[ESFiSt99] *Abdulmotaleb El-Saddik, Stephan Fischer, Ralf Steinmetz;* **Component-based framework for effictive visualization of educational algorithms;** H.-J. Bullinger, P.H. Vossen (Eds.): Adjunct Conference Proceedings of HCI International 99, Stuttgart Fraunhofer IRB Verlag, August 1999.

In: H.-J. Bullinger, P. H. Vossen (Eds.): Adjunct Confernce Proceedings of HCI International '99. Stuttgart:Fraunhofer IRB Verlag, 1999.

# Component-based framework for effective visualization of educational algorithms

**Abdulmotaleb El Saddik[1], Stephan Fischer[1], Ralf Steinmetz[1,2]**

1
Industrial Process and System Communications
Dept. of Electrical Eng. & Information Technology
Darmstadt University of Technology
Merckstr. 25 • D-64283 Darmstadt • Germany

2
GMD IPSI
German National Research Center
for Information Technology
Dolivostr. 15 • D-64293 • Darmstadt • Germany

{abed, Stephan.Fischer, Ralf.Steinmetz }@kom.tu-darmstadt.de

## Introduction

A vast amount of animations has been generated in the last few years, caused by the rapid growth of the WWW in combination with languages like Java and VRML. However, these animations often have some major drawbacks, such as:

- *Video-like nature of animations.* Most of the animations being used to visualize complex algorithms and techniques cannot be influenced by the user. S/he can only watch the ongoing animation and try to understand the underlying theory. As the only available form of interaction most animations use parameters to change the output.
- *Experiments.* The user cannot change the behavior of applets by omitting certain steps or by adding or exchanging components.
- *Connection of animations.* Most applets available nowadays are running in a stand-alone mode. The user, e.g. cannot connect an animation of a video decoder to that of a network and study the resulting effects.
- *Reusability.* Many animations have been developed without regard to software engineering in terms of reusability. The animation of JPEG [4] and MPEG [2] serves as a good example: Even though both compression schemes use the Discrete Cosine Transform (DCT), and the Huffman encoding, a reuse of a component coming from an already finished animation of JPEG can in most cases not be used to visualize a step of the MPEG-compression process.
- *Hierarchical structure.* Most applets do not deal with changing user requirements. Beginners, as well as intermediate students and experts become nowadays the same animation.

## Offering support through interaction

Before presenting the architecture of the ItBeanKit the requirements for such a toolkit must be identified: namely *interaction* and *support* of the learner.

*Interaction* implies that the learner is guided in the sense that he can get feedback if problems emerge. Assuming that the handling of the toolkit itself is intuitive such problems can only result from the difficulty of the topics to be learned. The *difficulty* of an algorithm to be animated can result either from the knowledge of the learner which might not be sufficient to understand the topic or from the amount of information presented by the animation. If the user's knowledge is not sufficient to understand parts of an algorithm we offer two possibilities to create the corresponding knowledge: a user can read a short explanation of the part of the algorithm he currently executes or he can invoke the chapter of the textbook explaining the underlying theory in depth. The latter includes search functions to get a more specific way of explanation.

The processing of an insufficient knowledge of a learner is performed in a traditional way by using hyperlinked multimedia documents. The second problem however, the density of the presented information has to be dealt with by another approach, the use of *levels of complexity*. The idea behind a level of complexity is that a user can reduce the information density of a part of an algorithm by splitting the part of an animation he/she is currently using into a particular number of steps which can be understood easier equivalent to a

smaller information density. This process is shown in Figure 1. While $C$ stands for complexity, the upper index denotes the level, the lower the number of a component.

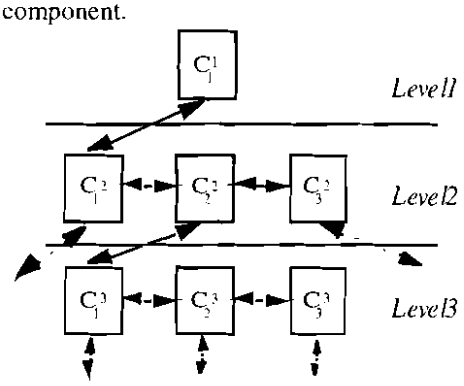


**Figure1: Levels of complexity**

We distinguish between two kinds of interactions the user is provided with: content dependent interactions and content independent interactions.

## Content-Dependent Interaction Model

These interactions are strongly bounded with the topic to be visualized:

- Variation of parameters of a running (interactive teaching Bean) itBean
  - Visualization (level of complexity)
  - Animation (speed, background color, foreground color,..)
  - Simulation (interactions by the user interface)

## Content Independent Interaction Model

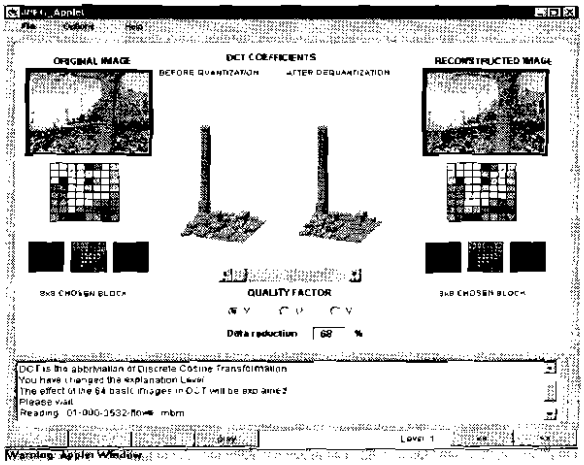


**Figure2: User interface of component beans**

This model represents those interactions, almost all developed applications will have in common:

- Guiding the user
  - Help function
  - Guided tour
  - Step function
- Language (at the moment we are supporting: English, German, Spanish)
- Explanation (Errors, Hints, Audio)
- Look And Feel (Java, Windows, Motif)

## Conclusion and outlook

In this paper we described a component-based architecture for animations using JavaBeans. Our approach extends other concepts by the use of hierarchies thus supporting the learner in an efficient way. To achieve a common look-and-feel we separate the graphical output from the itBean itself.

At the moment we study how the itBeankit can be integrated in a test environment. Instead of using multiple choice students can experiment with components. If these are given the student has to prove his knowledge by finding the right order in which the components have to be placed. It is thus immediately possible to observe if the answer is correct or not because the result is presented graphically. If a learner places the entropy encoding of the JPEG-compression in front of the DCT the result will change significantly.

## References

1. Land, B. R.: *Teaching computer graphics and scientific visualization using the dataflow, block diagram language Data Explorer.* In proceedings of the IFIP WG 3.2 Working Conference on Visualization in Scientific Computing, Uses in University Education, Irvine, USA, pp.33-36, 1994.
2. Joan L.Mitchell, William B. Pennebaker, Chad E. Fogg, and Didier J. LeGall: *MPEG video compression standard.* ISBN 0-412-08771-5, Chapman &Hall 1997.
3. Wernert, E.: A unified environment for presenting, developing and analyzing graphics algorithms. Computer Graphics, 31(3), pp. 26-28, 1997
4. Steinmetz Ralf, and Nahrstedt Klara: *Multimedia computing, communications, and applications.* ISBN 0-13-324435-0, Prentice Hall 1995.