Supporting early design stages of technical systems by knowledge-based simulation

U. Langer and A. Lehmann
Federal Armed Forces University Munich, Computer Science Department, D-85577 Neubiberg, Germany
Tel: 49-89 6004 2648 (A. Lehmann) #2547 (U. Langer)
Fax: 49-89 6004 2268
E-mail: {langer, lehmann}@informatik.unibw-muenchen.de

Abstract

Modelling of technical systems requires well founded knowledge not only in the application domain, but also about methods and tools for modelling and simulation, statistics, model verification and validation. Knowledge-based systems can provide useful support in these areas. In this contribution a conceptual framework and three projects on knowledge-based simulation realised at the Computer Science Department of the Federal Armed Forces University Munich are presented. They refer to the support of a problem adequate model specification and construction, to the selection of a modelling technique, to the check of model quality and validity, and to the modelling and performance evaluation of computer systems.

Keywords
Simulation, modelling, design, knowledge based systems

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35357-9_22

A. B. Baskin et al. (eds.), *Cooperative Knowledge Processing for Engineering Design*

1 INTRODUCTION

Short innovation cycles and a permanently growing complexity determine the design of innovative technical systems. This situation results in an increasing probability of design faults and bottleneck implementations. If such problems are discovered only in advanced stages of a system design or even not until the system has gone into production, corrections become very expensive. So these kinds of problems have to be discovered as early as possible. Because the system under design not yet exists for measurement, models have to be used for the analyses of the system's behaviour.

In early design stages normally quite a number of design parameters are still subject of changes. Different alternatives have to be evaluated and compared. This means that a modelling and analysis tool for early design stages should be distinguished by criteria like small expenses for model construction and modifications, support of well-considered abstractions, and acceptance of incomplete specifications. Nevertheless, the user should be able to specify all relevant aspects of the real system.

As a consequence, the support of early analyses of the dynamic behaviour of a complex system requires a top-down modelling approach with reusable model components and the availability of system and modelling knowledge. Relevant parts of the system should be describable in more detail than the others. This necessitates the support of hierarchical descriptions. Abstractions of the system behaviour are realised by the utilisation of heuristics and probabilistic values. Especially in the early design stages the specified model parameter values are also often more or less uncertain. The influence of these values and the resulting importance of their optimisation should therefore be estimated by sensitivity analyses.

2 THE IMPACT OF KNOWLEDGE-BASED SYSTEMS AND SIMULATION

For the analysis and prediction of dynamic system behaviour, for documentation and visualisation of system knowledge, as well as for training and tutorial systems simulation is a well established method. It is often used for predictions referring to the design, selection, modification, configuration, or capacity planning of technical systems. The importance of simulation can be recognised by a permanently increasing variety of simulation tools and techniques. Unfortunately, most tools and techniques have limited application domains. Thus, a potential user needs not only knowledge about the application domain to create an appropriate and efficient model. Just as much knowledge is required about modelling methodologies, simulation techniques and tools, and statistical analysis. In addition, he/she should

be well experienced in the verification, validation, and calibration of models - a fact which is often rather neglected.

Because nobody is expected to be an expert in all of these areas, an ideal modelling and analysis tool should possess features like decision support, a user-adaptable interface, and representation techniques for various categories and qualities (e.g. uncertain and fragmentary) of knowledge. It should also advise the user in model construction, validation, experimentation, and result interpretation. Knowledge-based systems (KBS) offer techniques to fulfil these requirements. They can contain the - formalised - knowledge of multiple experts. For the representation of vague knowledge and for problem solving heuristics are applied. The user can also request for explanations of certain conclusions of a KBS. For those reasons a combination of KBS and simulation systems can be very profitable. (O'Keefe, 1996, Lehman, 1987, 1990, 1993, Merkuryeva and Merkuryev, 1994)

Figure 1 shows the structure of a knowledge based design environment with co-operating knowledge sources for modelling, for analysis, for the system under design, and for the design process. These knowledge sources could be further refined (e.g. *analysis knowledge* could be refined in *functional analysis* and *performance analysis* knowledge) or supplemented by additional sources.

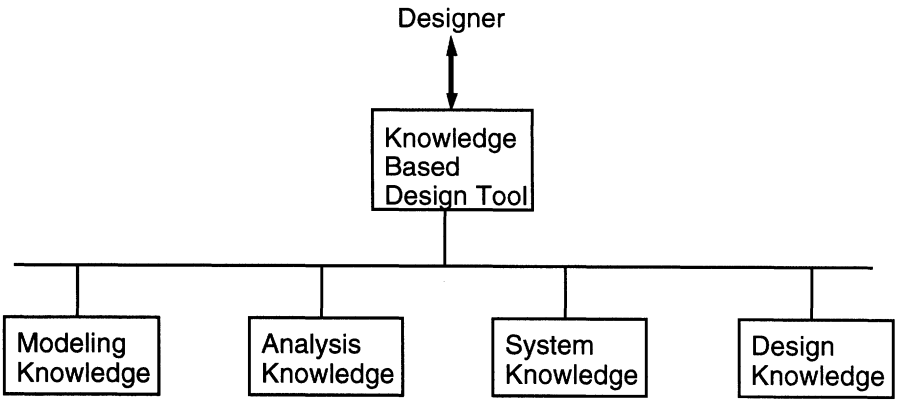

Figure 1: Co-operating knowledge sources.

3 A KNOWLEDGE-BASED SIMULATION ENVIRONMENT FOR SYSTEMS DESIGN

Regarding the potential of KBS for supporting simulation, we have developed a general framework for the application of knowledge-based simulation in this

context, summarised in chapter 3.1. Examples for its realisation and experiences for practical applications are presented in section 3.2.

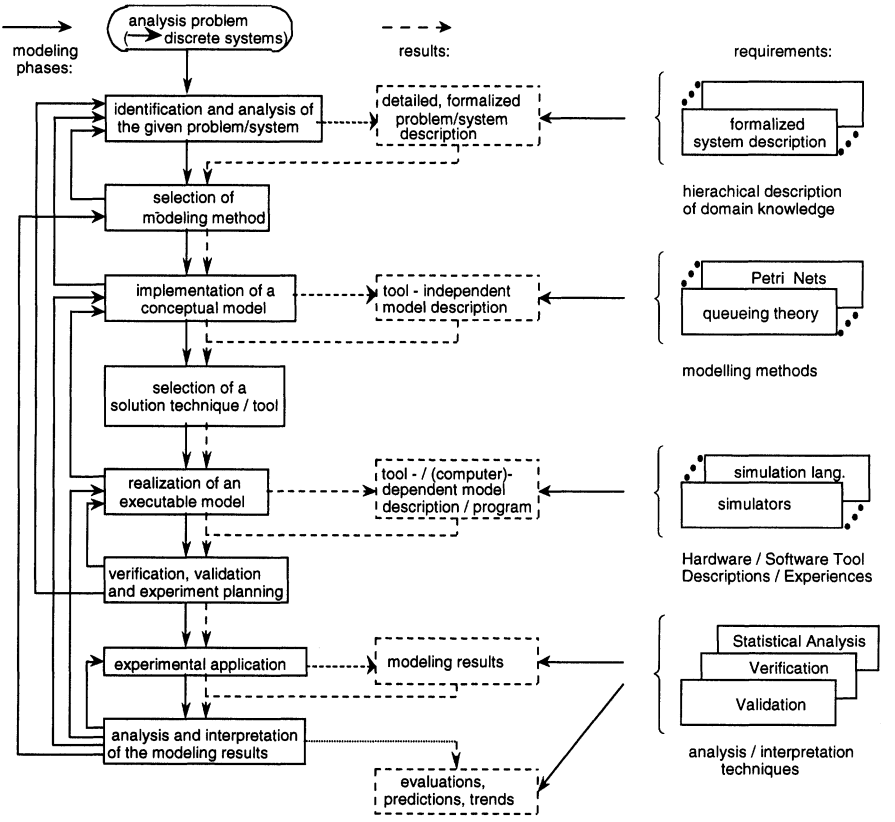

Figure 2: Phases, requirements and results of a modelling process.

3.1 Basic concept

Figure 2 shows the ***phases, requirements*** *and* ***results*** *of a well structured modelling process.* The first step is to analyse the given problem. This should result in a formalised problem description, detailed enough to construct a suitable model. Based on this information, an *adequate modelling technique* has to be selected. Only then the - conceptual - model should be constructed. Conceptual means that the model is to be specified according to the selected modelling technique but not yet in the syntax and with the restrictions of a certain tool. The tool selection and construction of a tool- and machine-dependent model has to be

done in the next modelling phase. This partitioning of the modelling process is in contrast to the quite common procedure, where a specific tool is given and the user directly tries to specify his problem as a tool-dependent model. The advantage of the above procedure is, that the problem / system specification and the constructed conceptual model are independent of the features and limitations of a certain technique or tool.

After the realisation of an executable model, the model has to be *verified and validated.* This phase is often treated like a stepchild, because it is time-consuming and the cost increases exponentially with the aspired degree of validity. But nevertheless, model credibility is very important and if a good compromise is found between effort and degree of validity, some much more expensive iterations in the modelling cycle - respectively in the system re-design - could possibly be avoided.

Each phase in this modelling process requires certain *types of knowledge*, represented in the right column of Figure 2. A knowledge-based system can support the user by supplying such information. In the following sections results and experiences of three projects on knowledge-based simulation realised at the Computer Science Department of the Federal Armed Forces University Munich will be presented. The projects are strongly related to the steps and phases of a modelling process described above and cover the topics:

- support of a problem adequate model specification and construction;
- selection of a modelling technique;
- check of the model's quality and validity;
- modelling and performance evaluation of computer systems.

3.2 Examples and experiences

In this section concrete applications of the general (basic) knowledge-based simulation concept are presented. The described projects led into prototypes realised with KEE and SimKit, trademarks of IntelliCorp. KEE is a lisp-based tool for the design of knowledge-based systems. SimKit is built on top of KEE and offers in addition a simulation environment.

3.2.1 Selection of a modelling environment

Figure 3 outlines the types of knowledge a problem specification may contain (phase 1; Figure 2). The environment for the *model construction support* is based on this structuring of the problem specification. It consists of an interactive, graphical interface for the system description with generic objects on predefined abstraction levels and the menu driven acquisition of analysis goals and constraints of the considered problem.

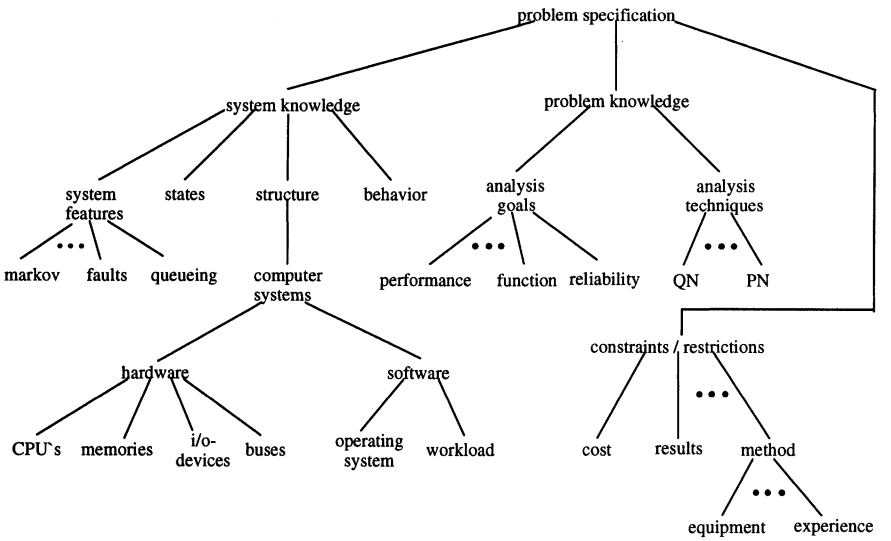

Figure 3: Structuring of a KB for a problem specification (only classes displayed).

Figure 4 shows the composition of the canvas for the *specification of a system structure*. The system's states and dynamic behaviour are specified according to the structure on a separate canvas. The user selects an abstraction level by mouse click. On each abstraction level he is offered only those generic objects applied at this level for usage in the design window. The features of the model objects are specified by parameter values on the right side of the canvas. The picture outlines a specification of a memory on the module level. As an example for a parameter, the capacity in MB is displayed. Parameter values can be provided by the manufacturer, measured or simply estimated. To distinguish the quality of the information for each value the source also has to be specified. In the given example the memory capacity is assumed to be read out from data sheets (i.e. manufacturer). (Schwarz, 1990)

This specification process is supported by the usage of knowledge about modelling and about the analysed system. The next step, the selection of a modelling technique, requires knowledge about the system, as well as about modelling tools and techniques, and about the analysis goals.

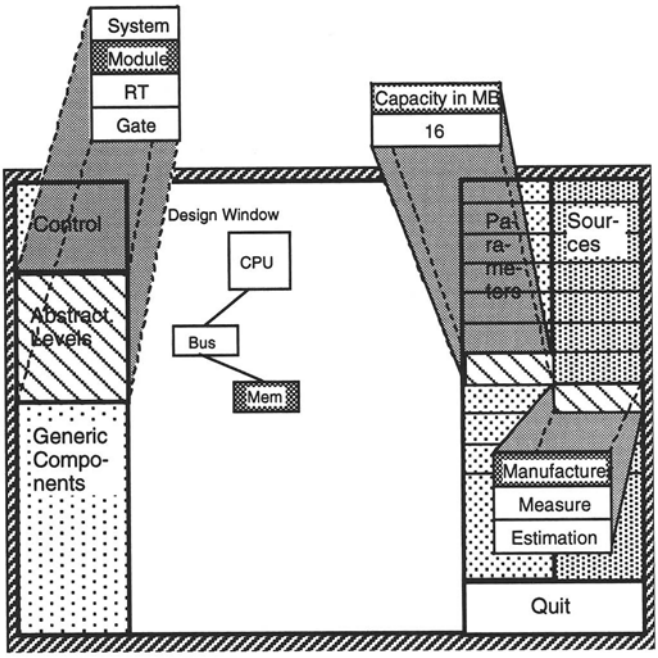

Figure 4: System structure specification canvas.

The tool for the *selection of a problem-oriented modelling technique* checks the problem specification and suggests an adequate technique. This is accomplished by the evaluation of different parameters, like the abstraction level of the model (see Figure 5). For every available modelling technique, the suitablility in relation to each evaluation parameter is determined. The most suitable method (according to the parameters taken into account up to this point) can be displayed in a separate window. The selection environment distinguishes between obligatory parameters, which are always questioned, and facultative parameters, which are only suggested to the user for evaluation. (Leitges, 1990)

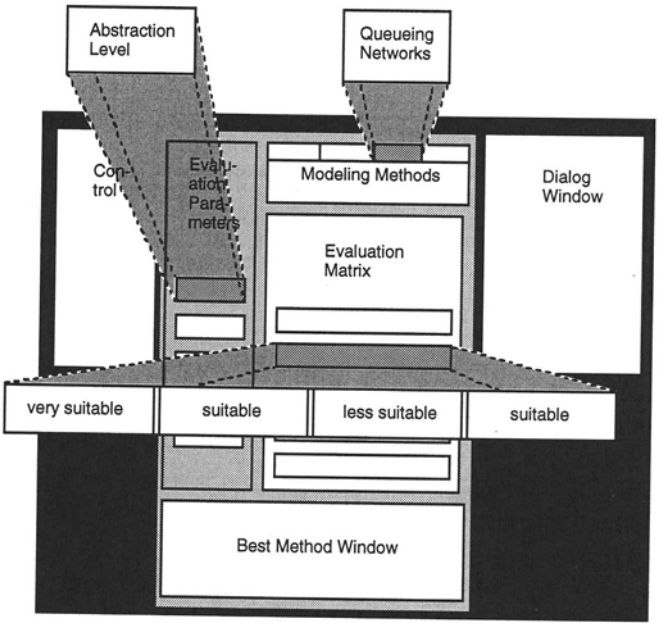

Figure 5: Modelling technique selection canvas.

3.2.2 Verification and validation of Petri nets

The *model verification and validation tool* checks timed Petri Net models for their correspondence with the system specification, and for weaknesses and faults in the constructed model. Thus, it uses system and modelling knowledge. The tests are divided into those independent of a specific modelling technique and into Petri Net oriented tests (Figure 6). The first step is the automatic association of real world objects with model objects (by equality of names). Transition firing times are checked against real world actions times given in the problem specification. Associations of one to multiple objects and impossible real world transitions in the model are searched. Leaf objects are examined to justify their particular position. To support the following model verification, additional information can be specified: cycles, sets of places where always a certain number of states is marked, and s-invariants, i.e. sets of places with a constant number of tokens.

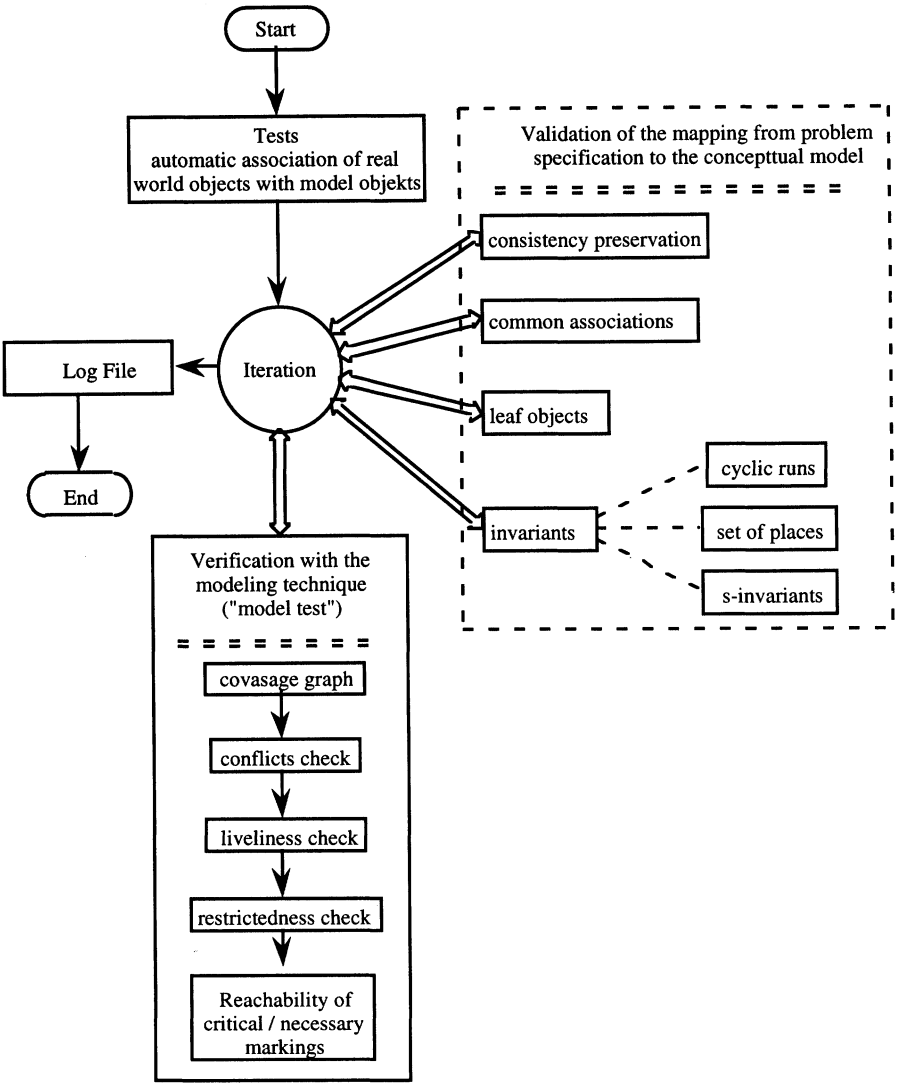

Figure 6: Verification and validation process.

This validation process is documented in a log file. If any inconsistencies are discovered the user has to correct them or to justify them by a comment. These changes or comments are also part of the validation document !

The next phase is the *verification with the modelling technique*, i.e. timed Petri nets. It starts with the generation of a coverage graph, which shows the - more or

less covered - reachable markings and the transitions between them. This coverage graph is the base for individual checks. *Conflicts* are situations where a non-deterministic choice between different actions is necessary. This is an indication for a potential model fault, because such situations do not exist in the real system. *Liveliness* means, that for all the observed transitions there exists no situation where they can no longer be activated. *Restrictedness* stands for the demand for places with a limited capacity. Finally, the user can specify *critical or necessary markings* and let the tool trace the coverage graph to check their reachability. These results are also included in the log file. (Müller, 1990)

3.2.3 Performance evaluation of computer systems

The *performance analysis and evaluation of computer systems* raises a lot of problems. One of the biggest is to find an architecture independent and representative description of the computer's workload. This applies especially to the case of a computer system design, where the prospective usage of the system can only be estimated. The question, what features of the real system can be neglected, is difficult to answer because interactions and interdependencies between architectural and workload parameters are hardly understood. Abstractions require heuristics (e.g. for access hit rates, resource conflicts), which have to be found in literature, acquired from domain experts or estimated more or less arbitrary. This leads to another problem: a lot of information usually already exists for a computer system design, e.g. because the system is built around an already existing CPU, memories etc. Nevertheless, this knowledge is often not directly available to the modelling tool, but has to be found in different sources and fed into the tools' database by the user.

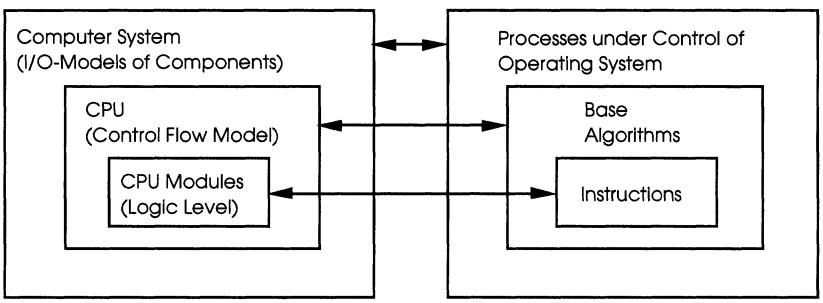

Figure 7: Hardware-/workload-hierarchy.

Because of the difficulties concerning performance evaluation, it is usually isolated from the actual design process and often incorporated too late.

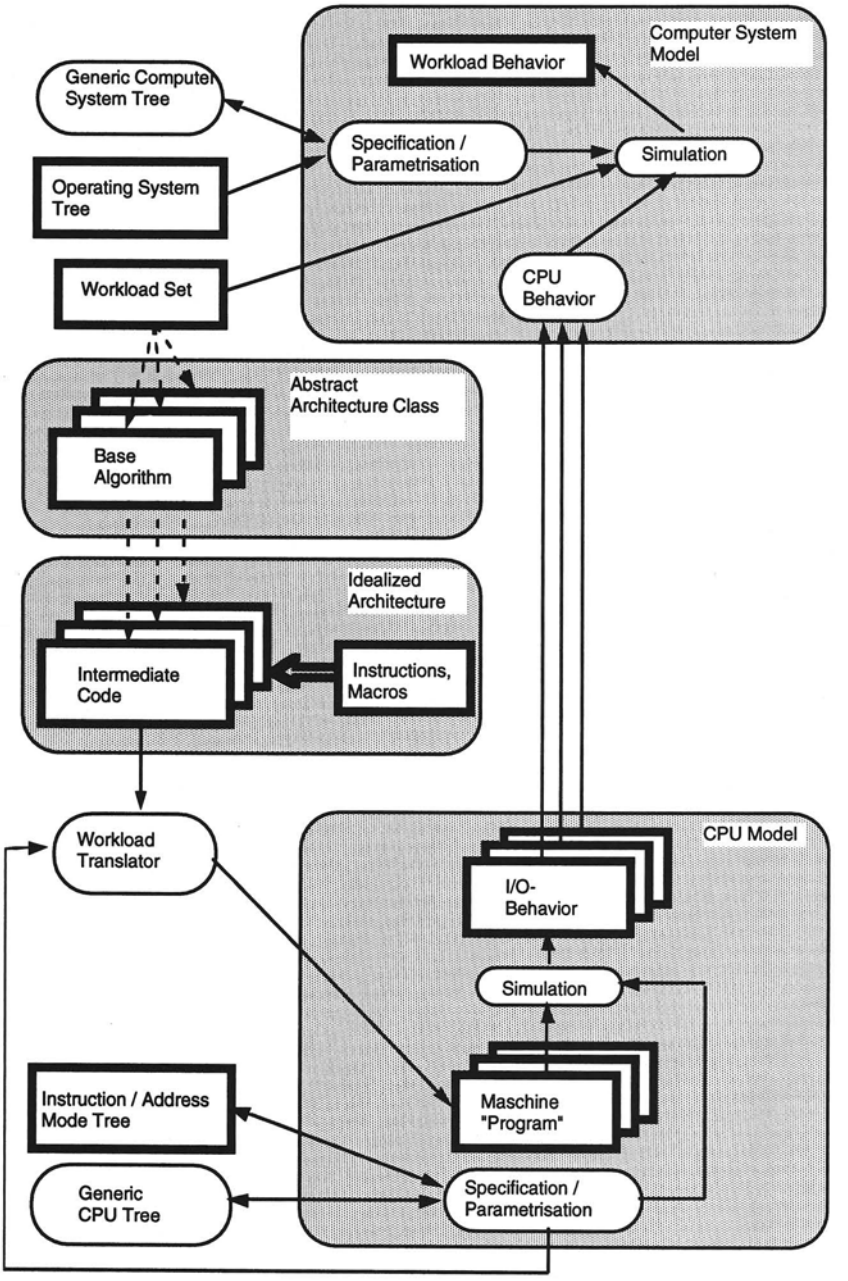

Figure 8: A simulation environment for performance evaluations of CPUs and computers.

To improve this situation, especially to incorporate performance evaluations at an earlier stage of a system's design, a concept has been developed, that offers generic objects for the modelling of CPUs and computer systems, as well as for the associated workload. By an "interface" between the two worlds of CPU and computer design, the results of a CPU analysis can be used as a description of the dynamic behaviour of the device in a computer system model. Thus, knowledge about modelling and about performance evaluation, knowledge about computer and CPU design, and system knowledge are incorporated in this concept.

The basic idea behind this concept is to associate with each hardware description level an adequate standard workload (Figure 7). Base algorithms, small common programs for typical tasks within real applications, are used to analyse CPUs during their design. The I/O-behaviour of these devices while executing the base algorithms is analysed, recorded and reused in computer system models, where processes are composed of these base algorithms. This results in the simulation environment represented in Figure 8.

At first, the base algorithms are compiled into an intermediate code. The CPU designer specifies his device as a control flow model. This means that all functional units of the CPU pipeline, caches and memory management unit are represented in the model; but their functionalities are only given by the execution times of their operations. All hardware devices and sub-devices, standard CPU instructions and workloads are stored as generic objects in knowledge bases. A workload translator generates instruction sequences for the analysed CPU out of the intermediate codes for the base algorithms. The CPU-designer analyses his device by simulation, and if he is satisfied he records its I/O-behaviour.

A computer system designer uses this information in his model to simulate the dynamic behaviour of his system for a certain set of processes with regard to the influence of the operating system. (Herzog, 1993, Langer, 1992, 1993, 1994, Weber, 1993)

3.2.4 Experiences

The tool for *problem specification and selection of a modelling technique* offers a good, extendable approach for the improvement of the first modelling phases (see Figure 2). Nevertheless, dependencies among different parameters, the representation of vague knowledge or the influence of not acquired information are topics which are not yet fully considered. The evaluation of the methods' suitability is quite straightforward (arithmetic mean of the individual values) and could be improved. Unfortunately, up to now there are only a few factors for the - desirable case of - exclusion of modelling techniques included.

The successful *model verification* very much depends on the complexity of the coverage graph, which grows very fast as the number of places is increased. The problem gets even worse, because the coverage graph has to be calculated again after each change of a model. A reduction of complexity could be achieved by the usage of subnets which is not yet implemented. In addition, up to now the

verification is only realised for timed Petri Nets. But despite these restrictions, it shows a way for better verification and validation of Petri Net models and the suitability of the proposed general approach.

The simulation environment for the *performance evaluation of CPUs and computer systems* enables a high degree of information reuse. It eases the difficult task of workload specification by mapping architecture independent standard workload descriptions to a specific hardware. Due to the specialisation of the application area, generic model objects with a quite extensive functionality can be offered to the designer. In a computer system model the processing of a program by a CPU is abstracted to the simulation of the corresponding I/O-behaviour of the device. All this results in a pronounced reduction of the model construction and simulation effort. Very detailed models can be specified in a short time. The hierarchical structure of the hardware and software model objects promises more insight in the dynamic behaviour of recent computer systems.

4 OUTLOOK

A carefully executed modelling process consists of some difficult, time consuming, and costly tasks. For some specific tasks, simulation techniques are not well applicable or efficient. Beyond it, expenses increase dramatically, the closer one tries to get to perfection in performing the individual tasks. For this reason, some of them are often treated very superficial or carried out very lately when almost all decisions are made. Knowledge-based systems, which support users in these tasks and help to reduce the cost, can be applied very profitable in this area. Three examples for the fruitful co-operation of KBS and simulation were presented in this paper. This co-operation of KBS and simulation concerns *decision support, knowledge classification, the consideration of vague knowledge* and the *object-oriented knowledge representation* with generic objects and inheritance.

Of course, these prototype realisations are only approaches and offer multiple possibilities for improvement. The selection of a modelling method could be extended by more sophisticated evaluation techniques, which also consider interdependencies of parameters and the quality of knowledge. The verification and validation tool should support additional modelling techniques, e.g. queuing networks. The complexity should be reduced by incorporating hierarchical modelling.

Although the three projects used computer system models as examples, their application is not limited to this domain. Even the simulation environment for performance evaluations of computer systems is based on ideas, which could also be applied to the analysis of other "adaptable" technical systems. Imagine a manufacturing plant, where the results of the analyses of the individual manufacturing stations for different typical tasks are used to build a model of the

whole plant. (Though in this case the conditions seem to be easier to cope with than in the computer system design area !)

The performance evaluation environment is useful for mono-processor systems and multiprocessor systems with a small number of processors. For higher degrees of parallelity, the administration overhead for the individual processors would be to high. Anyhow, in these kinds of systems the communication cost often is the critical point and the CPU behaviour should be further abstracted. Improved memory architectures could contribute to a reduction of communication in computer systems and networks. Especially for application areas like multimedia or distributed databases, the memory management is a very important point. This will be one of our forthcoming research activities.

Innovative computer architectures and applications (e.g. multimedia) require the construction and solution of even more complex models. For these new applications, questions like

- adequate workload representations;
- suitable modelling scenarios;
- the representation of a priori knowledge;
- the reusability of fragmentary results.

have to be posed and answered and new. co-operative knowledge sources can be very helpful in coping with these problems.

5 REFERENCES

Herzog, M. (1993). "Realisierung einer Spezifikations- und Simulationsumgebung für Rechner-analysen". *Diploma Thesis, Fak. für Informatik, Universität der Bundeswehr München.*

O'Keefe R.M. (1986). "Advisory systems in simulation", in: E.J. Kerckhoffs, G.C. Vansteenkiste, B.P. Ziegler (Eds.); *AI Applied to Simulation*, Simulation Series, Feb. 1986: 73-78.

Langer, U. (1992). "A Concept for the Design and Analysis of Computer Systems in a Knowledge Based Environment.". *Proc. European Simulation Symposium, ESS '92*: 236-240.

Langer, U. (1993). "Eine generische Lastbeschreibung für frühzeitige Leistungsanalysen beim Prozessor-Entwurf". *Messung, Modellierung und Bewertung von Rechen- und Kommunikations-systemen - Kurzberichte und Werkzeugvorstellungen, Aachener Beiträge zur Informatik, Band 2:* 19-24.

Langer, U. (1994). "HARUSPECS - A HierARchical Uniform Simulation environment for Performance Evaluations of Computer Systems". *To be published: Proc. European Simulation Symposium, ESS '94.*

Lehmann, A. (1987). "Taxonomy and Application of Expert Systems in Simulation". *Proc. of the International IMACS-Symposium on AI, Expert Systems and Languages in Modelling and Simulation.*

Lehmann, A. (1990). "Knowledge-Based Systems to Support Dynamic Process Simulation". *Proc. of the 2nd European Symposium "Nuclear Simulation"*: Springer-Verlag, 1990.

Lehmann, A. (1993). "Methodologies and Applications of Knowledge-Based Hybrid Systems". *Preprints KNOWHSEM '93*: 19-34.

Leitges, C. (1990). "Unterstützung der Auswahl der Modellierungsumgebung zur Leistungs-analyse von Rechensystemen". *Diploma Thesis, Fak. für Informatik, Universität der Bundeswehr München.*

Merkuryeva G.V., Merkuryev, Y.A. (1994). "Knowledge Based Simulation Systems - A Review". *Simulation, Feb. 1994: 74-89.*

Müller, J. (1990). "Entwicklung eines wissensbasierten, interaktiven Diagnosesystems zur Verifikation gezeiteter Stellen-/Transitions-Netzen". *Diploma Thesis, Fak. für Informatik, Universität der Bundeswehr München.*

Schwarz, R. (1990). "Entwurf und Implementierung eines Wissensakquisitions-Interface zur Analyse von Rechensystemen". *Diploma Thesis, Fak. für Informatik, Universität der Bundeswehr München.*

Weber, J. (1993). "Realisierung einer Spezifikations- und Simulationsumgebung für CPU-Steuerfluß-Analysen". *Diploma Thesis, Fak. für Informatik, Universität der Bundeswehr München.*

6 BIOGRAPHIES

Axel **Lehmann** is full professor for computer systems architecture and operation at the computer science department of the Federal Armed Forces University Munich. He published numerous papers and books on modelling and simulation. He is speaker of section 3 of the German Society for Informatics (GI) "Technical Informatics and Architecture of Computer Systems" and Vice President of the Society for Computer Simulation International (SCS). His research interests cover knowledge-based simulation, intelligent tutoring systems and knowledge-based diagnosis.

Uwe **Langer** received his diploma degree in electrical engineering from the Technical University Darmstadt in 1987 and his Ph.D. (Dr. rer.nat.) from the Federal Armed Forces University Munich in 1995 for a work on performance evaluation of prospected computer systems. His research topics are modelling simulation, and performance evaluation of computer systems.