ED 382 014 FL 022 912

AUTHOR	Ruhlmann, Felicitas
TITLE	Designing Templates for Interactive Tasks in CALL Tutorials.
PUB DATE	94
NOTE	17p.; Paper presented at the Meeting of EUROCALL (Karlsruhe, Germany, 1994). Figures may not reproduce well.
PUB TYPE	Guides - Non-Classroom Use (055) -- Speeches/Conference Papers (150)
EDRS PRICE	MF01/PC01 Plus Postage.
DESCRIPTORS	*Computer Assisted Instruction; *Computer Software; Courseware; Foreign Countries; Higher Education; *Instructional Design; Multimedia Instruction; Multiple Choice Tests; Problem Solving; *Records (Forms); Second Language Instruction; *Second Languages; Task Analysis; Test Items; Tutorial Programs

ABSTRACT

The development of templates for computer-assisted language learning (CALL) is discussed, based on experiences with primarily linear multimedia tutorial programs. Design of templates for multiple-choice questions and interactive tasks in a prototype module is described. Possibilities of enhancing interactivity by introducing problem-oriented elements into multiple-choice questions, and by designing multi-layered tasks, are also examined. The prototype was integrated as a model into a final year undergraduate course that aimed at the production of low-cost courseware by students with a view to counterbalancing reduced lecture time with tailor-made CALL modules. The question is raised as to what extent students themselves can be expected to design sophisticated innovative tasks and which pitfalls should be avoided. (MSE)

DESIGNING TEMPLATES FOR INTERACTIVE TASKS IN CALL TUTORIALS

Felicitas Rühlmann

University of the West of England, Bristol

Abstract

This paper is based on experiences with CALL template development for mainly linear multimedia tutorials. The design of templates for multiple choice questions and interactive tasks in a prototype module is described. Possibilities of enhancing interactivity by introducing problem-oriented elements into multiple choice questions, and by designing multi-layered tasks, are also examined. The prototype was integrated as a model into a final year undergraduate course which aimed at the production of low-cost courseware by students with a view to counterbalancing reduced lecture time with tailor-made CALL modules. The question is raised as to what extent students can themselves be expected to design sophisticated interactive tasks and which pitfalls should be avoided.

Introduction

Against the background of increasing student numbers and reductions in class contact time for lecturers, the Faculty of Languages and European Studies at the University of the West of England decided to adopt new approaches to teaching and learning. In line with this initiative, a CAL option was offered to final year undergraduate students in German with Information Systems. The option was set up as a starting point to explore the possibilities of in-house courseware design and development. The [illegible]n was to overcome a lack of off-the-shelf CAL materials suitable for integrating into the curriculum and the hurdle of up- front investment of time and money into professional courseware production. The CAL option concentrated on the design of self-study multimedia CAL tutorials that could add to or substitute parts of our current undergraduate teaching. Students were encouraged to reflect on concepts and principles of CAL design, as well as on the content of their own studies and on teaching methods, so that they could apply an improved methodology in their own design projects. Each group of students produced a design specification and programmable-ready material, i.e. a storyboard, graphics design sheets and flow diagrams for interactive tasks for the development of a short CAL module. In order to provide students with some guidelines and examples for screen design and interactive tasks, different existing programs were compared with a template and prototype module that had been produced by the author of this paper. The template was produced with the aim of achieving a certain amount of unity between modules designed by different teams. It was felt that it was important to develop a house style which ensured standardization and consistency of the user interface in all future modules. The design of customisable "shells" for learner activities within this general module template and their reflection in students' design projects will be the core discussion of this paper.

Learner interaction in template and prototype

The prototype is a flexible but guided self-study tutorial designed to impart factual knowledge through audio sequences which are accompanied by on-screen graphics. At the same time, key points are displayed on screen in a fixed textbox. At irregular intervals learning sequences which last between five and ten minutes are broken up by different types of interaction and appropriate feedback. A variety of fixed screen types are employed, such as pause screens, menu screens, information screens, presentation screens, interaction screens. Screens were designed for the user to form a conceptual model, with the templates, the human-computer interaction techniques and the terminology being consistent across all sections and chapters in the same course.

Two forms of structured learner interaction were included in the prototype: multiple choice questions and interactive tasks. Activities were classified in three main categories:

1. Multiple choice with one or more correct answers;
2. Matching or sequencing tasks;
3. Free text entry.

A major design consideration was to maintain learners' attention and to enhance problem solving skills through the given tasks. To achieve this, exercises from different categories were combined into more complex tasks, and text in instruction lines or help windows was phrased carefully in order to avoid typical sentences like "Click the buttons below in turn" (Soper and MacDonald, 1994).

Multiple choice questions

Two different shells for the development of multiple choice questions were set up. In shell one the courseware developer, editor or tutor is requested to input a number of variables and strings: respective field numbers that indicate the correct answer(s), a maximum of three lines of question text, between two and four possible answers to the question (i.e. student choices), feedback text for each possible answer, and up to eight lines of text for a global feedback explaining the correct answer, possibly including some further information on the subject. Shell two is essentially the same but with an additional feature. For a multiple choice question to become more complex, various on-screen graphics components with underlying hot-spots can be added and a correct final screen display can be assigned within the shell (figure 1).

figure 1

(template for multiple choice question with additional hot-spots)

The aim was to encourage learners to use their own initiative and imagination whilst attempting to answer a question. For instance, learners may be given a statement indicating the outcome of an operation for which several mouse clicks or key presses were required. To be able to verify the statement, they may need to retrace the steps leading to the result, i.e. operate a simulated device on screen in order to find the answer to the given question. In the prototype which teaches basic knowledge of computers together with German terminology, an example for this would be the question:

"Auf der Festplatte des Computers auf diesem Bildschirm ist das Programm WordPerfect installiert.

- *Ja*
- *Nein."*

The learner may now either click on the answer buttons at random, or click on the "advance" icon to get the final feedback with the correct answer, or input the appropriate commands on the simulated computer on screen until the correct setup is displayed which provides the answer to the question. An inquisitive learner is expected to opt for the third possibility.

Another possibility for extending multiple choice questions is embedded in the structure and routing of the prototype module. To find the correct answer to a question, learners may need to retrieve information located elsewhere in the program. For example, in "Computer auf Deutsch" Konrad Zuse is mentioned in connection with the introduction of relays and the binary system. A more detailed explanation of how the binary system works is given in a different chapter and the learner is prompted to explore this route. Therefore, if the learner has difficulties answering the following question:

"1 Bit ist die kleinste Informationseinheit. 1 Bit hat

- *den Wert 1,*
- *den Wert 0,*
- *den Wert 0 oder 1,*
- *die Werte 0 und 1."*

he/she may get a prompt, then set a bookmark at this particular screen, go to the prompted section in the tutorial (i.e. section 2, chapter 3), find some additional information about the binary system, return to the question via the bookmark and then confidently complete the task (figure 2).

figure 2

(screen with multiple choice question)

In multiple choice questions, the learner is given feedback immediately after he/she has made his/her choice. If the answer given by the learner is judged as correct, a beige touch-sensitive square in the answer box is displayed in green. If the answer given is judged to be wrong, the square turns red. At the same time, differentiated feedback concerning each respective answer given by the learner is displayed in a feedback box on screen below the question and answers field. If the learner chooses to avoid dealing with the question, advancing through the program will nevertheless provide him/her with the correct answer. This final feedback can either be the same as the feedback for the correct choice or it can be an additional reinforcement, perhaps containing more explanations. In the template students will get the additional global feedback after individual feedback for a particular choice.

Interactive tasks

Other examples for more complex interaction are tasks that involve simulation. They are particularly appropriate for operating complicated machinery, but also applicable for problem solving. Here, the learner may be allowed to find the solution to a problem via various different routes, each route including a number of possible key presses or mouse clicks.

In interactive tasks, the problem of feedback is more difficult to deal with than in multiple choice questions. If the student is supposed to learn by discovery, there is no point in giving feedback after every single learner input. The problem was addressed as follows. On entering an interaction screen, the learner will be given text and graphic with a task and an operating instruction. This structure is applied for both straightforward and more complex tasks. The following example shows a less complex task:

"Erinnern Sie sich an die Entwicklungsstufen des Computers. Setzen Sie die richtigen Begriffe in die richtigen Stufen von 1 bis 6."

Operating instruction:

"Ihre Eingaben bitte. Dann auf WEITER klicken."

After completion of the task the learner clicks on the "advance" icon. Only then does the program judge the learner's activity and give appropriate feedback. That is, the learner has complete freedom of changing his or her setup until he/she confirms the final setup by clicking on "advance". If then the setup is judged to be wrong, the learner is given another chance without any particular interference of the program. The instruction line reads:

"Bitte weitere Eingaben."

and the learner may choose to re-try the task or to proceed by clicking on the "advance" icon. On completion of an incorrect attempt, however, the setup is not reversed to the starting point. If for a new try a starting point has to be indicated, the learner will be advised of what the starting scenario was, e.g. *"Start from X and explore another route."* If the second try turns out to be wrong, a help message is displayed which points the learner in the right direction. If after the third attempt the result is wrong again, the program provides the correct screen setup together with a detailed explanation of how to get there. The learner may click on the "help" icon any time within an interactive task. Help messages are carefully phrased in order to point the learner in the right direction, but they are not to be confused with operating instructions (figure 3).

figure 3

(screen with help message in interactive task)

Within this structure it is left to the learner's discretion simply to advance through the task or to go through the complete learning experience. In this form of interaction it is important that the learner's efforts are not judged until a certain setup is complete. It remains to be proven, however, that the experience of getting a complex setup right is more satisfying for learners than getting detailed feedback on every single input. In the structure of tasks as described above, a certain amount of anticipated wrong setups could of course be addressed in a more detailed feedback after completion of the task. With an increasing complexity of tasks it has to be carefully judged whether the programming effort for this can be justified. This becomes obvious when we look at the programming shell for an interactive task, bearing in mind that courseware development is meant to be done by students or tutors with very little or no programming skills (figure 4).

figure 4

(template for interactive task)

In this shell, courseware developers, editors or tutors are required to input variables, text and graphic displays. For a sequencing or matching exercise a set of numbered hot spots (I) must be paired with their respective counterparts (II) to indicate the correct screen setup. Unlike in multiple choice templates, the editor need not enter any feedback text on each step the learner takes, a number of prompts are employed instead which suggest that the learner has not yet completed the task correctly. Text input, however, is required in the template for feedback after the learner's successful completion of the task, also for a help message and for the global feedback at the end of the interaction. Matching graphic displays for each learner input and for the correct screen setup may be assigned by their outline number according to their position in the relevant graphics file, the editing of graphic objects, however, may only be done within the graphics file itself.

Questions and tasks designed by students

Although students' CAL projects showed thoughtfulness and creativity in terms of content and screen design, details concerning lesson flow, routing and interactive tasks proved to be particularly weak in all projects. Particularly the flow diagrams had not been thought through accurately, and they were largely unsuitable for programming. Moreover, students tended to avoid feedback or reduce it to a minimum in the design of exercises or they omitted exit and/or help facilities. The restriction to pure courseware design without any chance for hands-on implementation prevented students from checking the feasibility of their designs. The development of prototypes would have helped discover design errors. Given the time constraints in which the projects had to be completed (two class contact hours per week over 10 weeks were allocated for this course), students would have benefitted from a given range of prepared templates for interactive tasks which they could have used or adapted to fit into their own projects.

Time constraints were not the only problem reflected in students' design documents. Another drawback was the lack of appropriate model CAL tutorials with a more complex interactive approach rather than just a repetition of drill and practice exercises or pure databases of information. During the design process, students explored ideas taken from computer games (project one: "European monuments"), traditional multiple choice tests (project 2: "German driving regulations") and pattern drills (project 3: "German political system") and incorporated them into their presentation of factual materials on screen. All interaction was put forward in question and answer form, serving for testing purposes after a series of instruction screens. Rather than venture into new forms of interactive tasks, students adopted established models of programmed instruction.

Conclusion

I believe that interactive tasks in a computer-based tutorial are crucial for its success as a medium of self-instruction. Interactive tasks are not just a means of self-checking or testing newly acquired knowledge, but generally enhance learner participation. A variety of types of interaction may be employed in a tutorial to add to learner motivation. Tasks may range from the repetition of factual knowledge to the use of intellectual abilities for problem-solving. An appropriate form for doing the latter may be designing the interaction in the form of a gradual reduction of the problem through increasing additional information and help facilities in each task and throughout the course. The careful selection of playful elements such as fun graphics, competitive elements such as time limits to solve a problem, and the creation of situative contexts for interactive tasks are equally important to enhance the learning process. However, the desired complexity of interaction must be considered against the time and cost involved in the in-house production of a self-study tutorial, and the result can only be a compromise.

References

Andre, T. and Phye, G. (1986) (eds), Cognition and Classroom Learning , San Diego CA, Academic Press.

Carroll, J.M. (1991) (ed), *Designing Interaction*, Cambridge, Cambridge University Press.

Cates, W.M., Fontana, L.A. and White, C.S. (1993), 'Designing an interactive multimedia instructional environment: The Civil War Interactive', *Association for Learning Technology Journal*, **1**, 2, 5-16.

Jonassen, D. (1988) (ed), *Instructional Designs for Microcomputer Courseware*, Hillsdale NJ, Lawrence Erlbaum.

Kulik, J.A. and Kulik C.C. (1988), 'Timing of feedback and verbal learning', *Review of Educational Research*, **58**, 1, 79-97.

Scott, D. (1991), *Human-Computer Interaction: A Cognitive Ergonomics Approach*, Chichester, Ellis Horwood.

Soper, J.B. and MacDonald, A.B. (1994), 'An interactive approach to learning economics: the WinEcon package,, *Association for Learning Technology Journal*, **2**, 1, 14-29.

Sutcliffe, A. (1988), *Human-Computer Interface Design*, London, Macmillan.

figure 1

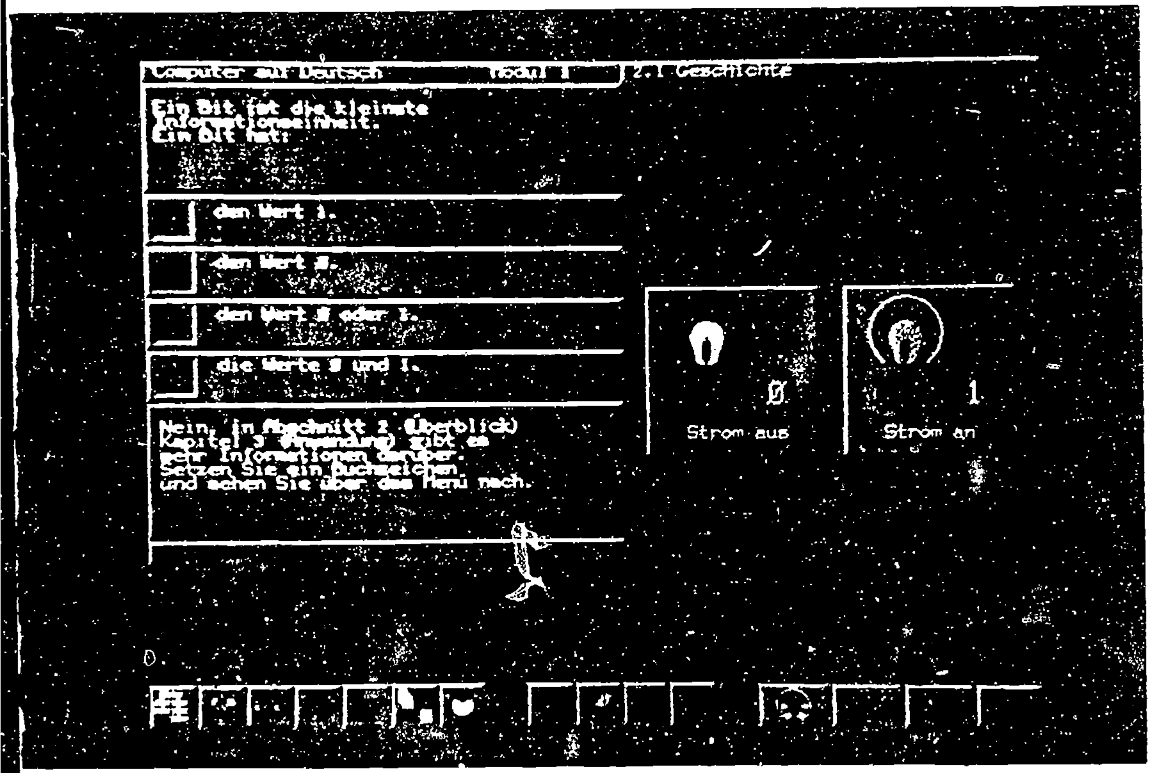
Only items in bold print need to be edited by the tutor.

Component: MC 5, dir command (PC)

Object list: (50 objects)

```
 1. content *DEFAULT* GR1: ALL HOT SPOTS ACTIVE
 2. * *** ASSIGN CORRECT FINAL SCREEN
 3. assign:right_page[1]◂16
 4. *
 5. * *** ASSIGN CORRECT ANSWER NO.
 6. assign:answer◂-1
 7. *
 8. * *** QUESTION
 9. assign:question[1]◂"WordPerfect  is installed"
10. assign:question[2]◂"on  the hard disk "
11. assign:question[3]◂"of  your PC on this screen."
12. *
13. * *** CHOICE 1
14. assign:choice_A[1]◂"True"
15. assign:choice_A[2]◂""
16. *
17. * *** CHOICE 2
18. assign:choice_B[1]◂"False"
19. assign:choice_B[2]◂""
20. *
21. * *** FEEDBACK  FOR CHOICE 1
22. assign:feedback_A[1]◂"Well  spotted!"
23. assign:feedback_A[2]◂""
24. assign:feedback_A[3]◂""
25. assign:feedback_A[4]◂""
26. assign:feedback_A[5]◂""
27. *
28. * *** FEEDBACK  FOR CHOICE 2
29. assign:feedback_B[1]◂"Have  you paged your directory"
30. assign:feedback_B[2]◂"on  c: drive?"
31. assign:feedback_B[3]◂""
32. assign:feedback_B[4]◂""
33. assign:feedback_B[5]◂""
34 *
35  . * *** GLOBAL FEEDBACK
36. assign:global_feedback[1]◂"Correct  operation:"
37. assign:global_feedback[2]◂""
38. assign:global_feedback[3]◂"   To find WordPerfect"
39. assign:global_feedback[4]◂"   you had to enter the command"
40. assign:global_feedback[5]◂"   c:\dir. This gives you"
41. assign:global_feedback[6]◂"   a list of all files "
42. assign:global_feedback[7]◂"   in this directory."
43. assign:global_feedback[8]◂""
44. *
45. * **** GRAPHICS FILE NAME
46. assign:Gr_file_name◂"compgr1.cd3"
47. * **** GRAPHIC COMPONENT
48. assign:component◂"2.2.1"
49. *
50. content *DEFAULT* MC LOGIC (2 CHOICES) + HOT SPOTS
```

figure 2


Computer auf Deutsch | Modul 1 | 2.1 Geschichte

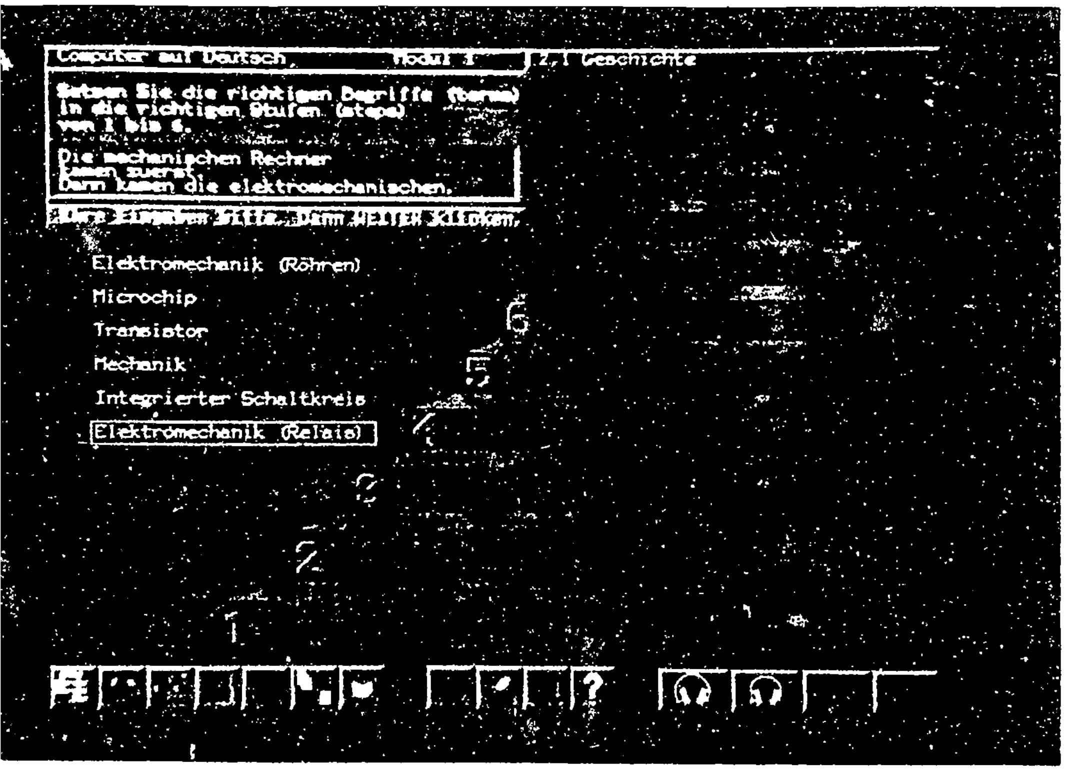
Ein Bit ist die kleinste
Informationseinheit.
Ein Bit hat:

- den Wert 1.
- den Wert Ø.
- den Wert Ø oder 1.
- die Werte Ø und 1.

Nein, in Abschnitt 2 (Überblick)
Kapitel 3 (Anwendung) gibt es
mehr Informationen darüber.
Setzen Sie ein Buchzeichen
und sehen Sie über das Menü nach.

BEST COPY AVAILABLE

figure 3



Computer auf Deutsch, Modul 1 | 2.1 Geschichte

Setzen Sie die richtigen Begriffe (terms)
in die richtigen Stufen (steps)
von 1 bis 6.

Die mechanischen Rechner
kamen zuerst.
Dann kamen die elektromechanischen.

Ihre Eingaben bitte. Dann WEITER klicken.

Elektromechanik (Röhren)
Microchip
Transistor
Mechanik
Integrierter Schaltkreis
Elektromechanik (Relais)

BEST COPY AVAILABLE

figure 4

Only items in bold print were edited by the tutor.

Component: IT Template TYPE 1 (correct order)

Object list: (61 objects)

1. * *** EDIT THE FOLLOWING LINES
2. *
3. * **** CORRECT HOT SPOTS (-999 if not required)
4. assign:anskeys[1]◄**-19**
5. assign:anskeys[2]◄**-21**
6. assign:anskeys[3]◄**-16**
7. assign:anskeys[4]◄**-18**
8. assign:anskeys[5]◄**-20**
9. assign:anskeys[6]◄**-17**
10. assign:anskeys[7]◄-999
11. assign:anskeys[8]◄-999
12. assign:anskeys[9]◄-999
13. assign:anskeys[10]◄-999
14. *
15. * **** MATCHING HOT SPOTS (-999 if not required)
16. assign:anskey_match[1]◄**-22**
17. assign:anskey_match[2]◄**-23**
18. assign:anskey_match[3]◄**-24**
19. assign:anskey_match[4]◄**-25**
20. assign:anskey_match[5]◄**-26**
21. assign:anskey_match[6]◄**-27**
22. assign:anskey_match[7]◄-999
23. assign:anskey_match[8]◄-999
24. assign:anskey_match[9]◄-999
25. assign:anskey_match[10]◄-999
26. *
27. content PROMPTS AFTER WRONG ANSWERS
28. *
29. * **** FEEDBACK FOR CORRECT ANSWER (enter text)
30. assign:feedback_D[1]◄**"Genau die richtige Reihenfolge."**
31. assign:feedback_D[2]◄""
32. assign:feedback_D[3]◄""
33. *
34. * **** HELP MESSAGE (enter text)
35. assign:question[1]◄**"Die mechanischen Rechner"**
36. assign:question[2]◄**"kamen zuerst."**
37. assign:question[3]◄**"Dann kamen die elektromechanischen."**
38. *
39. * **** GLOBAL FEEDBACK (enter text)
40. assign:global_feedback[1]◄**"Der Abakus kam als erstes. "**
41. assign:global_feedback[2]◄**"Viel später folgten die "**
42. assign:global_feedback[3]◄**"mechanischen Rechner."**
43. assign:global_feedback[4]◄**"Danach folgen die Stufen "**
44. assign:global_feedback[5]◄**"wie auf dem Bildschirm gezeigt."**
45. assign:global_feedback[6]◄""
46. assign:global_feedback[7]◄""
47. *
48. content GRAPHICS FILE COMPGR1.CD3
49. * **** GRAPHIC COMPONENT outline number (1.1.1 if not required)
50. assign:graph_component[1]◄**"1.1.3.1"**

```
51. assign:graph_component[2]◄"1.1.3.2"
52. assign:graph_component[3]◄"1.1.3.3"
53. assign:graph_component[4]◄"1.1.3.4"
54. assign:graph_component[5]◄"1.1.3.5"
55. assign:graph_component[6]◄"1.1.3.6"
56. assign:graph_component[7]◄"1.1.1"
57. assign:graph_component[8]◄"1.1.1"
58. assign:graph_component[9]◄"1.1.1"
59. assign:graph_component[10]◄"1.1.3.7"
60. *
61. content *DEFAULT* IT LOGIC TYPE 1 Right Order 6x
```