NAUR 1970

CONSIGLIO NAZIONALE DELLE RICERCHE

ISTITUTO DI ELABORAZIONE DELLA INFORMAZIONE

LEZIONE "LEONARDO FIBONACCI" 1969

P. NAUR

"PROJECT ACTIVITY IN COMPUTER SCIENCE EDUCATION"

I. E. I. - VIA S. MARIA, 46 - PISA
PUBBLICAZIONE N. A 70-10

PROJECT ACTIVITY IN COMPUTER SCIENCE EDUCATION

P. NAUR
University of Copenhagen

In introducing the subject I have chosen for to-day's lecture I have a slightly unpleasant feeling that it may appear to you as perhaps trivial and not worthy of the occasion. For that reason it may be relevant to remind ourselves that what is trivial may sometimes, in the right context, assume great importance. As an example of this, let me recall how the Japanese got their phonetic syllabic script. At the time when this happened, around A. D. 750, the Chinese already for many centuries had developed a literary culture, based on their picture script. Chinese and Japanese are quite different languages, but Japanese pronunciation is based on a fairly small number of syllables, and it so happens that these syllables existed as words in Chinese. What happened was that the Japanese took over from Chinese the written pictures for these words, but used them to denote the sound of Japanese syllables. To a Chinese scholar this procedure, which resulted in putting together his familiar signs in a, to him, completely nonsensical order, must have looked like an absurdity not worthy of attention. In any case it has had no influence on the Chinese script, which has remained a picture script to this day. In reality, as we will appreciate immediately, this phonetic script was a tremendous step forward.

The trouble of advocating project activity as a significant part of higher education is that education is that the elements of this activity tend to appear trivial, both in description and in actual practise. This is in striking contrast to the usual course contents of mathematics, the sciences and humanities, where there is a large body of highly structured knowledge. Successful project work depends on the skilled application of certain techniques. The real difficulty lies not in the mastery of each of these, but rather in their interplay, and in their consistent, yet flexible, adaptation to the actual project at hand.

Another way to express this is to say that project work to a large extent is a question of experience. This is not usually felt to be the province of systematic higher education, but is something which candidates expect to acquire later, on the job. The main question raised in the present article is whether the great need for people experienced in project activity in computer science can be relieved by course work in higher education.

The discussion has two parts, the first centered around the elements of project activity, with special attention to computer science. The second is concerned with some educational issues. The discussion has been influenced by experiments along these lines carried out at Copenhagen University.

What project activity is.

By a project we mean the work going into solving a partly defined, not too small, but definite problem, involving design and planning for new construction as essential parts. Thus in architecture the development of a plan for a town, or for a building, are projects. In the work with computers the development of a compiler, or of an operating system, or of a large application program, are typical projects. By simple extension, project activity is the type of activity going into projects.

Project activity, by its very nature, is characterized by several interrelated aspects. Thus, first, it involves *problem solving*. Second, in addition to solving, project activity involves *problem definition*. Indeed, the initial definition of the problem to be solved in a project is normally quite incomplete, or even logically inconsistent. It is an important part of the project activity to discuss the definition of the problem, to clarify it, to make it more definite, to modify it, to discover contradictions in it, and to discuss and resolve the contradictions.

Third, project activity involves the *contact* and *organization* of the group of people engaged on the project. Typical projects can only be successfully tackled by having several, or many, persons working in parallel.

Projects in computer science.

Work with computers offers a variety of opportunity for project work. The design and development of the computer itself is a large project, likewise the development of its associated basic software. The development of the solution to each application problem beyond a certain size is again a project.

If we consider the need for project work in connection with higher education in computer science, it must be noted that project work already is a well established part of engineering education. Thus, that part of computer science which already is strongly tied in with engineering, viz. the design and development of computer hardware, is already taken care of.

By contrast, the introduction of design and development of basic software and of application programs in higher education will need a new emphasis on project work. Indeed, in many academic fields the primary interest has so far been toward knowledge and truth, rather than toward design and construction. To quote G. Forsythe [6]: « To a modern mathematician, design seems to be a second-rate intellectual activity ». Thus the work on basic software, which tends to take place in fairly close contact with mathematicians, has a strong need for a change of orientation toward project work.

Where application programs are concerned, the work will have to proceed in close collaboration with practitioners and scholars of other fields. In certain fields, such as experimental physics, there is already a tradition toward projects, so the contact with computer science will give rise to no new situation in this respect. However, there is a wide range of fields, covering practically all of the traditional university subjects, which will have to include a new, computer project oriented attitude as part of themselves. In many of these fields the tradition is toward work in a very individualistic style, which is likely to contrast rather strongly with the concept of project work as here described. If in these fields the computer is to fulfil its promise as a tool of revolutionary power, then the attitude toward project activity will have to change deeply.

For this to take place there is first of all a need that the computer scientists realize the challenge in this situation, and develop project activity in computer science as a distinct and important part of their field, and find ways to include it in the education. This is the background of the discussion and suggestions which follow.

Problem solving.

A major element in project activity is problem solving. Both the design of the overall plan of the project and the working out of the minor parts, even down to minute details, can be regarded as problems to be solved. For this reason a sound attitude to problem solving is a vital prerequisite to successful project work.

Problem solving is the subject of a small, but very useful, literature. Polya in several books [2, 4] has discussed problem solving with special regard to problems of mathematics. Hyman and Anderson have discussed problem solving in a wider context [3]. Their discussion is centered around eight rules for problem solvers:

1. Run over the elements of the problem in rapid succession several times, until a pattern emerges which encompasses all these elements simultaneously.
2. Suspend judgement. Don't jump to conclusions.
3. Explore the environment. Vary the temporal and spatial arrangement of the materials.
4. Produce a second solution after the first.
5. Critically evaluate your own ideas. Constructively evaluate those of others.
6. When stuck, change your representational system. If a concrete representation isn't working, try an abstract one, and vice versa.
7. Take a break when you are stuck.
8. Talk about your problem with someone.

A list like this is likely to remind us again that the difficulty of project work does not lie in the understanding of the elements of it, but in their skilful combination.

Design techniques.

Somewhat related to problem solving are the design techniques required in project work. The problem of design is concerned with the order in which decisions about the solution should be made and about how to arrive at reasonable compromises between conflicting requirements. This probably is the area where our present principles are the least helpful and where the most is left to individual talent and experience. The problem is, however, the subject of much thought. Thus one interesting approach has been described by Christopher Alexander [5]. In this all the design requirements are first listed. Second, for every pair of requirements it is decided whether they are related or not. Third, the interrelationships are used to group the requirements in such a manner that strongly related requirements come into the same group. Finally, the design is developed by starting with the groups of strongly related requirements and only later bringing the more loosely related requirements together.

In the field of software design, the proper sequencing of the design decisions has been the subject of several papers, thus particularly those given in [1] by E. W. Dijkstra (p. 181-185), S. Gill (p. 185-188), B. Randell (p. 204-208) besides the discussion given on pages 45-53. Randell makes the distinction between a « bottom-up » design, where the designer starts with the primitive operations available in the computer and gradually builds up more complicated actions, to end with the complete system required, and a « top-down » design, in which the designer starts from the characteristics of the complete system and gradually breaks this up into smaller units, until it is expressed in terms of the primitive operations available. The remark is made however, both by Randell himself and by Gill, that actual design can follow neither a pure « bottom-up » nor a pure « top-down » approach, but must necessarily follow a far more complicated pattern of reasoning. In fact, the mental path followed in a complicated design will depend on a vast store of the designer's past experience and probably will defy description if this is to be stated within reasonable limits.

Project documentation and the design process.

One of the most important things to stress in computer application projects is that the outcome of the project must consist of extensive documentation, in addition to the program. Thus, as a minimum, the documentation must include a description of the function of the program in ordinary terms, and of the formats of input and output.

However, what should be equally stressed is that the working out of the documentation should be an integral part of the design process, and not just something added after the event. This follows for several reasons:

- The sheer size of a project makes it necessary that what is done during its development is recorded in writing.
- The collaboration of several people over the solution of a complicated problem requires permanent records which can be referred to be all participants.
- Even when done by an individual, systematic, conscious work requires a written formulation. There is a great difference between going through an argument mentally and recording it in writing. The act of putting the words on paper forces you to consider every step more carefully, and will often provoke a revision and improvement of the solution. This observation is closely related to the problem solving precept no. 8 of Hyman and Anderson [3]: « Talk about your problem with someone ».

The documentation developed during the design process should as far as possible be written and arranged in such a manner that it may enter, with only slight change, into the final documentation. For this to be possible the following points should be followed:

- From the start of the project, maintain a list of contents of the intended final documentation. Let every part of the documentation written during the design process be identified with an item in that list, immediately as it is produced.
- Spend the very first project effort, say the first percent of the intended expenditure of work time, on the writing of a first, crude description of the whole project. This description may conveniently be written as a synopsis of the final project report.

If the above approach is followed, the final documentation will include, not only a description of the finally adopted solution, but equally the reasoning leading up to it. Thus the alternatives considered during the early stages will appear, with the arguments used to select the one actually chosen.

This is of great value, for two reasons. First, the reader of the documentation will usually benefit greatly from being able to go along the same mental path as the designer, getting thereby a deeper understanding of the solution chosen. Second, whenever a change to the solution is contemplated, it is vital to be able to find out, for each design decision entering the solution, how essential it is to the total. Otherwise there is a great risk that whoever makes a change will impair the correctness of the system.

The typewriter as a tool in documentation.

If we grant that one of the principal lessons of the work with computers is the realization of the strong mutual influence of the tool used to solve a problem, the problem itself, and our thinking about the problem (see ref. [7]), then it should be clear that in project work, and particularly in the documentation part of it, we should expect a strong influence on our work from even the elementary tools used in our writing. In computer science, it is tempting to suggest that the use of a powerful text handling system in a computer would be of major benefit in the documentation process. As reported in [1] by Gillette (see page 62) this is not directly confirmed in practise, however. The problem seems to be that education has failed to train the students properly in the primary act of doing the reporting about their development work when and where it actually takes place.

For these reasons there is good reason to turn the attention to this elementary point: how is the primary information, the considerations going into the design, put on paper in the first place? I would like to suggest that it might perhaps have a significant effect on the work of our students if we made it a regular part of the project work that the reporting were done by means of a typewriter, and that consequently our students should have access to typewriters in ample quantities.

It should be stressed that this suggestion should not be taken to imply that our students should be required to acquire great skill or speed in typing. The important thing is that they should acquire the habit of recording, as and when their design work proceeds, all the relevant arguments and conclusions in typed form. As part of this habit they should get used to using a uniform column and page format, including suitable identification of pages, headings, etc. all in such a way that these notes will be suitable for further rearrangement and editing solely with the aid of scissors and adhesive tape.

Use of check lists.

The simple use of lists of items to be remembered or checked at suitable stages of the project must be regarded as one important element in project work. The establishment of the check lists themselves then becomes a central part of the formulation of the rules of the project work and thus one place where the teacher can convey his knowledge of the proper procedures to the students.

Examples of check lists can be found in [1] pages 160-180 and 209-211, and in [2] on page XVI-XVII. These are not primarily oriented towards the education considered here, however, and for this purpose many others would be required. Two examples, of such lists are:

MAJOR DECISIONS IN DESIGN OF DATA SYSTEM

1) Of the total data processing, what should be done by people, and what by machine?
2) What devices and data representations will be used at the interface between man and machine?
3) What will be the processing cycles of the total system?
4) What programs are needed in the system, and what is the function of each?
5) How are the larger data collections stored and represented?
6) Processing done by people: who will do what, where, and when, and how often?
7) Which design goals cannot be satisfied, and why?

POINTS TO BE CHECKED IN A PROPOSED DESIGN

1) How well are the goals covered?
2) Simplicity of the design: check that design has no unnecessary parts or functions.
3) Performance: estimate the approximate performance of the system.
4) Stability: does the system remain well under control when subjected to isolated cases of malfunction or mistake?
5) Supervisability: will the responsible people be able to keep track o the operation at all times?
6) Changeability: is the design conceived with the mind to possible later modifications?

The need for group work.

Because of our insistence that a project involves a fairly large amount of work, project work must necessarily be conducted by people working together in groups. This is sufficient reason for insisting that our education in project work must involve work in groups of students.

A more specific reason for advocating group work is that certain phases of the work on a project in a deeper sense involves the interplay of several people with somewhat diverse interests. Thus particularly the initial phases of a project will consist in deciding precisely what aims the project should have, on the basis of general ideas about what is desirable and what can be accomplished with the resources available. This decision should be made in a discussion among representatives of the various groups of people who will be in touch with the system to be constructed.

If the education is to give a complete picture of the project activity, this type of discussion must enter into it, which necessarily requires that group work takes place. Even so the difficulty remains that the students do not naturally represent diverse interests. Perhaps this could be overcome by letting the students in a group assume different roles during the discussion.

Introducing project work in higher education.

In this section we shall consider some of the considerations which may help us to decide when and how project work is best introduced in higher education. First, let us consider the most appropriate time to introduce it. It may be argued that projects will have to be introduced at a fairly late stage of the study, because otherwise the students do not have an adequate background of knowledge to build on. This in not a valid argument, however. An important part of the characteristic flavor of project work is that the work must proceed from whatever knowledge is at hand, whether this be incomplete or not. Indeed, this must be recognized to be the terms of work for projects at any level.

Quite contrariwise, strong reasons may be put forward that project work should be put in early in education. For one thing, the systematic principles of the work are simple enough. For another, the real grasp of these principles requires time and experience, in other words that the student has lived through vain attempts to solve problems. As a third point, the attempt to solve a problem which has arisen through a project at an early stage of the education, even if very unsuccessful, may act as a strong motivation for working at that problem more systematically at a later stage of the education.

Projects to be done early in the education of course must be fairly limited in size. This raises the additional problem that some of the principles of project work do not come into full play in small projects. This problem is just one of the broad class of educational problems which derive from the fact that the educational environment cannot be like the real world in all respect. The only reasonable way to overcome this difficulty is to insist on the application of the proper techniques, even in smaller projects.

Guidance of project work via the evaluation procedure.

It must be recognized that the introduction of project work in formal education raises some difficult problems of evaluation and grading. Thus, because of the freedom of choice left to the student in the work, solutions which are acceptable may differ greatly in many respects, and a reliable comparison between them tends to become difficult and time consuming.

In dealing with this problem it is appropriate to distinguish between two different purposes of evaluation and grading. First, evaluation provides a measure of the ability of the students, and thus as the basis for issuing diplomas to those reaching a certain level. Second, the very way in which the evaluation is carried out acts as a direct motivation to the students. This effect is sometimes deprecatingly described as the tendency of students to work for the exams.

Some of the solution to the evaluation problem lies in a conscious exploitation of the second of these effects. Indeed, this effect shows that the evaluation, far from being an ad hoc addition to the course work, may be used as a powerful means for conducting the attention of the students in the direction where it is best spent.

Applied to project activities, this way of looking at the evaluation shows that the evaluation criteria must be chosen carefully, to correspond with those aspects of the student answers which show evidence of the students' ability to carry through the project work in the proper fashion. Further, these criteria must be communicated to the students when the problem is given, to guide them in their efforts.

In an experiment along these lines, the students were given the following guidance together with the problem:

The solution to the problem must be submitted in the form of a typed report having the following sections:

1. Discussion of the problem, including such points which have needed clarification, and the alternatives considered during its solution.
2. A description of the structure of the program, with explanation of its relation to the method of solution.
3. The program text, suitable annotated.
4. A tabular description of all variables used in the program.
5. A description of the input to the program, in the form of a guide to the user of the program.
6. Description of the output from the program.
7. Test data, suitable for a full test of all actions of the program.
8. Test report, describing the evidence that the program works correctly.

The solution will be judged on the following points, which will have equal weight:

a. The quality of the report with respect to orderliness, readability, clarity of expression.

b. The quality of the discussion of the problem, particularly as regards clarity and the presentation of the justification of the solution chosen.

c. The quality of the description of the program structure.

d. The program and its documentation, with special attention to correctness, adequacy of commentaries, and readability.

e. The ambition of the solution, *i. e.* wideness of the class of problems which the program will process successfully.

f. The adequacy of the test report.

If problem requirements along these lines are formulated and adhered to strictly during the evaluation, a sound basis for turning the students' attention in the direction of the proper project activity has been laid.

Students' mutual evaluation.

In project work with large numbers of students and a limited number of teachers it may be worth while to supplement the normal grading of the project reports with a mutual grading among the students. A practical way of doing this is to have the students go through an evaluation process including four steps:

1. Self evaluation: As a part of the report, each student is requested to submit his own evaluation of it.
2. First mutual evaluation: Immediately upon submission, the reports are exchanged among the students, so that each student gets one report for study and evaluation. No two students exchange reports directly. Evaluation period: approximately 1 hour.
3. Second mutual evaluation: Like the first evaluation, only with a different exchange pattern.
4. Confrontation: Each student is faced with the two costudents who have evaluated his report. The three together are supposed to discuss the report, and as far as possible arrive at an evaluation to which they can all agree.

A procedure of this kind has proved in practise to act as a powerful stimulus to the students. It gives each student an opportunity to study the report of two follow students and to discuss them with their authors. It should thus be regarded as a motivation tecnique, not primarily as a grading technique. If an attempt is made to use the results formally for grading, the whole procedure will probably meet with strong protests from the students, and the value of it will be lost.

Automatic grading of programs.

It need hardly be mentioned that the programs written as part of project work, as far as possible should be graded by a grader program. The structure and use of such programs have already been reported in the literature [8, 9]. It may be added that the use of a grader tends to turn the students' attention strongly towards the program correctness, and less towards the documentation. It may therefore be wise also to include projects which aim solely at the production of the documentation, while the program itself may be left in the form of a sketch.

Evaluation of group work.

Project work done by groups of students raise several evaluation problems. First, by its very nature such work must obscure the individual contributions, and it will be difficult or impossible to use it for individual grading. Second, it may perhaps be felt desirable to include the effectiveness of group activity as such among the points to be graded. It will be hard to devise a suitable way of doing this which will not at the same time tend to counteract that collaborative attitude, which should be part of the value of the group work. As the third difficulty, the success of work in a group will depend quite heavily on the individual personalities involved. Thus some individuals may find themselves strongly degraded in group work.

A way to overcome these difficulties, which has been found to work at least reasonably well, is the following: First, the formation of groups is made entirely free for the students, as long as the group size is at most four. Thus those students who prefer so may work individually. Second, the work done in groups is expected to be of higher quality than individual results. In particular, it should have very few trivial mistakes and should be more carefully worked out. This is taken into account in grading. Third, all students within the same group will receive the same grade.

Conclusion.

The introduction of project activity in computer science education will meet with difficult problems, both related to the subject matter itself, which will not obviously be found worthy of course work in higher education, and to the difficulty of evaluation of the outcome of the work. However, in view of the great need for personnel trained in this activity these problems must be solved. For the present it is appropriate to start experiments on projects placed early in the curriculum, based to some extent on work in groups, and with guidance in the form of detailled description of the issues on which the work will be graded. The students' mutual evaluation and automatic grading of the programming using a computer are possible ways to overcome some of the evaluation problems.

REFERENCES

[1] Software Engineering (ed. P. Naur and B. Randell), Nato Science Committee 1969, available from Scientific Affairs Division, NATO, Bruxelles 39, Belgium.

[2] G. POLYA: *How to solve it,* Doubleday, 1957.

[3] R. HYMAN and B. ANDERSON: *Solving Problems,* Int. Sci. Tech. (Sept. 1965), 36-41.

[4] G. POLYA: *Mathematics and Plausible Reasoning,* Vol. I and II, 1954 Princeton University Press.

[5] C. ALEXANDER: *Notes on the synthesis of form,* 1964, Harvard univ. press.

[6] G. E. FORSYTHE: *A university's educational program in computer science.* Technical report no. CS39, 1966, Computer Science Department, Stanford university.

[7] P. NAUR: *The Place of programming in a world of problems, tools, and people,* IFIP Congress 65.

[8] P. NAUR: *Automatic grading of students' Algol programming.* BIT 4, (1964) 177-188.

[9] G. E. FORSYTHE and N. WIRTH: *On Algol education: Automatic grading programs.* Comm ACM 8 (May 1965), 275-278.

TIPOGRAFIA ODERISI EDITRICE - GUBBIO - 1970