. If the instructor is careful to introduce certain flaws, the student can be led in a desired direction to further explorations.

The pattern must be carefully used if student honesty is an issue. It is easy for one student to point to the locations of errors in C++ programs, for example. One way to address this is to use large artifacts that require teamwork. Another is to ask questions concerning the structure as well as the errors. Finally, the students can be asked to examine the artifact before they are given the full set of questions that will be asked about it.

Some students are frustrated by large artifacts. The instructor must be prepared to provide support and encouragement that the real world really is like that and that it is ok to initially (a) be frustrated and (b) lack knowledge.

Note that in the United States, a "fixer upper" is a house for sale that is in poor condition. They are sold to people, mostly young, with more energy and enthusiasm than money. The concept is that you need to repair it (perhaps extensively) after purchase.

***

Joseph Bergin has used this pattern when teaching beginning programming. He uses a program illustrating a number of syntactical constructs that have not yet been introduced in class. (It has been used as the first assignment.) The program can be large enough that its structure is not obvious, with e.g. two or three classes with several short methods.

Sometimes one part of the program is more complex. The errors can be mostly syntactical and lexical, so that the compiler can find them. By introducing one or two semantic errors, the program does not perform as expected.

Jutta Eckstein has used this pattern for teaching advanced design. She presents design results from a former course and asks the students to improve it by discussing it and correcting the errors. Here the flaws are not carefully introduced by the instructor, but accidentially by the former students.

Linda Rising has used this pattern with very large artifacts –large enough that they can't be understood by a single person at the level of the students and therefore require GROUPS WORK.

EARLY BIRD **

This pattern was originally contributed by Joseph Bergin [JB1] and revised by Jutta Eckstein.

You want to ensure that your students remember (at least) the most important ideas.

***

Students have sometimes difficulties to distinguish between the important and the unimportant ideas. However, students often remember best what they learn first.

***

Therefore, organize the course so that the most important topics are taught first. Teach the most important material, the "big ideas," first (and often, using a SPIRAL approach). When this seems impossible, teach the most important material as early as possible.

Important (big) ideas can be introduced early, even if they can't get complete treatment immediately.

You have to mine the course for its most important ideas. These ideas become the fundamental organizational principle of the course. The ideas, and especially their relationships are introduced at the beginning of the course and are returned to repeatedly throughout the course. This way the most important things in the course receive more focus from the instructor and the students. Students can be made more aware of what is paramount.

Often only simple aspects of an important idea can be introduced early. If there are many important ideas, it can be LARGER THAN LIFE. Sometimes it is enough to give important terms and general ideas. Some "big" ideas are thought of as advanced. It is difficult to introduce some of these early. Sometimes a really big, but difficult, concept can be introduced incompletely. Then as other material that relates to it is covered, the relationship to the big idea is carefully explored.

You have to be able to analyze deeply what are the consequences of developing material in a particular order. It is often helpful to have a forum in which ideas

can be discussed and refined. It is also often necessary to develop your own materials, which requires time and effort.

⁂

It may be a mistake to try to use this pattern with material that has clear prerequisite ideas to the important ideas. This would be especially true if the relationship between the prerequisite idea and the big idea is especially subtle or if the prerequisites are especially difficult to master.

⁂

Joseph Bergin provides several examples of what he defines as most important, e.g.: Teaching objects first (or at least early); Teaching design first; Teaching concurrency first in operating systems; Teaching user requirements first in Database; Teach recursion before loops. Of course, these are Joseph Bergin's definitions of what is most important. You may disagree, but then it is your course, so discover and implement your own "firsts."

The book Karel the Robot [RP] was designed with this pattern in mind as a way of teaching procedural programming (procedures first). Its successor, Karel++ [BSRP], attempts to do the same with Objects (classes first).

RESTRUCTURE *

This pattern is based on Big Picture on a Small Scale, written by Kerstin Voigt [KV] and revised by Marianna Sipos.

You want to teach state-of-the-art technologies and concepts that have become essential, and you want the existing material to take account of the new developments.

⁂

Often you are facing the challenge that new ideas come and go into practice in very short timescales. As some concepts become important, others become less important. By the time a new idea becomes established as a base concept, other new ideas have come along.

Some years ago, when you considered teaching them, they had been taken to the end of the course, because in order to understand them, it was often necessary to learn some prerequisite knowledge. New technologies help you to apply and understand these new ideas while performing familiar tasks. New developer tools also support the development of this understanding through practice. This is useful because complicated ideas at first appear to be simple. At some point, these new ideas become base ideas and you have to modify your teaching of them, and introduce them early on in the course. However, you also know that teaching these new base ideas early on is essential, because considering existing state-of-the-art technologies and concepts will influence the students general thinking about the subject matter.

⁂

Therefore, you should rearrange your syllabus, by starting the course with the new base concepts.

You have to recognize that the new base concept is the new "big idea" and teach it as an EARLY BIRD. In order to take the prerequisite knowledge into account, you need to teach the new idea briefly at the beginning and come back to the subject matter as often as possible using an SPIRAL approach.

Rearranging the first part of the course has an impact on the succeeding topics. So you need to rearrange the whole course in order to respect the new ideas.

***

RESTRUCTURE has been applied several times: when structured programming came up, or when object-oriented programming became fundamental [KV], and nowadays when distributed systems are rising to become everyday technology. If you do not restructure your material you will not have enough time to teach and to delve into the more important subjects.

You might want to introduce the new base concept on an abstract level followed by concrete practical examples, as suggested by ABSTRACTION GRAVITY. Or you might consider teaching the new ideas incrementally following EXPERIENCING IN THE TINY, SMALL AND LARGE.

TOOL BOX *

This pattern was originally contributed by Joseph Bergin [JB3] and revised by Jutta Eckstein.

You want to enable your students to profit from a personal toolkit.

***

The typical intent of a course is that students benefit later from the knowledge gained earlier in the course. But knowledge is not the only result of the learning effort. GROUPS WORK or REAL WORLD EXPERIENCE may also result in tools. However, these tools are hardly treated as helpers to foster successive learning.

***

Therefore, develop not only knowledge, but also concrete tools, which students can use in later courses. Ask your students to build things in early courses that will actually be used later in the same course and in later courses as well.

Student exercises have multiple parts. One part of each exercise is to build a general tool that might be useful in other projects and to take some effort in its proper formulation for reuse. The design for reuse must be explicit and must be discussed by the students and commented on by the instructor. GROUPS WORK can help to combine individual designs of the tools, discuss the relative merits of each and then build a common implementation that improves each of the individual designs.

Both students and the instructor should spend time to evaluate the tools for correctness and also for the potential of reusability. Instructors in later courses

must be aware that students have these personal tool kits and should use exercises and projects that exploit the tool kits and permit their extension. Thus coordination with later courses is an important element of this pattern.

You need to give thought to the design of the early courses as to what tools are broadly useful, but especially which tools will necessarily be useful in later courses. Thus you need to build a plan and must be able to provide implementations of those tools that the students will need, but which they will not build themselves.

***

Students become apprentices in the same sense that young people once served as apprentices in medieval guilds. There they spent their early years building tools they would need if they were to achieve master status in the guild.

Students also gain skill in the early courses building reusable components. To the extent that they fail, the lessons are reinforced in the later courses when they need to rebuild parts of the tool kit for use in the projects of those courses.

***

Kernighan & Plauger's Software Tools books [KP] were a good use of this pattern. This book should probably be read by anyone who wants to implement this pattern. It shows how a high degree of reusability can be achieved with extremely simple tools.

Instructors can provide some tools (e.g. data structures and algorithms) in the form of class hierarchies or other libraries. Students complement this collection. For example, the instructor can give a singly linked list and have students build a doubly linked list. Both would be included in the tool kit.

The students can build their personal tool kit incrementally using a SPIRAL approach.

Thumbnails

The following patterns are not part of this language, but they are referred to by one or more patterns above.

ABSTRACTION GRAVITY [EMWM]

Concepts that must be understood at two levels of abstraction require time for a SPIRAL approach to learning. However this can be time consuming.

Therefore, introduce a concept at its highest level of abstraction and use reflection on the concept to link the higher-level abstraction to the lower one.

EXPERIENCING IN THE TINY, SMALL AND LARGE [EMWM]

A complex concept is difficult to understand unless you have experienced it by example. However concepts are often so complex that experiencing the whole in one step doesn't help either.

Therefore, introduce the concept in three stages, tiny, small and large, which allows you to monitor the students' progress on a topic-by-topic – *tiny* – basis, to test if the student can combine the topics and apply them in a larger – *small* – setting and to solve a real-world – *large* – problem using all parts of the concept, thus seeing the *big picture* respectively.

GROUPS WORK [EBS]

You are only one resource for the students. Given the number and difficulty of student questions and concerns you are actually a rather small resource. Your students need frequent feedback on what they do and how they do it.

Therefore emphasize group work in your courses. Use both large and small groups. Use both long lived (weeks) and short lived (minutes) groups. Groups can come together for a few minutes in a class to consider a question posed by the teacher. They can work for an hour or two together in or outside the classroom or lab. They can work in teams for days and weeks on larger projects.

LARGER THAN LIFE [EBS]

When faced with a new concept, students often focus on low level details, ignoring its higher-level aspects. Students are able to read, understand, and modify artifacts larger and more complex than they can themselves build. They also need to see problems of a realistic complexity so that they don't get the idea that all problems are small and simple.

Therefore, give students an example problem or artifact that is too large for them to be able to focus on the details in the time available. Structure the assignment so that they can work effectively with large-scale conceptual or structural knowledge of most of the artifact.

MISSION IMPOSSIBLE [EMWM]

Often new learners arrive at an abstraction not via generalization from a deeper understanding but from a simplification of something they do not yet understand. Such simplistic truths are dangerous, because they lead learners to construct simplistic solutions that do not really solve problems. Worse, the learners' lack of experience prevents them from recognizing the shortcomings in their thinking.

Therefore, present the learner with a problem that seems straightforward to solve but whose complete solution requires a much deeper understanding than the basic concepts afford.

REAL WORLD EXPERIENCE [EBS]

A lot of concepts are too abstract for students to conceive their value. And even worse students often doubt the viability of these concepts. Assigned

problems or lab projects help to make those abstract concepts more concrete. However restricting students to lab environments deprive them of exercising the issues in their rightful habitation – namely the work place.

Therefore involve the students in real world situations, by inviting them to accomplish a project in a real world environment.

Involving a domain interest allows the students to experience the real project life, from the time pressure of a deadline to the pride of demonstrating the result.

REFLECTION [BEMW]

Sometimes, learners believe that the trainer has to deliver all the knowledge, but the students would learn much more if they would explore problems by themselves. Furthermore, students often anticipate that an instructor will solve each and every problem for them, but the knowledge of the instructor is also limited. You want the students to uncover solutions for complex problems by drawing on their own experience rather than just letting them accept what they have learned by listening.

Therefore, provide an environment that allows discovering and not one which is limited to answering questions. It is the students' debt of delivery or of inquiry. Train students so that they are searching for solutions by exploring the problem.

ROUND ROBIN [EMWM]

One of the most difficult aspects of teamwork is getting everyone in the room to work on equal footing. However, you want to get everyone's participation and input and you especially want to encourage the quieter members to take a more active role.

Therefore, use a round robin technique to solicit suggestions. Go around the room or table. As each member of the team contributes an idea, write it down on the board. The goal of the round robin is to allow the group to move ahead at an even tempo but to give people enough time to think.

SPIRAL [BEMW]

You want to enable students to solve meaningful problems as early in the course as possible. Students learn best when they are doing things, and meaningful problems motivate them to work harder.

Therefore, organize the course to introduce topics to students without covering them completely at first viewing so that a number of topics can be introduced early and then used. The instructor can then return to each topic in turn, perhaps repeatedly, giving more of the information needed to master them.

Acknowledgement

We thank all those who have contributed to the Pedagogical Patterns project, especially the authors of the patterns we have revised and incorporated into this paper. We also thank our EuroPLoP 2003 shepherd, Joseph Bergin, for helping us to improve the paper.

References

BEMW	Joseph Bergin, Jutta Eckstein, Mary Lynn Manns, Eugene Wallingford. *Patterns for Gaining Different Perspectives*, Proceedings of PLoP 2001.
BSRP	Bergin, Stehlik, Roberts, and Pattis. *Karel++*, Wiley, 1997
BMRSS	Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M. (1996). *Pattern-Oriented Software Architecture: A System of Patterns*. Chichester, England: John Wiley & Sons.
CA	Christopher Alexander et.al., *A Pattern Language: Towns – Buildings – Construction*. Oxford University Press 1977
EBS	Jutta Eckstein, Joseph Bergin, Helen Sharp. *Patterns for Active Learning*. Proceedings of PloP 2002.
EMWM	Jutta Eckstein, Mary Lynn Manns, Eugene Wallingford, Klaus Marquardt. *Patterns for Experiential Learning*, Proceedings of EuroPLoP 2001.
JB1	Joseph Bergin, *Early Bird*, http://sol.info.unlp.edu.ar/ppp/pp34.htm
JB2	Joseph Bergin, *Multiple Pronged Attack*, http://sol.info.unlp.edu.ar/ppp/pp57.htm
JB3	Joseph Bergin, *Tool Box*, http://sol.info.unlp.edu.ar/ppp/pp36.htm
JB4	Joseph Bergin, *Fixer Upper*, http://sol.info.unlp.edu.ar/ppp/pp31.htm
JM	Jeanine Meyer, *Team Teaching*, http://sol.info.unlp.edu.ar/ppp/pp40.htm
KP	Brian Kernighan, P.J. Plauger, *Software Tools in Pascal*, Addison Wesley, 1981
KV	Kerstin Voigt, *Big Picture on a small Scale*, http://sol.info.unlp.edu.ar/ppp/pp46.htm
MF	Fowler, Martin (1997). Analysis Patterns. Reusable Object

	Models. Reading, MA: Addison-Wesley Longman, Inc.
PPP	Pedagogical Patterns Project Home: www.pedagogicalpatterns.org
QC	Klaus Quibeldey-Cirkel, *ETHOS*, http://sol.info.unlp.edu.ar/ppp/pp27.htm
RP	Richard Pattis, *Karel the Robot*, Wiley, 1981
VF	Markus Voelter, Astrid Fricke, *SEMINARS*, http://www.voelter.de/seminars
WNCD	Webster's New Collegiate Dictionary. G & C Merriam Co, 1959