ED 384 216 FL 023 017

AUTHOR Dodigovic, Marina
TITLE Multimedia CALLware: The Developer's Responsibility.
PUB DATE Mar 94
NOTE 8p.
PUB TYPE Viewpoints (Opinion/Position Papers, Essays, etc.) (120)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Audiovisual Aids; Audiovisual Instruction; *Cognitive Processes; Communication (Thought Transfer); *Computer Assisted Instruction; *Computer Software Development; *Course Content; Course Objectives; *Courseware; Foreign Countries; Instructional Films; *Interdisciplinary Approach; Role Perception; Second Language Instruction; Teacher Role
IDENTIFIERS *Germany

ABSTRACT

The early computer-assisted-language-learning (CALL) programs were silent and mostly limited to screen or printer supported written text as the prevailing communication resource. The advent of powerful graphics, sound and video combined with AI-based parsers and sound recognition devices gradually turned the computer into a rather anthropomorphic partner, especially as far as language learning is concerned. This advance has revolutionized the role of the courseware developer in many respects, making it extremely complex. He or she must be a linguist as well as a language teaching specialist. Additionally, programming skills might be required as well as designer know-how, basic facts of screen ergonomics, and the metacommunicational and cognitive aspects of CALL. A little bit of an artist is a part of the role, too. If the program is to use knowledge representation, the CALLware developer has to be a knowledge engineer as well. If the program is to have integrated videos, the developer may have to do the job of a film director. The importance of the human factor behind the machine increases. Bremen Multimedia Initiative explores all these aspects of multimedia courseware development by gaining experience through the development of new applications. (Contains 10 references.) (Author/CK)

**Marina Dodigovic**
Hochschule Bremen

# Multimedia CALLware: The Developer's Responsibility

## *Abstract*

The early CALL-programs were silent and mostly limited to screen or printer supported written text as the prevailing communication resource. The advent of powerful graphics, sound and video combined with AI-based parsers and sound recognition devices gradually turned the computer into a rather anthropomorphic partner, especially as far as language learning is concerned. This has revolutionised the role of the courseware developer in many respects, made it extremely complex. He or she must be a linguist as well as a language teaching specialist. Apart from that, programming skills might be required, but also designer know-how, basic facts of screen ergonomics, but also the metacommunicational and cognitive aspects of CALL. A little bit of an artist is a part of the role, too. If the program is to use knowledge representation, the CALLware developer has to be a knowledge engineer as well. If the program is to have integrated videos, the developer may have to do the job of a film director. The importance of the human factor behind the machine increases. Bremen Multimedia Initiative explores all these aspects of multimedia courseware development by gaining experience through the development of new applications.

## *Multimedia Design*

The development of multimedia courseware or CALLware is a challenging task requiring a vast amount of knowledge in various disciplines. Apart from subject expertise and didactic know-how on the one hand or the whole range of film production skills on the other hand, the now expanding complexity of courseware design demands a much greater awareness of detail as well as of harmonious interaction of altogether different elements presented on one screen. Sometimes, these elements can be treated separately, but the layout as a whole would still need a judgement of a person whose holistic approach would provide for the constellation of separately designed pieces. That is to say, the development team should be supported by a person with interdisciplinary experience.

The design factors, sometimes also called the human factors, are of varying origin and significance. They are usually classified as text, colour, image, animation, video, sound and interaction. Three of them are dependent on the others: colour, animation and interaction. There is no colour or animation apart from the frame it is attributed to as there is no interaction without basically non-interactive elements. Nevertheless, the first two of the above three factors are, together with all the others very important encoding devices, used to transmit or enhance certain metatextual messages, e.g. about the function of different screen parts. Therefore we shall treat them separately. Interaction is a goal in itself, especially with educational software, so it deserves a passage of its own as well.

## *General design principles for the development of CAL-software*

The goal to create a system that is easy for the learner to use is by all means important for software design in general, but it is especially critical for training and instructional applications. This is due to two reasons. Firstly, in most cases the student is a novice who will not use the program for a long period of time (as does a person using a word processing system, for example). Secondly, in instructional applications, cognitive resources need to be devoted to the content and not to trying to find out one's way through the instructional software. For these reasons the instructional designer must focus on developing a system that is simple and straightforward in its use.

Among other things, the system should exhibit the following characteristics:

At any given point the user should understand what is being presented, what they are required to do or have the option of doing, and how to accomplish their current goal. The system should also display what the system is currently doing, if anything (e.g. an hourglass when the system is working and the user must wait).

### *Design elements*

Let us now proceed to single elements whose characteristics can make the system either user friendly or the opposite.

### *Text*

In the early eighties CALLware design was chiefly based on behavioural theory and programmed learning from the psycho-pedagogical point of view, whereas from the point of view of screen design ergonomics it was more or less confined to text presentation.

Enthusiastic amateurs flooded the software market with screenfuls of endless, cramped text, messages running over messages, boxes popping open practically everywhere, blinking in shrill colours right into the mind of a desperate novice, who at that point spontaneously decided to hate computers all his life long.

A much better starting point for screen design is anchored in educational theory, behavioural as well as cognitive. Both theories agree to a certain extent as far as computer based instruction/learning is concerned. Most cognitive theories support some form of information processing, where mental events are described in terms of non-observable transformations of information from input (stimulus in behavioural theory) to output (response) (Hannafin,Rieber,1988). Behaviourally based principles for designing programmed instruction include objective specification, empirical testing, self-pacing, overt responding, immediate feedback, controlled sequencing, small step size, low error rates, prompting and confirmation. Cognitive learning foundations include encoding and retrieving as two basic sub-processes within an information-processing system, which implicitly relies on small basic bits of information that have to be elaborated and organised by the learner, prompted by the events of instruction.

The implications for screen design are the same: nice clear screen with fragmented units of information that belong together is preferred. Moreover, fragmentation of messages is introduced into instructional science in connection with media in general (media being anything from book over lecture, audio vision and film to computer) by John H. Mason (Mason,1988). He defines a fragment as "a recallable incident which spans a single block of time" whose "content...can be agreed upon by two observers of the event" (Mason,1988:198). Supposedly influenced by semiotic (Eco,1973), cognitive (frames), and knowledge representation (scripts) theories and close to film domain, but also close to constructivism (construal), Mason tries to establish the basic unit of a system, which would be universally identifiable on the paradigmatic level. On the syntagmatic level, these "fragments are then vowen together to make a story about the experience, which is what is often called construal or making sense" (Mason,1988:195). Teachware produced in accordance with this hypothesis should "be prepared so as to focus students' attention on the need to reconstruct the content for themselves from the fragments they can recall" (Mason,1988:207). In video media this is, in Mason's opinion, achieved by zooming, split screen, colour separation, short segments, animation, ability to stop/replay, change of visual frame, subject and pace.

Fragmentation or segmenting of information has also strong claims on text presentation in the context of computer based instruction or learning. The research does not leave it unconsidered. For instance, Mark G. Gillingham supplies very useful guidelines for text presentation in computer-based instruction. He introduces five basic terms describing the chief text features: length (how much text on screen at a time), amount (how many single screens), rate (how fast the single screens exchange), colour (as a factor influencing readability and comprehension) and structure (logical organisation). As regards the length of text he quotes the conclusion of International Reading Association: "Display as many legible double-spaced lines as the screen will comfortably hold" (Gillingham,1988:1). Rate, colour and structure of text must be such as to enable eye-fixation and comprehension. Logically well organised small items and individually adaptable rates are preferred.

Sallie Gordon of the University of Idaho compiled a guideline to courseware design relying on several other experts in the field like Mayhew, Jones, Tullis. Here are some useful suggestions for text and static screen design.

- Don't try to place too much information on the screen. Alphanumeric characters shouldn't take up more than 25-30% of the total screen space.

- Include all information that is necessary.

- Group characters so that they are moderately packed.

- Group the items logically.

- Leave adequate space around each group.

- A smaller number of groups is better than a larger number of individual items or a larger number of groups.

- Minimise the complexity of the layout.

- Maximise visual predictability. Put the same items at the same spots throughout the program.

- Use redundant coding (e.g. colour and shape).

- Use symbols that are consistent with the user's previous knowledge. Strive to use words and icons.

- Don't use hidden buttons except for the hot parts, but then change the cursor form.

- Start in the left-hand corner of the screen, as this is where people habitually begin search and reading.

- Use upper and low case for text. Use minimal or no text that is all upper case.

- Use blinking only in emergencies and critical situations.

- Messages should be brief, concise, specific, helpful, comprehensible. The following examples illustrate poor and improved messages:

Poor:
The processing of the text editor yielded 23 pages of output.

Improved: Output 23 pages.

Poor: Cannot exit before saving.

Improved: Save file before exiting.

Poor: Bad\illegal\invalid file name

Improved: Maximum file name length is 8 characters.

### *Use of colour*

In the same book Gordon offers advice on the use of colour:

- Don't use bright colours such as red, yellow, pink, orange for large areas. Use them only in small amounts to gain user's attention.

- Don't use too many colours on a screen, unless it is a full photo or video image type of situation. Some authors suggest limit as being 9.

- Use colour coding and be consistent in colour use. For instance, if light blue is used for video control buttons in one part of the program, it should be used so consistently throughout the program.

- Limit the colour coding to eight distinct colours.

- Use colours to draw attention, communicate, or organise information. Showing certain types of information in a particular colour can enhance finding the right piece of information.

- Do not use colour images or text on colour background. Use colour on achromatic background such as black, grey, white or use achromatic images on colour background.

- Avoid using bright blue for text or small, thin-lined graphics. We are not good at distinguishing blue objects from their background, and our eyesight for blue decreases with age.

- Avoid the combined use of opposing colours, highly saturated colours or colours far apart on the colour spectrum. This applies within one screen as well as across screens.

### *Image and Animation*

Image and Animation are related to one another and together they play an important part in courseware design.

Alphanumeric screen terminal was the first output device to replace or accompany the printer. It introduced little novelty into the world of basically textual interactive media. However, it was soon followed by graphic station, which led a step further into the fantasy world of senses and communication. It offered image and animation, facilities to go hand in hand with modern cognitive theories. Studies on the instructional effects of visuals have abounded. The most prominent theoretical support for pictures is the dual-coding hypothesis, which suggests that humans possess both visual and verbal encoding mechanisms. Fundamental differences between learning styles have been identified, based upon the relative abstractedness of to-be-learned information: the more concrete, the easier to picture and remember; the less concrete, the harder to picture and remember (Hannafin, Rieber,1988).

Audio-visual language teaching method anticipates in part this development. It tries to present the matching non-verbal context of a language situation in the form of pictures and videos. It suppresses, however, and sometimes even completely bans the native language verbal explanation, which eventually does not facilitate comprehension. At its best it is represented in the PLATO system. The first PC-based CALL programs were chiefly written by language teachers, amateurs in programming, and therefore based on text mainly. Professional programmers, strange to language teaching, on the other hand preferred to bring in graphics and animation as the potential of the medium. So for example, John Higgins, basically a language teacher, had his first CALL programs developed by his wife, Muriel Higgins, a computer specialist. These early programs are enriched with graphics and animation, whereas the later ones, produced by John Higgins himself, who in the course of time acquired basic programming skills, are exclusively text-based, using however the benefits of colour and animated text-shifting.

It would, of course, be wrong to use graphics and animation just because they are possible, without any real reference to the content. In my program PREPOSITIONS I use graphics to demonstrate the exact field of meaning of German spatial prepositions and animation to show the difference between static place and dynamic direction. I also use animated text to visualise the concept of word formation.

Russian CALL programs written in mid and late eighties show perfection in both glotodidactic and programming aspects. Designed by teams of language teaching and computing specialists, they achieved a perfect balance between verbalising and visualising, between formulation and meaning, between colour and movement, without ceasing to be interactive, without taking away the chance for the learner to assess his or her own answers.

### *Sound*

We all know that an important part of natural language is sound. The early CALL programs were in that respect frighteningly silent. That is in the first place why certain language specialists decided to ban the computer into the isolation of individual drill-and-practice.

Two major achievements turned against this scheme: speech synthesis and digitised speech. CD-ROM came just in time to support storing the enormous amount of information immanent to speech. Software to process digitised speech made it possible to alter the speed without the change in pitch and vice versa. Convenient dialect and sociolect variation was made possible. Background noises can be inserted, altered or completely erased, depending on the difficulty of the educational task or the preference of the user. Sound mixing became that way one of the standard component parts in the process of developing language software.

Checking the student aural input by means of computer also became possible. Through sound facilities the human-computer communication made an important step further towards being 'natural'.

However, audio design for instructional purposes has not been fully explored yet. Felicitas Rühlmann says for instance in a paper of hers that the audio-voice should be pleasant to hear and in hi-fi quality. She also stresses the importance of aural messages in that she quotes Nick Corston, according to whom we remember:

- 10% of what we read (text)
- 20% of what we see (image)
- 30% of what we see and hear (text, image, audio)
- 70% of what we see hear and do (text, image, audio, interaction)

### *Interaction*

Interaction is maybe the most important of the so called 'human factors'. When a user sits down to interact with an instructional system, it should exhibit a quality known as transparency. That is to say that at least two things should be obvious: the nature of the program (what it is used for) and how to interact with it.

In addition to this general rule Gordon mentions some more useful guidelines for designing interaction:

- At the beginning present an overview of how the system works, and how the learner interacts with the system.

- Provide an advance organiser.

- Make use of previously existing knowledge

- In general, a graphical interface would be better than command language input.

- Use consistent design elements from one screen to the next.

- For most hypermedia systems make the underlying organisational structure hierarchical.

In summary, a design team must create a simple and obvious system for the user interface, and stick to that system consistently throughout the instructional program.

### *User Testing*

Designing a user-friendly system requires effort and creativity. However, even if they are at work, at some point it becomes impossible for the designer to see the product with the eyes of a novice. This is why the prototyping and user testing becomes extremely important for the development of courseware.

According to Gordon, the questions that user testing should yield answers to are about the learner's attitude towards the program, the usability of the program, the efficiency (do the persons learn from it?) and whether it affects the user's attitude towards the program.

The first variable is the reaction of the users toward the program. The most common method for measuring this variable is simply by asking the users (e.g. a questionnaire). The second variable is usability. Here are the methods for measuring this variable:

- Ask learner after interaction how easy the system was to use.

- Measure errors committed in using the instructional system.

- Note verbal expressions during performance.

- Measure performance on tasks.

Comprehension, retention and transfer belong to the third variable. Comprehension can be measured by asking the learner to verbalise during or after the session. The other two are measured by knowledge tests and tasks.

Attitude towards the subject matter is the fourth variable. Sometimes the instructional programs can increase the motivation for a certain activity.

### *Conclusion*

The appearance of hypermedia systems in the late eighties enabled seamless fusion of text, video, computer graphics, animation, sound, natural language processing, indeed any technologically feasible form of presentation. Authoring tools and systems, such as ToolBook, IconAuthor or Authorware professional made the creation of most exciting intelligent multimedia software accessible to practically anyone willing to undergo the now reasonable software and hardware expenses. This, in turn, increased the demand for knowledge and responsibility on the part of the developer who is expected to have reached mastery in various disciplines.

### *REFERENCES:*

Eco,U.:Kultura - informacija - komunikacija. Nolit, Beograd, 1973

Hannafin, M. and Rieber, P. L., Psychological Foundations of Instructional Design for Emerging Computer-Based Instructional Technologies: Part II. ETR&D 27, 102-114 (1989)

Gillingham,M.G.:Text in Computer-Based Instruction: What the Research Says, Journal of Computer-Based Instruction 15(1), 1988, 1-6

Gordon,S.: Computer Interface Design Guidelines, CALICO Annual Meeting "Human Factors", Flagstaff, March 1994

Higgins, J. Language Learners and Computers, Longman, London (1988)

Mason,J.H.:Fragments: the implications for teachers, learners and media users/researchers of personal construal and fragmentary recollection of aural and visual messages, Instructional science 17, 1988, 195-218

Obermeier,K.K.:Natural Language Processing Technologies in Artificial Intelligence, Ellis Horwood Limited, New York, 1989

Rumelhart,D.E.:"Schemata: The Building Blocks of Cognition" in Spiro, Bruce, Brewer (Hrsg.) Theoretical Issues in Reading Comprehension. Perspectives from Cognitive Psychology, Linguistics, Artificial Intelligence, and Education. Laurence Erlbaum Associates: Hillsdale, N.J., 1980, 33-58.

Stratil,M. et al.:Computer Aided Language Learning with Speech Synthesis: User Reactions, PLET 24(4), 1987, 309-317

Trollip, S.R. and Lippert, R.C., Constructing Knowledge Bases: A Process for Instruction, in Hancock, P.A. and Chingell, M.H. (Ed.) Intelligent Interfaces, Elsvier Science Publishers B.V., North-Holland (1989)