Conception of an E-Learning CBT-Framework

Prof. Dr. Rainer Kelch
Fakultät für Informatik
Hochschule Augsburg
rainer.kelch@hs-augsburg.de

Abstract

The planned CBT-Framework should facilitate the use of CBT-Programs, which support the solving process for diverse requirements in fact, but follow an analogous methodologically and didactically approach. The framework should not only enable a unified look-and-feel for the user, but should also facilitate creating and embedding of new products for the developer by designing these products according to similar principles.

Currently the framework is realized Flash-based. Old and approved CBT-programs should be embedded after a redesign. In the moment ten different CBT-programs are available, coming from areas related to computer science, like programming, design of combinatorial and sequential circuit, modelling, optimization and arithmetic.

In the following report the planned mode of operation and architecture of the CBT-framework will be explained at a glance. Additionally the realized and planned general features will be illustrated with the help of the web-based SAP UI-technology BSP (Business Server Pages), realized with the “E-Learning-BSP”-product. For teaching BSP-Programming, there is a big support needs for learning the correct data flow programming. A simulation editor enables to get focused on learning the four different data flow techniques.

Keywords: CBT-Programs, E-Learning, SAP-Business Server Pages, BSP, Simulation Editor.
Requirements for CBT-Programs

CBT-programs support the practical and theoretical learning process for degree programmes in computer and engineering sciences. All this programs can be offered always in any number of languages. The core component of these CBTs is a simulated and simplified development environment, called simulation editor, with the purpose of processing sub-steps of an in general multistage (sequential or parallel) solving procedure. Examples for methods to be trained are programming, optimization of circuits and computer arithmetic. The simulation editor delivers for the students useful semantic verifications, so the designed program resp. construction is not only syntactically correct, but works also in that way it is written in the requirements. First prototypes are running and could be performed and tested.

In addition to the visualization of the most important correlations and operating modes in all various types of application programs, models, circuit types, conversion algorithms, etc. the CBTs should enable primary to accelerate the process of learning of the correct programming technique, solving procedure, etc.

Procedure for Building a CBT-Program

The creating process of a CBT-Program could be segmented according to Fig. 1 in the following four steps. For illustrating this we choose programming resp. circuit construction as CBT-examples.
In the first step the aspects are identified, which generally cause problems for beginners during realization of a requirement like e.g. a correct coding of an exercise resp. designing an electrical circuit.

In the second step, we are looking for approaches to describe systematically analytical and constructive the solving process of typical requirements – and to find with this entry a methodical approach for solving these exercise types.

In the third step, either the correct entries of the second step, mixed with empirical collected resp. forecasted frequently wrong solving approaches are proposed as possible solutions, so the CBT-user has to decide for one of these proposed solutions – or he has the possibility to enter free text resp. to select and to configure components coming from a toolbox. These user entries will be checked for correctness by the program. In both cases the user get a feedback in the kind of “right” or “wrong” and additional hints in case of
“wrong”, why it could be wrong, resp. what is necessary for a correct solution. In case of need the correct solving pattern could be required by the system. A limitation of possibilities for free text input resp. for selection from a toolbox could methodologically give sense either by the user handling or by help texts. Mostly it’s a question of a combination of incomplete coding resp. circuit fragments, where to fill the gaps – and to do this exactly by either selection of proposed alternatives or by free text input resp. by free design of a circuit or by combination of both possibilities.

- In the fourth step the concept of the third step will be implemented in form of a software product. This CBT-program substitutes quasi the tutor, we talked about above. With this the CBT could be used independently of time and place.

This is a short description both of the creating process of a CBT-Program and of the purpose of the methodology to be behind it.

**Hint**

The third step has not to be realized compulsive by an E-Learning-Product in a fourth step. A performance with a human tutor is an alternative, too: a collegiate tutor presents the exercise and an incomplete program coding resp. an incomplete circuit draft, where the students try to fill the gaps and where they discuss their ideas with the tutor. After this the various approaches will be tested in the real development environment: looking for runtime resp. logical errors and for correct realization of the exercise tasks.
CBTs for Application Programming

The requirements, currently realized by prototypes, try to catch the most important problems, indicated by examination results, experiences coming from practical exercises and from teaching for experienced programmer in the industry. You can see the general procedure for identifying the features to be trained, in fig. 2. Analogously it is possible to model the methodology to be trained for e.g. design of circuit drafts.

We choose a concrete special UI-Technology as our starting point for this section. This is only to give a concrete example. It can be also a concrete special optimization algorithm for sequential circuits, etc.. For this, it exist generally programming techniques, which can be subdivided in sub-techniques. This process of subdividing may contain an arbitrarily number of steps, which is for reasons of overview not be considered in figure 2. In this tree, the leaves are representing the real concrete programming statements.

In the next step, the concrete features and requirements are listed in a structured way. Starting from this list, a catalogue of requirements will be build, from which a multiple parameterized (degree of freedom for input data, level, total amount, data for exercises) exercise formulation will be created.
BSP-Data Flow

As a concrete example, realized in an executable prototype, we look on the web-based dialogue programming with BSP. For dialogue programming we refer to the 4-layer-model of SAP-Dialogue Applications (see [Kelch2008], [Kelch2010a], [Kelch2010b]). Fig. 3 illustrates this dialogue application architecture for the UI-technology of Business Server Pages (BSP).

The main problem in this case is the realization of the vertical and horizontal data flow, then the handling of user-events. Because there exist in the moment according to a structuring more in an overview three different UI-technologies, which are relevant for the practice in the SAP-Environment, we should handle all three in principle.
But the most problems appear concerning BSP Applications (Business Server Pages), because the SAP-Framework will not regularize so much as for Dynpro-based SAP Transactions (running in the SAP-specific SAPGUI-Frontend) or for the strongly framework-supported Webdynpro programming model, which is similar to the BSP-Model web-based.

From the manifold possibilities of realizing the data flow, we limit us on the auto-, cookie, class- and MVC-technique. Also we have to consider the vertical data flow separately, and all in dependence of the state model (stateless resp. stateful).

Because of the well working data flow inside a page and also between two pages, the following aspects are to be considered:

- Selection of the state model
- Selection of the data flow technique
- Decision for native-HTML resp. BSP-Extensions

The concrete kind of realisation of the event handling process depends on, if we work with BSP-extensions or not. But it has only an indirect influence on the data transport.
What shall be considered for a correct working data flow?

- In general you have always to proof, if the names of the UI-elements, the names of the page attributes inside the first page and also the names inside the second page are exactly the same (Equality of the names).

- Those UI-elements, where the values are to transfer, have to be placed inside of a form. If several forms are used, they have to be places inside the forms with the same name.

- Additionally, the conditions, specific for the chosen technique, have to be fulfilled.
E.g. fig. 4 illustrates the programming logic for the data flow using the cookie-technique.

**Data Transport – Cookie Method**

![Diagram of Data Flow Logic using Cookie Technique]

Now you can get an idea, why there are for the beginner because of these requirements many possibilities of writing wrong code – resp. the bug fixing is very difficult. Therefore here is a “best practice”-check list very helpful and the CBTs will proof exactly the conditions, listed here, to get a well working data flow.

In fig. 5 the requirements for an E-Learning-CBT-Program are formulated, which are necessary to learn the methods of data transport techniques for BSP-applications.

Starting from this point, it is possible to design a pattern-based creating of a CBT-kernel, i.e. a parameterized exercise template.
In the past there were several project teams, in which the ideas of the author would be partially realized. For example, the project “Visualization of the data flow for BSP-Dialogue Applications”, realized in 2006, would be rewritten by the project team “E-Learning BSP” in 2009 like the following: it got a completely new design, the content would set on an actual state, got an extended functionality and got more flexible (see [BSP2010]). Later on, extensions are developed and actually this work is in progress.

For this prototype, we got first very positive evaluations related to use by students. The focus of this program is the realization of the data flow for BSP-Applications. For the visualization of the data flow, they developed animations.

CBT-Program "E-Learning-BSP"

Fig. 5: CBT-Requirements for E-Learning-BSP
Fig. 6 shows the kernel of the BSP-E-Learning-Program: the simulation mode. The user can fill the gaps inside the coding, useful messages are the response by the program. Here not only syntax errors are displayed, but also semantic errors, which would not result in a runtime- or compiletime-error, but would result in an incorrect working data transport.

The colouring shows, which input is correct (green) and which is incorrect (red). You find the error messages in the feedback area. They allow the user not only to locate the errors, but they give hints for the correct solution, too, without only giving the right answer. Keep in mind by displaying all errors, that they are displayed for all tabs and panes together. The areas marked in fig. 6 show e.g. that for two pages and their in four areas (Layout, event handler for onInputProcessing and onInitialization, page attributes), i.e. eight different areas totally, you have to handle data input and checking the data input.
Fig. 7 shows a snapshot of the related animation mode. This mode illustrates, when which data in which components of the BSP-application were transported.

You can influence the velocity of the animation by using a slider, stop and restart the animation, or demonstrate the animation step by step. Parallel to this animation the currently active components are marked in a colour, so you can really follow, what will happen at which button press and what will be necessary for a correct data flow.

**Evaluation**

There was coming a very positive feedback from the project “E-Learning BSP”, especially from the multi-lingual version (at this moment in the past: German + English). In the meanwhile exist also a Portuguese version, some other are planned.

In an international Master course, one team of students was able to realize their specific requirement in their examination task with MVC-BSP – although this topic
was not discussed in the classes. The students learnt and trained this data flow technique only by using this CBT-program!

A large sized evaluation is planned. Because of the great variety of the curricula of the different study courses at different universities, this is a complex task, and additionally very difficult for finding measure. Groups to be comparable can be built, but should only be built as an optional offer inside the area of teaching classes for reason of fairness. Conceivable is to coach three alternative groups in a different manner: one with a tutor, one only with the CBT-program, the third with both. The learning effort should not be measured in form of a mark, but in time which is necessary to complete one examination task independently and correctly.

**Conclusion and Perspective for CBT-Programs**

In general the CBT-programs got a very positive evaluation. The CBT-Program for BSP-data flow will be extended with components for the BSP-Extension-based event-handling. Furthermore, a CBT-program for the data flow for SAP-Transactions (classic and SAPGUI-Control) and for Webdynpro-applications will be build. For both UI-technologies modules for training the event-handling will be offered, as realized for BSP.

As soon as these components are complete, they will be integrated in a "ABAP-Learning-CBT"-program. All programs and the framework are offered with the minimum of two languages. The already existing versions in three languages serve as pattern. Furthermore planned is the embedding of a mini-SAP-System. So the filled gaps can be tested live.

The other of about a dozen already realized CBT-programs, coming from the area of computer-hardware-components /-design, will be redesigned to be embedded in a Flash-based Environment, too.
CBT-Framework

Topic is a framework for CBT-programs (Computer Based Training), which should embed the concrete CBT-program (see fig. 8). In this case the training of methods with concrete exercises is in the foreground. Other features, well-known from E-Learning like a questionnaire for learning facts, learning videos, animations of technical processes, online-help and documentation build a complementary frame and are offered additionally.

The main idea, on which these training programs are based, is a comparison from user input values with one resp. several proofed solutions, stored in the system resp. if they are parameterized or the solution could be generated with a formal / generic approach, will be interpreted during runtime with the actual parameters.

One important idea for constructing a CBT-framework is to generalize this comparing philosophy of the single CBTs for the framework, so that a central comparison tool makes possible all embedded CBTs to execute correct program-specific proofs of correctness.

You find possible and useful CBT-programs in fig. 8, which fit in the CBT-framework to be realized. Under hierarchical perspective the lowest leaves in this tree represent the already realized CBTs. The leaves marked with “…” demonstrate that here may be embedded an arbitrarily number of other possible components.
This verification method will not differ in the very first moment from a classic multiple-choice questionnaire for fact knowledge. The decisive difference is, that the exercise formulation and also the solution for methods knowledge is distinct more complex. According to the exercise a different number of sub-steps is necessary. Partially these sub-steps can be solved independently from each other, but often they have to be processed sequentially, so that a useful processing of the next step is possible only, if the previous step would be verified correctly solved.

Nevertheless, to apply the above formulated main idea is possible, even if to be well thought-out in detail it is very complex. To validate the learning success, a very sophisticated verification- and help-system is necessary, so the user is able to get a correct solution by not only using the try and error method.
If you try to map that method in a software product, what a human tutor is able to give hints and what he needs as feedback from the exercising people, then you have grasped the main principles of these CBT-programs. One important detail for implementing this idea is, to permit only user inputs for the essential problems – but to propose these parts, where experience has shown that only few problems will appear within a correct solution. According to the level of difficulty the input resp. the select options for a correct solution can be designed different.

**Architecture**

A framework should make possible to offer very different CBT-programs with this learning philosophy embedded in the same software frame. It should be possible, to embed very different exercise types coming from very different sciences in the same framework. The exercise formulation will be embedded in form of a XML-file. One of the essential problems yet to be solved in this wide research project is to embed the method to be trained without the necessity of writing the complete program from the scratch.

From the user perspective the CBT-framework offers the advantage that the look & feel is well known and similar for all types of exercises, and it is possible to add new exercises at any time, without the necessity of reinstalling the product once more.

From the developer- resp. producer perspective, the framework delivers the advantage to be able to realize permanent new requirements, without the need of adapting the principal program architecture permanently. For methodologies, which are differing very strong because of the exercise formulation, different fitting program templates will be offered, which can be built in the framework modularly.

The CBT-framework contains the modular designed sub-programs for the according application areas. In fig. 8, an useful example architecture was presented already.
UI-Structure

From the user perspective the offered functionalities are in the foreground, which are ordered, due to the framework for all CBT-programs similarly.

Fig. 9 shows stylized the UI-structure of the initial screen resp. the view of the simulation editor form the perspective of structuring in components.

Fig. 10 shows stylized the general UI-structure using the active simulation editor in the perspective of the UI-ordering.

Fig. 9: EasyCBT-UI-Components
**Designtime**

For designing the concrete exercises so, that they can be used by a framework conform CBT-program, the developer of the exercises to be trained are orienting themselves at the algorithm modelled in fig. 11.

First a solution related to the exercise will be created. Then an exercise pattern will be created in a manner, that the relevant parts are identified and marked as later gaps. Depending on the difficult level solving possibilities related to these gaps are offered resp. free text input (with/out hints for possible input patterns) is allowed. According to this error messages will be defined, which help the user in the case of error to find without help from outside the correct solution.
Fig. 11: EasyExercise-Design

_Runtime_

The user runs through the activity diagram, displayed in fig. 12. This workflow should be self-explaining as possible. This can be realized by an ergonomic user interface and useful help capabilities.

After the user was selecting an exercise and filling the gaps, the tool checks by comparison of input values and stored required values, if the user input is correct. If yes, a success message will be sent and working on the exercise is finished. If no, the detailed error messages will be displayed, e.g. also the number of errors. The error messages should be so detailed, that they give a useful hint for the user, without giving directly the correct solution.

Now the user can correct his input data, execute once more the check of his input values, and if necessary take once more the actual error list to correct these errors in the next run. After a fixed number of attempts (e.g. three), the tool offers beside the list of errors, to show the correct solution, if the user wants this now. Here it should be possible to realize in detail, if resp. which of the already existing errors should be
automatically corrected and for which of the errors the user likes to start a correcting process by himself.

![Diagram of EasyExercise-Run](image)

**Fig. 12: EasyExercise-Run.**

### Comparing Algorithm

We use the comparing algorithm, displayed in fig. 13. After loading of the chosen exercise, parallel will be done the following: on one side the user solution will be loaded, the user input data will be stored, on the other side the system solution(s) will be read and the contents for the gaps to be filled, are temporarily stored.

For each gap the according entries of both lists will be compared. From this a list with “yes-no”-Entries will be generated. If all entries contain “yes”, a success message will be displayed. In the other case only the “no”-entries will be considered and the related error texts will be displayed.
Conclusion and Planning of the CBT-Framework

The construction of the CBT-Framework is very complex in detail, in spite of the simple main idea. It will require a lot of work, but this will be worthwhile: as soon as it will be available, totally different methods can be used with the same CBT-Portal. For the developer as well as for the user, the effort for extensions for additional exercises, languages and exercise types will reduce to a minimum.

Fig. 13: EasyCompare-Algorithm.
Copyright

The following expressions are under Copyright of the author Rainer Kelch: EasyCBT-Creating, EasyPPA / Easy Programming Pattern Approach, Easy4Level-Architecture, Easy-Dataflowlogic, E-Learning-BSP, EasySimulation, EasyAnimation, EasyCBT, EasyExercise, EasyCompare. All diagrams, figures and screenshots are under Copyright of the author, too.

Literature


[BSP2010] Deisenhofer, Chr., Hörmann, M., etc.: E-Learning-BSP, project work under guidance of Prof. Dr. R. Kelch 2009, extended development 2010


About the Author

Rainer Kelch
Diplom-Mathematiker, University of Karlsruhe, 1986; Doctor of Science, 1989; SAP-certified Solution Architect for SAP ERP in 2009 and erpSIM in 2010. Professor at University of Applied Sciences, Augsburg, since 2000. Working in the Faculty of Compute Science, focus on Wirtschaftsinformatik, subject area Commercial Applications. Since 2001, responsible for foreign affairs of the Faculty of Computer Science. Guest Professor at the HSW (University of business Administration) Lucerne, Switzerland, for the whole Summer semester 2004 in the study cours business Administration and Wirtschaftsinformatik. Since 2007 Guest Lecturer at the University of Liechtenstein, Vaduz in the Institute of Wirtschaftsinformatik for the Course Objektorientierter Softwareentwickler, in the Master of ScienceCourse in Business Process Engineering (English-Lessons) for the courses Object Oriented Software Development and Business Process Implementation.