Evaluating Software Engineering Modeling Concepts in the Context of Narrative Systems Design

Mikael B. Skov* and Lars Bo Eriksen#
Department of Computer Science
Aalborg University
Niels Jernes Vej 14
9220 Aalborg Øst
Denmark

+45 9635 8080
{dubois, lbe}@cs.auc.dk

Abstract:

Research has shown that the design and creation of interactive multimedia systems is a difficult and challenging task. Most interactive multimedia systems differ from conventional computer systems by being functionally more complex and by addressing a wider and broader range of inexperienced or novice users. The increased complexity in functionality is often a result of integration, time variation, and navigation. Furthermore, years of research within software engineering shows that improvements in software design processes require systematic work practices that involve well-founded methods.

This paper reports from an empirical study of a conventional software analysis and design for the design of an interactive multimedia system. The primary purpose of the study is to investigate software engineering concepts for the design of interactive multimedia systems. The actual activities and concepts of the method are empirically evaluated against a typical desktop-based interactive multimedia system and weakness and strength of the analysis and design are discovered. It is concluded that more software engineering concepts are not well suited for interactive multimedia systems design. The main reason is that more of the concepts are intended for the analysis and design of work processes in a given, specific work environment. However, the rigid focus on work processes and tasks is useless since the complexity of the future system is not hidden in work tasks. The challenge of designing a coherent and affective interactive multimedia system lacks conceptual support and foundation. It is indicated that future analysis and design methods need to address other aspects of systems design than previous methods.

1. INTRODUCTION

The design and creation of multimedia systems is difficult and challenging (Grosky 1994). During the last couple of years with the evolvement of hardware technologies, numerous multimedia applications have been focus for industrial development and research interest, cf. (Druin and Solomon 1996, Plowman and Luckin 1999, Webb 1996). The diffusion of multimedia technologies has been so intensive that multimedia systems have emerged and penetrated almost every type of environmental settings. These applications include of wide range of computer-based systems ranging from interactive storytelling for children at a hospital unit, cf. (Umaschi Bers et.al. 1998), to educational titles for work professionals, cf. (Lewis et.al. 1998).

Webb (1996) points out that most multimedia systems are functionally more complex than conventional computer systems and are often used by a wider range of inexperienced or novice users. The increased complexity in functionality is often a result of integration, time variation, and navigation. The broad variety of multimedia application makes it difficult to identify and delimit the class of applications. In fact, Eriksen, Skov and Stage (2000) state that it is difficult to identify a unified and accepted definition on the term multimedia system. McKerlie and Preece (1993) put up a broad definition with an asset perspective saying that a multimedia system is an *umbrella term for the integration of different elements, such as text, graphics, video, still photographs and sound, in a single application*. Nemetz and Johnson (1998) expand by an interaction perspective stating that *multimedia systems encompass both input and output media and focus on human-computer interaction rather than on the technological aspects*. Skov and Stage (2000) defines a multimedia system as *a computer-based system that integrates a multitude of assets to facilitate user immersion and activity in a virtual situation. The assets are representing fragments of the virtual situation and are based on modalities such as text, graphics, pictures, video, animations, sound, tactile information, and motion.* They continue by saying that multimedia systems *involve interaction with objects in the virtual situation and it is limited by certain temporal and spatial structures. A plot defines the development of the virtual situation*. Webb (1996) argues that multimedia systems are more mediums or theatres than tools bringing new kinds of complexity into the design of these systems. Multimedia systems also often incorporate a higher degree human-computer interaction than more conventional systems.

The design of multimedia systems is more difficult and less understood than conventional software design, cf. (Grosky 1994, Sutcliffe and Faraday 1994). During the development of conventional software, designers are primarily concerned with identification and description of work processes and work tasks, cf. (Booch, 1994; Rumbaugh, 1991; Jacobson, 1992; Mathiassen, Munk-Madsen, Nielsen, and Stage, 1999). However, Webb (1996) stresses that the business metaphor does not apply well for multimedia systems development. Furthermore, Sutcliffe and Faraday (1997) state that research within multimedia systems development shows that contemporary

multimedia systems are designed and created primarily by intuition and hence lacking method support and systematic approaches to work practices. Years of research within software engineering shows that improvements in software design processes require systematic work practices that involve well-founded methods, cf. (Fairley, 1985), (Pressman, 1996), (Sommerville, 1992). Furthermore, Eriksen, Skov, and Stage (2000) recognize that no methodological approach to multimedia systems development that embodies an established tradition exists.

This paper investigates the usefulness of software engineering concepts for the design of a typical multimedia system. This is done by discussing the key concepts and activities of a systems development method in context of a specific multimedia system. The paper is organised as follows: Chapter 2 presents the background of systems development methods and concepts, and the example of a multimedia system, which will be used for the discussion. Chapter 3 illustrates the applicability of these concepts for multimedia systems design. Finally, chapter 4 and 5 discusses the results and concludes the work.

2. BACKGROUND

We will investigate the applicability or usefulness of an established systems development method for the design of multimedia products. First, we will introduce focus and concepts of systems development methods and show how development methods have evolved over time. Second, we will describe a real multimedia system, which will be used for the discussion. Finally, we will present the chosen method for this discussion.

2.1 Systems Development and Methods

Mathiassen, Munk-Madsen, Nielsen, and Stage (1999) state that system development involves the adaptation of computerised systems to peoples needs. That is, during the development of computer systems, you are focusing on making the computer system useful and useable for people. Long tradition has formed the system development process to consist of three activities, analysis, design, and programming, where programming is the key activity with its focus on the actual construction (Booch, 1994; Rumbaugh, 1991; Jacobson, 1992; Mathiassen, Munk-Madsen, Nielsen, and Stage, 1999). Prior to programming the system needs to be understood and described. The goal with the analysis and design activities of systems development is to create an overview of requirements to the system and establish a basis for implementing the system, cf. Mathiassen, Munk-Madsen, Nielsen, and Stage (1999). Jacobson (1992) states that since systems development is a relative young industry, the development of software systems has not reached the level of maturity typically found in more traditional branches of industry. As a consequence, developed systems often suffer from a lack of the established practices required for their development and

exploitation as commercial products Jacobson (1992).

System development methods are an attempt to systematize and mature the development process by establishing standard notations, guidelines, and processes for the development. Booch (1994) defines a system development method as a desciplined process for generating a set of models that describes various aspects of a software system under development using some well-defined notation. Sommerville (1992) continues by arguing that most system development methods can be characterized as either top-down structured design, data-driven design, or object-oriented design. Top-down structured design methods, e.g. (Yourdon and Constantine, 1979; Myers, 1978), are characterized by algorithmic decomposition of the problem. Top-down structured design methods have shown their usability for many years and are probably more used than any other methods Booch (1994), but do not address issues of data abstraction or information hiding and have problems when modeling extreme complex systems. Data-driven design methods, e.g. (Jackson, 1975; Orr, 1971), are characterized by the direct mapping of system inputs and outputs and have been successfully applied in modeling complex domains like information management systems. Object-oriented design methods, e.g. (Booch, 1994; Rumbaugh, 1991; Jacobson, 1992), rely on that one should model software systems as collections of cooperating objects, treating individual objects as instances of a class within a hierarchy of classes, cf. Booch (1994). During the 1990's object-oriented design methods have become state-of-the-art both within research and industry, cf. Mathiassen, Munk-Madsen, Nielsen, and Stage (1999) and with the invention of the Unified Modeling Language (UML), this position has been even more strength. (mangler reference)

Even though system development methods have evolved in response to the growing complexity of software systems, cf. Booch (1994), the methods are direct symbols of their time. Monarchi and Puhr (1992) argue that most object-oriented analysis methods emphasize modeling the problem domain. The methods are concerned with abstracting relevant objects from the problem domain, defining the structure and behavior of the objects, and determining the interrelationships of the objects. The goal is to model the semantics of the problem in terms of distinct but related objects. Booch (1994) extends by saying that object-oriented analysis is a method of analysis that examines requirements from the perspective of the classes and objects found in the vocabulary of the problem domain. Hence, the focus is on analyzing and describing the problem domain and on work tasks of future users of the system. Züllighoven (1998) even stresses that the systems of focus are intended to support the work of domain experts of a given problem domain. Typical examples would be administrative systems, e.g. a bank account system, or technical, embedded systems, e.g. a cruise control system for a car. With the introduction of new kinds of computer systems, e.g. multimedia systems, systems development methods face new and more challenges.

2.2 The System Development Method Case: *OOA&D*

The chosen development method for this investigation is a Scandinavian systems

development method and is based on object-oriented description tools. It is called Object-Oriented Analysis and Design (*OOA&D*), cf. Mathiassen; Munk-Madsen, Nielsen, and Stage (1997, 1999, 2000). The motivation for choosing this particular method is straightforward, it combines strengths from a number of well-known system development methods (Booch, 1994; Jackson, 1983; Jacobson, 1992; Rumbaugh, 1991). In addition, it integrates the Unified Modeling Language (UML) as notation.

We have chosen to only include the analysis part in this part. The analysis part consists of three major activities, namely System Choice, Problem Domain Analysis, and Application Domain Analysis. First in the System Choice activity, one has to establish a foundation for the remaining analysis be selecting and describing the overall structure of the future computerized system. Second, in the problem domain analysis, the computerized system must be understood from an information perspective. It contains a model of the relevant elements in the problem domain that enables the users to administrate, monitor, or control phenomena in the problem domain. Third, in the application domain analysis, the computerized system should be understood from the user's point of view. One should understand which people, apparatuses, and other computerized systems it should interact with and also which functions to offer these actors Mathiassen, Munk-Madsen, Nielsen, and Stage (1999, 2000).

2.3 The Multimedia Case: *Virtual Management®*

Virtual Management® is a multimedia product developed by a Danish software firm InterAct. *Virtual Management®* can be viewed as two different systems. First, it is a tool for recruitment consultants during the process of evaluating and assessing job candidates for various work positions. Second and for this paper more important, it is a narrative multimedia system placing the job candidates in difficult work situations in a virtual environment. The basic idea is to simulate typical work situations and compel the job candidate to make decisions and manage in these situations. Figure 1 shows a typical situation from *Virtual Management®* where a video sequence from a staff meeting is played.

Figure 1: A scene from *Virtual Management®,* where a video footage from a staff meeting is played.

The job candidate plays a virtual figure in the story and is going to attend a staff meeting as the new manager of the department. The job candidate is to assess the situation of the staff meeting and later make a decision upon the situation. The candidate can select from different predefined decisions, leave the meeting, interrupt people talking etc. Often these video sequences depicture conflicts in the organisation forcing the candidate to make both unpleasant and tough decisions. All actions of the job candidate are recorded for later analysis and assessment.

Further aspects of the *Virtual Management®* will be discussed during the evaluation in the following sections. We will not discuss concepts and activities from the actual design process of *Virtual Management®* in this paper, the system will merely serve as concrete example of a typical multimedia system. Discussion of the design process can be found in (Eriksen, Skov, and Stage, 2000).

3. MULTIMEDIA SYSTEMS ANALYSIS

Having established the foundation for this discussion, we will in this chapter discuss key concepts and aspects of the three activities of the analysis part.

3.1 System Choice: Two Levels of Situations

A computerized system in OOA&D is put into use in a specific use context called the application domain. The purpose of this system is to administrate, control, or monitor a problem domain. Users interact with the system, either to feed data into the system about changes in the problem domain, or to get information from the system about the problem domain. These actions make sense in a broader organizational view of a situation. E.g. a librarian that uses an information system to register that a book has been loaned by a particular user of the library, or the same librarian uses the information system to check whether a book is available for loan or whether it has been reserved etc. The librarian being in the application domain, administrating the problem domain of books and book loaners. The problem domain and the application domain are part of the same *situation* in which objects, users, structures and processes are causally coupled. The purpose of applying the method is to understand which parts of the situation belongs to the problem domain, which parts belong to the application domain, and which parts of the complex reality does not belong in the system development project at all.

Let us take a look at the *Virtual Management®* case. The situation for this system would be the environment in which the job candidate takes the test or uses the system. This environment could be any kind of room or office where the job candidate can interact with the system without being disrupted. In fact, the environment could be any

kind of room, the job candidate could in fact sit at home doing the test. Therefore, analysis and definition of this situation does not necessarily bring any important information to the development on how the system should be used. On the other hand, one could argue that another kind of situation exists namely the situation, which is depictured by video footage in the system. As illustrated in figure 2 here the situation is a staff meeting where the job candidate as the new manager of the department are confronted with a virtual situation. The video footage depicts a situation where the manager is having a conversation with the employees. However, this is not the situation presented in OOA&D, and one could argue that we need to analyse the situation at different levels. Though multimedia systems often require analysis of situations at two levels: The situation which is to be depicted by the multimedia system (we name it System Content Situation), and the situation in which the multimedia system is to be used (we name it System Use Situation). Whereas administrative, control, or monitor systems have a tight coupling between problem and application domain, multimedia systems may have very little. Using the two types of situations mentioned above, the System Content Situation describes the simulated management environment in which the job candidate reacts, and the System Use Situation describes the environment, e.g. an office, in which the job candidate takes the test.

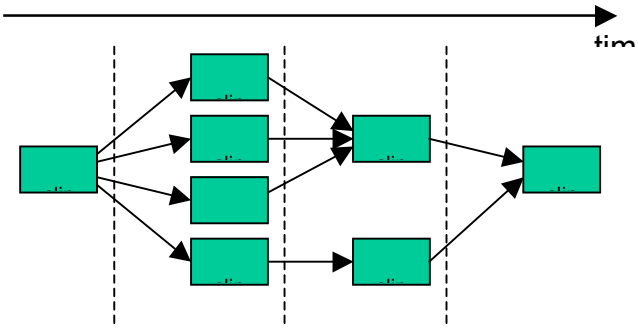
Figure 2: A scene from *Virtual Management®,* employees from the department outline the current problems in the department to the new manager (the user).

The consequence of this no single situation is, that we cannot simply look at the future use context to get information about what is to be in the system. We may identify needs or problems in an application context, but these will not necessarily help us in understanding the situation to be depicted by the system. Thereby the principle: "*Use the situation of the users as a basis, but be critical towards the task description"* does not have the same impact as it has when problem domains and application domains are tightly coupled. Whereas an administrative system development project will gain much information by looking into the everyday of future users to learn about

processes, structures and objects being managed, the situation of a multimedia system user will not inform system developers as to what the system is to contain. This is because the system situation is not part of the user situation. We may identify a need of some sort by looking at the user's situation. But when it comes to realizing a multimedia system that fulfills this need, we are left almost blank on whether we are to shoot 35 hours of movie, segment it into a hyper-structure, replicating a multiple choice test, or whether we should produce an interactive book, or something completely different.

3.2 Problem Domain Analysis: No one-to-one relation between environment and system

Having separated the situation description into two situations for multimedia systems, the next question arises whether the concepts of problem domain and application domain still are useful for describing multimedia systems in greater detail. At a first glance the definition of problem domain poses problems by the terms of which it is constructed.



Multimedia systems are typically not used to *administrate, monitor or control* something. Multimedia systems often convey a story (or plot) and serve as the basis for an experience of some duration. To make the definition more adequate in terms one could extend it to: "*[...]* administrated, monitored, controlled *or simulated* by the means of a computer-based system". This extension would allow for a broader range of system, possibly including multimedia systems, to fit into the scope of OOA&D., but at a more fundamental level the definition still is troublesome in its reliance on an environment. Our argument is, that there is no environment of relevance, at least not in the terms of OOA&D. The environment referred to in the definition of problem domain must necessarily be the environment of the user. But as argued in the previous section this environment is part of system use situation, and not necessarily system content situation. And the study of the environment of the user does not inform us on how to structure the system.

OOA&D assumes that there will be a 1-1 correspondence between the users environment and a model of that environment within the system. The purposes of the users interaction with the system being to either update the model in the system, or to extract data from the system to get information on the real world objects. Since this is not the case for a multimedia system in which a virtual environment is created, the definition of problem domain leaves the developers using OOA&D blank on how to make sense of the problem domain definition in multimedia system development. The argument on the lacking 1-1 correspondence between user environment and system

model also indicates problems with the definition of object system. "*A future users understanding of a problem domain".* The definition of object system is reliable on the definition of problem domain, and as we have argues above, the problem domain definition is not valid for many multimedia systems, hence the object system definition is probably not valid either.

Virtual Management® conveys around a temporal plot or structure. The data in the system is a representation of the segments that make up the story, with functionality that will alter the flow of the story in response to the users interaction. Figure 3 illustrates a general representation of the temporal structure in *Virtual Management*®, where the boxes represent video clips depicturing different work situations, and the arrow-lines represent different choices or selections made available to the user. Each selection made by the user poses a certain video clip to be played thereby making up the entire plot of the system.

Figure 3: Illustration of the temporal structure in *Virtual Management*®

Let us return the situation illustrated in figure 2 as illustration of this structure. Here the user is the manager of a department taking part in a staff meeting. Suddenly the phone rings, and the video sequence is stopped. The user is then forced to select one of normally four predefined actions, eg. answer the phone, continue talking with staff. Based on the choice of the user the story continues and this pattern continuous through the entire use of the application. The actions of the user in a temporal plot only moves in one direction: Forwards. There may be some diversions on the way, but the nature of a temporal plot is that the experience will progress towards an ending. *Virtual Management*® has actually integrated an explicit representation of the temporal aspect. A clock is shown at all time in top-right corner on each screen picture indicating the virtual time of the day to the user. From figure 1 to figure 2 you will notice that the time has changed from 9.05 to 9.10.

Whereas the *Virtual Management*® evolves around a temporal structure, other kinds of multimedia systems are built upon spatial structures, eg. various arcade games, where the user typically orientate and navigate in 3D virtual environment. Take for instance the famous action game Quake developed by id Software. Here the user is a virtual figure in a virtual spatial world where the goal is to eliminate other virtual figures.

The two types of system realized from relying on either *temporal or spatial* organization of the plot, rely on very different models. Whereas the spatial plot is represented in a spatial structure in which the user orientates and navigates, the temporal plot is represented in a temporal structure indicating flow of progression. This distinction we believe to be fundamental, in that it is informing at a basic level on how the content of the system will be structured. Both the definitions of object system and model are highly reliable on the definition of the problem domain. Having argued that the definition of problem domain is not applicable when it comes to developing multimedia systems, the consequence is that one of the basic building blocks of OOA&D, the model, in its definition becomes blurred. The consequence is the same

as for the system choice activity, the developers are left with a weaker apparatus for defining the content of the system.

3.3 Application Domain Analysis: Lack of Focus on Interaction

User actions are related to either feeding information to the system for the purpose of maintaining the model, or to extract information from the system with the purpose of getting information on the real world objects. Thereby using the system to automate trivial tasks. The application domain may or may not be trivial identified for future computer-based multimedia systems. Some multimedia systems are intended for training purposes for specific work tasks. As an example take a training multimedia system for engineers working at a nuclear power plant where the objective is to train engineers in operating a specific part of the main control system. Here you are able to describe the application domain and furthermore identify work tasks of these engineers. However, another type of multimedia training systems, namely *Virtual Management*®, do not share the same characteristics. Here you cannot identify the same type of user and the environment may vary from time to time.

Further, OOA&D suggests that aspects of using the system should form the basis when describing the application domain. The focus is on actors and use patterns. Conceptually we question the firmness of the definition of an actor: *"[A] role including users [...] which has the same use pattern."* We find this definition to be somewhat wrong. An actor fulfills a role, an actor is a instance of a human being, the role is the general pattern. E.g. many different actors has played the role of Hamlet. OOA&D mixes up the concepts of actor and role into one definition. Secondly we claim, that the concept of *use pattern* is not applicable to multimedia system use. Lets us take a look at the distinction between spatial and temporal structured multimedia systems. First, in spatial structured multimedia systems, the actions of the user are of the type orientation, navigation, or general actions. These actions may not necessarily arise from predefined patterns of interaction, they may arise from the history of the interaction, the current state of the system, or the users intentions. Second, in temporal structured multimedia systems, specifying a progressive plot, the temporal structure of the system is exactly a use pattern. Any user interaction with a temporal structured multimedia system, will undertake the predefined plot of progression, possibly with some deviations dependent on the sophistication of the plot structure. But whereas use patterns by definition are relatively small sequences of interaction with the system, a temporal plot structure is a specification of the complete interaction, including a beginning, middle, and an end.

Another assumption of OOA&D failing to meet the form and structure of interaction with a multimedia system, is the assumption that there is a strong relationship between real world activities (tasks in the application domain) and what is represented in the system. This assumption does not hold for multimedia systems, rather the situation is the opposite. Multimedia systems at best simulate real world activities, and in case they do, these activities are not necessarily related to tasks in the users environment.

The everlasting principle of "the inseparability of form and content" is the key to the conflicts between method concepts and multimedia system development. The temporal plot structure is a form for how to present the actual content of a multimedia system. The observation is that a temporal plot structure is one single use pattern, and that spatial plot structures does not afford specification of use patterns lead us to conclude, that the concepts of actor and use pattern, not immediately can transcend to multimedia system development.

4. CONCLUSION

This paper has illustrated some of the discrepancies between the traditional software engineering concepts for modelling computer systems and requirements for multimedia systems development. Our main critique of the concepts applicability in multimedia systems development is on the basic assumption of a single situation compromising a problem domain and an application domain. We have argued by example that this initiation view of a situation is not applicable in multimedia systems development. The consequence of this observation is far reaching for applicability of the method, because the conceptual framework of terms created in the method builds on the one situation assumption. As argued in this paper both the problem domain activity and the application domain activity builds on the one situation perspective, and hence these activities can not inform multimedia systems developers with the same degree of detail as the method afford more traditional systems development.

Future research within the field may show how methodological support can be provided the development of multimedia systems. The key question could be whether methods are appropriate or applicable in multimedia systems development at all. Some would probably argue that creating and designing multimedia systems is an even more creative activity than designing traditional systems and hence, one cannot set up activities and guidelines to be followed for this kind of systems development. Webb (1996) argues that multimedia systems are more theatres than software tools and should though be treated as such. However, Eriksen, Skov, and Stage (2000) found that multimedia systems developed as a narrative or movie production suffer in software quality with respect to performance and stability. This is supported by Sutcliffe (1994) saying that most multimedia systems are developed and created by intuition. However, we believe that more empirical studies of the design process of multimedia systems are needed like the one presented in Eriksen, Skov, and Stage (2000).

ACKNOWLEDGEMENT

The research behind this article has been partially financed by the Danish Research

Councils' joint Multimedia Program No. 9600869. We would also like to thank various anonymous reviewers for comments on earlier drafts of this paper.

REFERENCES

Booch, G. (1994). *Object-Oriented Analysis and Design with Applications*. Benjamin/ Cummings, Redwood City, California.

Druin, A. and Solomon, C. (1996) *Designing Multimedia Environments for Children.* John Wiley & Sons.

Eriksen, L.B., Skov, M., and Stage, J. (2000). An Empirical Study of Two Training and Assessment Multimedia Design Processes. Submitted for publication.

Fairley, R. E. (1985). *Software Engineering Concepts*. McGraw Hill.

Jackson, M. (1975). *Principles of Program design*. Orlando, Academic Press.

Jackson, M. (1983). *System Development* . Prentice-Hall, Englewood Cliffs, New Jersey.

Jacobson, I., Christerson, M., Jonsson, P., and Övergaard, G. (1992). *Object-Oriented Software Engineering*. Addison-Wesley, Wokingham.

Grosky, W. I. (1994) Multimedia Information Systems. In *IEEE Multimedia*, no. 1, pp. 12 – 23

Lewis, C., Brand, C., Cherry, g., and Raber, C. (1998). Adapting User Interface Design Methods to the Design of Educational Activities. In proceedings of *Human Factors in Computing Systems: CHI98.* April 1998

Kautz, K., Malmborg, L., and Pries-Heje, J. (1998). Does University Education Lead to Adoption. In: *Proceedings of the IFIP WG 8.2 and 8.6 Joint Working Conference on Information System: Current Issues and Future Changes.* Helsinki, Finland, 10-13 december.

Mathiassen, L., Munk-Madsen, A., Nielsen, P. A., and Stage, J. (1997). *Object-Oriented Analysis and Design* (in Danish). Aalborg, Marko.

Mathiassen, L. (1997). Reflective Systems Development. In *Scandinavian Journal of Information Systems.*vol. 10, no. 1&2, 1998, pp. 67-117

Mathiassen, L., Munk-Madsen, A., Nielsen, P. A., and Stage, J. (1999). *Object-Oriented Analysis and Design*. Aalborg University.

Monarchi, D.E., and Puhr, G.I. (1992). A Research topology for Object-Oriented Analysis and Design. In: *Communications of the ACM.* September 1992, vol. 35, no. 9.

Myers, G. (1978). *Composite/Structured Design.* New York, NY: Van Nostrand Reinhold.

Nemetz, F and Johnson, P. (1998), Developing Multimedia Principles from Design Features, in A. Sutcliffe, J. Ziegler & P. Johnson (eds*), In Proceeding of Designing Effective and Usable Multimedia Systems*, Kluwer Academic Publishers, pp.57-71.

Orr, K. (1971). *Structured Systems Development* . New York, Yourdon Press.

Plowman, L. and Luckin, R.. (1998) Designing Multimedia for Learning: Narrative Guidance and Narrative Construction. In proceedings of *Human Factors in Computing Systems: CHI98.* April 1998

Rumbaugh, J., Blaha, M., Premerlani, S., Eddy, S., and Lorensen W. (1991). *Object-Oriented Modelling and Design*. Prentice-Hall, Englewood Cliffs, New Jersey.

Sommerville, I. (1992). *Software Engineering*. 4th edition. Addison-Wesley, Workingham.

Sutcliffe, A.G. and Faraday, P. (1994). Designing Presentation in Multimedia Interfaces. In: Adelson, B., Dumais, and Olson, J. (Eds.), *Proceedings of Computer-Human Interaction Conference '94*. 92-98.

Sutcliffe, A.G. and Faraday, P. (1997). Designing Effective Multimedia Presentations. In: Ware, C. and Wixon, D. (Eds.), *Proceedings of Computer-Human Interaction Conference '97.*

Yourdon, E. and Constantine, L. (1979). *Structured Design*. Prentice-Hall, Englewood Cliffs, New Jersey.

Umaschi Bers, M., Ackermann, E., Cassell, J., Donegan, B., Gonzales-Heydrich, J., DeMaso, D. R., Strohecker, C., Lualdi, S., Bromley, D., and Karlin, J. (1998). Interactive Storytelling Environments: Coping with Cardiac Illness at Boston's Children's Hospital. In proceedings of *Human Factors in Computing Systems: CHI98.* April 1998

Webb, B. R. (1996). The role of users in interactive systems design: when computers are theatre, do we want the audience to write the script. *Behaviour and Information Technology.* Vol. 15, no. 2, pp. 76 - 83

Züllighoven H. (1998). Das Objektorientierte Konstruktionshandbuch nach dem Werkzeug & Material-Ansatz. dpunkt.verlag, Heidelberg, Germany