Courseware Development at the Crossroads?

Jef MOONEN
Department of Education, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands

Considerable effort is being made to produce more courseware of better quality. Different strategies have been used to support the courseware development and production process, examples of which are: individual, team, and industrial approaches. In general, the quality of courseware is improving and the number of courseware products is growing. However, development and production costs remain high and, although schools have many computers, use of courseware in classrooms is still limited. Is it that available courseware does not provide for the needs of teachers and students?

Perhaps inappropriate courseware is being produced as a result of poor analysis of how to use courseware in schools. Do we proceed by stressing formative evaluation during the development process? Or should we use other strategies, such as the stimulation of design choices which reduce the need for formative evaluation?

Keywords:
Courseware production, Didactical approaches, Methodologies for courseware development, Sunshine project, POCO project, Individual approach to courseware development, Cost of courseware development, Formative evaluation, Teacher toolkits.

Courseware and Crossroads

Courseware Directions

Courseware has been around for quite a while. In the beginning, courseware was seen as an automated extension of programmed instruction. More recently, we have seen courseware evolution in two other directions: open-ended programs and sophisticated intelligent-tutoring systems.

Courseware Production

From a logistic and organizational point of view, as well as for financial reasons, production of a large number of courseware packages for a great variety of subject areas is not a practical proposition. Vonk [12] has used an effective metaphor to visualise this situation. He compares available courseware packages with small streets in a large city. On a map of the city, streets are coloured according to the subject area concerned. The result is a map with a lot of very short coloured lines scattered around. If the metaphor is carried further, one can compare the city with the body of knowledge and skills to be mastered throughout school. The streets allow us to manoeuvre through this body of knowledge. Clearly, manoeuvring will not be helped by having only small stretches of street available here and there. What is needed are big boulevards that will speed up traffic going from one point to another. Of course, small streets are also necessary to allow access to the big boulevards:

Crossroads

For financial and organizational reasons, it is impossible in practice to have ‘boulevards’ of courseware packages available within the near future. We are left with the question: what courseware to develop as a first priority? Various aspects are relevant to this question.

One obvious aspect to be considered is the educational sector for which the courseware is to be developed: courseware designed for primary
education differs from courseware for secondary, vocational, or higher education.

Other aspects are the views of both student and teacher, which can be presented in the form of opposing views in different dimensions. Do students prefer to work individually or in small groups? Do they prefer courseware which offers a lot of flexibility or more convergent, computer-directed courseware? Do teachers prefer courseware that can be adapted to a specific curriculum and specific didactical approaches? Or do they prefer ready-to-use courseware? Do teachers want to use computers in a limited number of subject areas, or in as many as possible? Do they want to use the courseware in a laboratory situation, or in the classrooms during regular lessons? Do teachers was courseware which helps to design tests and which keeps track of student performance? Or do they only want courseware for demonstration purposes? Do teachers want courseware that can ‘do it all’ in an integrated way, or do they prefer content-specific, single-focus courseware?

Towards In-Depth Investigation of Didactical Approaches

For financial reasons and in view of the fact that educational innovation and change are very slow, we suggest that, for the present, effort should be concentrated on the development of courseware which is not overly complex to use or to produce. The kind of courseware to be developed depends on local, curricular, and cultural circumstances. The main point is that courseware development should follow and support established didactical approaches, instead of trying to suggest new approaches. Traditional didactical approaches will then have the necessary time to develop new strategies. In summary, we recommend at the moment the design of courseware which can be introduced on a large scale in existing curricula, with minimal logistic, organizational and financial problems.

Evolution of Courseware Development in The Netherlands

The Current Situation

Development of courseware has evolved over the years. At first, courseware was mainly developed by individuals. As a result, each such individual had to be a specialist in at least three areas: subject matter, didactical approach, and programming. Later, a multi-disciplinary approach was used, incorporating the different specializations in a more appropriate, ‘professional’ way. Ensuring smooth cooperation of team members from different disciplines became a new issue. Finally, an ‘industrial’ approach was recommended, mainly to strengthen project management of the development and production efforts, in order to improve cost control and time schedules [5,6]. The overall reasons for this evolution were the conviction that courseware should be integrated into existing curricula and the belief that education needs many reasonably priced courseware packages. The integration should be facilitated by an interdisciplinary team approach and the need for software packages should be fulfilled by an industrial approach to development.

Dutch Experiences in Courseware Development: An Overview

During the past five years, from 1984 to 1988, the Informatics Stimulation Plan (INSP) was in operation in The Netherlands [10,11]. Courseware development was a major activity in this plan. What was learned from this activity? In the next three subsections, we shall describe some examples of courseware development, and comment on these.

Development of a Methodology for Courseware Development

Background

When the Dutch government started the Informatics Stimulation Plan, there was no generally accepted methodology for courseware development. One specific line of action in the plan was aimed at the creation of such a methodology for the development of courseware in the The Netherlands. It resulted in a set of manuals for courseware development [2,9].

A general methodology

The main characteristic of the development methodology laid down in the manuals is the explicit separation of design phase and technical implementation phase [8]. These two phases are
interfaced by a ‘script’, which contains a complete and detailed description of the product under development. In addition to the script, a prototype of the product is produced. This approach to courseware development draws heavily on development tools and associated hardware and, because of this, provides an opening for solving the problem of portability. The hardware used in the design phase is much more powerful than the hardware needed to run the final product. Powerful tools can therefore be used in the design phase, in the development of a prototype and in the iterations needed to optimize the prototype. SUN-workstations and powerful PC-ATs were used as design machines, while PC-XTs, or compatibles, were target machines for the final product.

Results of a pilot project

As an example and a specific test case, the ‘Sunshine project’ was carried out. Evaluation of the project [3] revealed limited applicability of the methodology. The problem was not financial—considerable means were available—but one of project management; especially a failure to realize smooth cooperation between the different actors in the development process. This partial failure was due to unclear project goals and specifications. Also, system designers and programmers did not keep to specifications. Whether these problems were a result of ‘theory’ failure, or of ‘implementation’ failure, is a matter for discussion [1]. The next section will make it clear that implementation failure was implicitly thought to be the major cause.

The POCO Project

A second example of Dutch courseware development is the POCO project. POCO stands for “Programmatuur Ontwikkeling voor Computers in het Onderwijs” (or, in English, “Courseware Development for Computers in Education”).

Background

In 1987, in the fourth year of INSP, it became clear that the methodology for courseware production presented above, was too expensive for large scale application. However, the other courseware development projects within INSP, using more traditional approaches to courseware development, mostly following a team approach, were also not able to deliver enough courseware packages on time. The Dutch government therefore decided to start a new major courseware development project, the POCO project (1987–1991), to produce courseware for primary, general secondary, and vocational education. The POCO project was to develop courseware ‘very closely’ fitting existing curricula and ‘frequently’ used by teachers in their lessons. In addition, the project had to build on previous results of INSP, especially the work done on the development methodology, described earlier. The project was to separate courseware design and development in well-defined phases and to use techniques and strategies for strong project management.

Description of the POCO project

A ‘plan of action’ [7] was defined, outlining four phases:

(A) priority assessment,
(B) product description,
(C) technical production management, and
(D) courseware distribution.

In phase A, it is decided which courseware to develop, taking into account the available budget, needs of the educational sector involved, and estimated success of computer use in an application area.

In phase B, the courseware is designed; its result is a global functional description, the ‘script’ of the product to be realized, and a prototype.

In phase C, global specifications are drawn up, which are then further detailed; the technical implementation, coding and debugging, is done.

In each phase, different groups of specialists take part in the activities. Therefore, a quality assurance plan was needed to ensure that the end product of each phase is delivered in accordance with specifications, agreed standards, and within agreed time limits.

A special and very important feature of the plan of action is a ‘confrontation’ at the end of each of the first three phases, in which the end product of a phase is put before an audience, selected from the courseware target group. This kind of formative evaluation has to ensure that the product under development stays within the goals originally set by the target group. In this way, revision of the fully developed product is no longer
necessary. More cost-effective revision of 'partial' products can replace it.

Results
At the end of the so-called first cycle of the POCO project—December 1988—the first products should have been delivered. This was not the case. The project was officially started in August 1987 and should have delivered a first series of 18 courseware packages by the end of 1988. Possibly, the time plan was unrealistic. The delivery of the courseware packages is now planned for October 1989.

The conclusion is that even very strong project management, working on the basis of a very detailed quality assurance plan, has not succeeded in keeping production on time. Lack of qualified personnel, elaborate contract negotiations between the many parties involved in the project, and time-consuming revisions, because of faulty communication between parties, caused substantial delays.

As a result, the planned 'confrontations', where intermediate products were to be put before a target group audience, either did not take place at all or just in a nominal way. Furthermore, the development team, including the project management, seemed not very interested in the opinions of the target groups. Possibly, this lack of interest was caused by a fear of still more delay. At the present moment, the consequences of the omission of 'confrontations' for the final acceptance of the product by the target group are unclear.

Other Experiences
As a third example of courseware development in The Netherlands, we shall discuss a more individual approach.

Background
Students of the Division of Educational Instrumentation, Department of Education, University of Twenty, complete their studies with project work. They can choose for development either a courseware prototype, an audio-visual program, or an interactive videodisc. The students have to go through all relevant stages of development, from feasibility study to implementation phase, and finish with a formative evaluation of the prototype.

Description of the individual approach
The student work is supervised and supported by a staff member of the Division of Educational Instrumentation. Technical support for programming or audio-visual work is provided by the laboratory of the Department of Education. However, the work of the students is generally highly individual in nature. This individual approach forces the students, in a very limited period of time, to acquaint themselves with the subject area in which their prototype has to operate. The acquired knowledge has to enable them to communicate with subject area specialists in outside agencies, which are the target groups of their prototype.

Results
A major characteristic of the projects is lack of time for adequate completion. Communication with subject area specialists takes much more time than planned and the specialists keep changing their minds about the courseware specifications. Even if evaluation milestones have been formally incorporated into the development process, these kinds of problem still arise.

Even with student designers highly skilled in applying educational technology and specially prepared for communication with the target group, it is very difficult to do formative evaluation as part of the design process, in a proper way.

The Cost of Courseware Development
Before we can make generalizations based on the Dutch experiences described above, another important issue should be considered: the cost of courseware development.

Courseware is very expensive to develop and produce. The industrial approach to courseware development was based on the assumption that the developed software would to some extent be portable. Availability of extensive documentation of functional and technical specifications and a prototype, in addition to the final product, would facilitate transfer to other hardware systems. Development, production and maintenance costs could thereby be brought down. However, as the transfer of courseware to other systems has not yet happened for packages developed under MS-DOS, and as maintenance of the courseware is not yet an issue, the value of this approach cannot be
judged at this moment. In order to be able to decide whether the industrial approach does result in cheaper programs, we should immediately begin to collect data about the cost of courseware development. On the other hand, the more individual approach followed by the students of the University of Twente, has its merits, too. The first development phase, up to prototype realization and formative evaluation, is cheaper than in the industrial approach.

**Interpretation of Dutch Experiences in Courseware Development**

In The Netherlands, a considerable amount of money has been invested in the development of a methodology for courseware development and production. Different approaches have been tried out, but no solution has been found which is satisfactory overall. Neither a strong project management, nor individuals with highly developed skills, has been able to keep development projects within time limits and meet the wishes of target group customers within the specifications agreed beforehand.

The main problem appears to be communication between developers and the target group in the time-consuming formative evaluation. In almost all projects, formative evaluation was not done according to plan, even if these plans were exactly scheduled.

**How to Proceed?**

Courseware is still a new kind of product for most teachers, with interactivity as its main characteristic. To describe beforehand, in detailed specifications, what courseware should do, remains very difficult. In order to improve the tuning of a program to the needs of the target group, regular formative evaluation of partial courseware products seems to be necessary. This evaluation should be done on several occasions during the development process. However, in practice, formative evaluation activities “are expensive and time-consuming if approached seriously” [4]. Experience shows that these activities usually get very low priority. And even if the formative evaluation process is well planned and executed, a feeling of ‘independence’ or ‘professional creativity’ makes developers and programmers not really receptive to the results of formative evaluation. What can be done about this?

**Two Options**

In the face of the dilemma of how to give formative evaluation its proper place, two options can be pursued:

(1) to put even more emphasis on formative evaluation in the development process, or

(2) to reduce the importance of formative evaluation in the development process.

It seems likely that time will keep putting constraints on the courseware development process and that creative people’s attitudes will be difficult to change. Therefore, pursuing the first option may not lead to real solutions. Some improvement may be expected if people with a very strong opinion about the necessity of formative evaluation and with highly developed skills in educational technology, take the lead in the development process.

On the other hand, the importance of formative evaluation might be reduced if we were able to define confidently the global didactical framework in which courseware is used.

The importance of formative evaluation in the development process might be further diminished, if the final presentation of a courseware package on the screen was not unchangeably fixed by the developers, but could be reshaped by the end-user, via installation options, or other means. For example, screen lay-out, specific phrasing of the content and ways of interaction, might be changed.

In view of the above, it seems most productive to pursue solutions in two directions:

(1) development of courseware which fits traditional didactical approaches and is not too complex to be used and modified by inexperienced teachers, and

(2) development of teacher toolkits, enabling modification of partial products to fit the needs of a specific target group [6].

Earlier, we have discussed the logic behind limitation of the didactical focus of courseware development projects, as indicated by the first point. The recommendation embedded in the second point will be further examined below.
Teacher Toolkits

Many authoring languages and authoring systems have been developed in the past. This ‘authoring’ approach did not bring the problem of courseware development to a solution. Partly, this is due to the limited scope of the didactical framework of the authoring systems offered, partly to lack of time and imagination of teachers in the creation of new materials. Creativity of teachers, with regard to development of more traditional learning materials, is optimal when adapting existing materials to their own needs. These adaptations are quite common and easily executed in the case of printed materials. The observed lack of effort, to date, of teachers in adapting courseware, may be a result of limited skills and tools. Tools should be provided to allow the same possibilities for adapting courseware products as for adapting products created in other well established media.

Conclusion—Is Courseware Development at the Crossroads?

To reduce development costs, an industrial approach to courseware development should be taken, in which a development methodology is followed, based on detailed specifications. However, the target group does not yet appear to be ready to specify, beforehand and in sufficient detail, what kind of courseware product is needed. In an industrial setting, marketing and advertising pave the way for the products and create a market. In education, however, bringing about a change of attitude and innovation is not as easy as selling industrial products. Tradition, conservatism and inertia are factors that cannot be changed over-night. The strategy for courseware development should therefore take these inevitable circumstances into account.

The crucial element in current development methodologies is formative evaluation. Improvement of such evaluation or reduction of its importance, are the two choices which can be made. Improvement of the formative evaluation procedure can be accomplished by concentration of evaluation activities within the first phase of the courseware development process—up to the prototyping of the product. During this first phase, a person with a strong background in educational technology should be in charge in the development. The importance of formative evaluation can be reduced by limiting courseware use to didactical situations familiar to teachers, and also by providing toolkits to end-user teachers, for adaptation of the courseware product to their own needs.

The use of courseware will evolve and mature in education. At the same time, the methodology for courseware development will evolve and change. The important lesson is that the direction of courseware development will be determined by teachers’ choice. Crossroads, as described in the first part of this paper, are crossroads for teachers: they will install the road signals. Courseware development will have to obey those signals.

References

Onderwijs en Informatietechnologie, Enschede, PMI-reeks nr. 54, 1988).

