Combinatorial Modeling Techniques in Conjoint Simulation

Axel Hein and Wolfgang Hohl
Institute for Computer Science III (IMMD III)
University of Erlangen-Nürnberg
Martensstr. 3
D - 91058 Erlangen, Germany
e-mail: alhein@immd3.informatik.uni-erlangen.de

1. Introduction

The performability analysis of computer systems is a complex and important issue. Massively parallel computer systems designed for the computation of critical and time-consuming applications have to provide fault-tolerance mechanisms in order to tolerate failures of components and to continue working even with deteriorated performance. Sophisticated and enhanced modeling tools and techniques are required to evaluate computer systems as soon as possible during the early design stages. We present a modeling framework called Conjoint Simulation [9], and put the focus on the use of combinatorial modeling techniques which are well known and frequently used in dependability analysis.

Conjoint Simulation strives to fulfill some of the requirements which have been raised in a survey of performability analysis [13], and which might play an important role in the future of the combined evaluation of performance and dependability. For instance, the improvement of the model construction interface is demanded by applying techniques which are familiar to system designers; the design-oriented model and the evaluation-oriented model are distinguished, where the first one is automatically converted into the latter representation form. Design-oriented models are close to the designer’s knowledge and abstraction, whereas evaluation-oriented models are representation forms which can be directly evaluated.1

2. Conjoint Simulation

We refer to the technique of combining object-oriented, process-based simulation models with Petri net and combinatorial dependability models as Conjoint Simulation to distinguish it from other hybrid modeling approaches. Hybrid approaches typically simplify a detailed model and combine simulation and analytical techniques to reduce the time needed to evaluate a model; these techniques provide an approximation of the results obtained from a detailed model. The primary objective of Conjoint Simulation, however, is to facilitate the representation of a system in detail so as not to abstract away system characteristics that are crucial to the performability of the system.

To design a Conjoint Simulation model, we partition the target computer system into four domains: architecture, workload, failure modes, and repair-maintenance mechanisms. These four characteristics cannot easily be integrated and varied in a monolithic model. Having investigated various modeling and representation techniques, we feel there is no single modeling technique that facilitates the representation of these four key characteristics in an optimal and easy-to-handle manner. Therefore, different modeling techniques, which are well-suited to represent the various parts of the system, should be used and combined to form the overall model.

With Conjoint Simulation, the designer is encouraged to develop parts of the system model independently and with different techniques. In fact, the system model consists of an architecture-workload model (AWM) and a failure-repair model (FRM). Breaking down the system model into an AWM and a FRM simplifies the representation and allows separate, independent evaluation of the two models. The AWM can be evaluated to determine the performance characteristics of the target system, and the FRM can be analyzed to investigate its dependability. Finally, the two models are combined to conduct performability analysis. Yet another advantage is that a given AWM can be evaluated with various FRMs, and vice versa. Usually, the AWM is the more complex part of the system. By separating the failure, repair, and maintenance characteristics into a separate FRM, the user can readily experiment with different FRMs without any changes to the AWM.
The **AWM** is based on object-oriented modeling and process-based simulation, while we use timed Petri nets and combinatorial models such as series-parallel diagrams and fault trees to construct the **FRM**. We use a version of timed Petri nets where time is assigned to the transitions of the Petri net (**TTPN** - timed-transition Petri net); these **TTPN** correspond to the well-known **GSPN** (generalized stochastic Petri net [12]) apart from the fact that also non-exponential distribution functions are allowed.

A simple model of a computer system is shown in Figure 1 illustrating the basic modeling concepts of the **AWM**. The model consists of two processor objects prc1 and prc2 and a link object lnk1. The link object simulates a generic link through which data are passed between a source and a destination. A simulation process is assigned to each processor object to perform the scheduling of workload processes (c10 and c20 in Figure 1). The simulation process c10 of the link object models the forwarding of messages sent between prc1 and prc2. The workload is represented by the two workload processes c11 and c21 which are assigned to the processor objects prc1 and prc2, respectively; c11 and c21 may model data processing such as numerical algorithms as well as the sending and receiving of messages.

![Figure 1 Example of an AWM](image1)

The interaction between **AWM** and **FRM** is based on the events of the **TTPN** model, that means on the enabling or firing events of transitions in the **TTPN** (an example is given in the paragraph below and in Figure 2). Therefore, two sets CAi and CFi are assigned to each transition ti of the **TTPN** representation of the **FRM**. The sets CAi and CFi contain the operations which are performed in the **AWM** as soon as transition ti is enabled or fires, respectively. Thus, we specify methods of the **AWM** which are invoked as soon as a transition is enabled or fires. It should be noted that these operations performed in the **AWM** can not only start activities to influence and alter the behavior of the **AWM**, but they can also query the status of the **AWM** in order to control and modify the **FRM**.

The **FRM** is typically represented as a **TTPN**. The example in Figure 2 shows a **TTPN** whose transitions control and trigger activities of the **AWM** (in the figures, thin bars represent immediate, i.e. timeless, transitions, and filled boxes symbolize timed transitions). As soon as the transition t_inj is enabled the applications of the workload model are started or, if the applications have been interrupted by an error or recovery mechanism, the applications are rolled back and are restarted from the last valid checkpoint. After a period of time defined by the distribution function f_inj of the timed transition t_inj, an error is injected into one of the processor objects of the **AWM**. After the firing of t_inj, the timed transition t_recover is enabled and the applications of the workload part of the **AWM** are stopped. The distribution function f_recover assigned to t_recover defines the time between enabling and firing of t_recover; when t_recover fires the faulty component of the **AWM** is replaced and the **AWM** is reconfigured. Finally, t_inj is enabled again.

![Figure 2 TTPN example](image2)

### 3. Combinatorial modeling techniques in Conjoint Simulation

Petri nets are the key modeling technique of the **FRM**; the development of large-scale and complicated Petri net models, however, is cumbersome and error-prone. Places, transitions, and arcs have to be correctly positioned to depict the logical relationships of the target system. For quantitative analysis, time behavior has to be assigned to the transitions, and the proper memory policies have to be chosen to model the dynamic and temporal characteristics. Therefore, we use an approach which is intended to retain the flexibility and the modeling power of Petri net modeling, and to facilitate the model design and model development for dependability analysis. For this objective, we use the graphical representations of series-parallel diagrams and fault trees. An interesting discussion of various methods used in dependability modeling has been presented in [10], where the modeling power of combinatorial and Petri net techniques is compared.
3.1 Conversion of combinatorial model types to Petri net representations

Series-parallel diagrams (SD) and fault trees (FT) are widely used techniques for the dependability analysis of non-repairable systems [15]. Typically, modeling tools providing methods for the analysis of these kinds of models do not consider repair. For instance in the modeling tool SHARPE, the analysis techniques for SD and FT models are specialized for reliability analysis and do not allow components to be repairable [14]. Only under very restricting conditions, SD and FT models can be used for the modeling of repairable systems [15]. Since we are using simulation to evaluate Conjoint Simulation models, we are able to take various distribution functions - such as deterministic time, Weibull or Normal distribution - to define time spans.

The equivalent SD and FT representations of a simple example comprising four components b1, b2, b3, and b4 are shown in Figure 3 and Figure 4. The system is in the failed state if b1, b2, and b3 or b4 are in the failed state.

- Each block is either repairable or non-repairable. In the first case, a repair distribution function for the time to repair is defined.

Since SD and FT techniques are well known and widely used in industrial applications, and the more powerful Petri net techniques are applied almost exclusively in academia, we take SD and FT techniques as the interface for model construction, and provide simulation-based evaluation techniques on the basis of Petri net models, which are automatically generated and are not visible to the model designer. For the ease of FRM modeling, we have implemented methods in the modeling environment SimPar providing a graphical interface for the development of SD and FT models, and automatically convert these models in corresponding TTPN representations [1], [4]. Furthermore, we extend the SD and FT representations by allowing shared repair facilities and repair policies such as fcfs (first come first served); these extensions are defined at the SD or FT level, but are evaluated exclusively in the Petri net representations [1]. Similar approaches have been discussed in [3] and [11]. Ereau et. al. use synchronized Petri nets in order to model in a FT local or global maintenance and further maintenance dependencies such as shared repair facilities [3]. An approach to incorporate various repair scenarios in FT models with repeated events is presented in [11].

The basic principle of the transformation from SD and FT models into TTPN models is the identification of the possible model states and the transitions between the states. Henceforth, we concentrate on the generation from a SD model without any loss of generality because of the correspondence of series diagrams and OR gates, as well as of parallel diagrams and AND gates. Obviously, each block, each series diagram, and each parallel diagram is in one of the two possible states, failed and ok. These states and the corresponding transitions have to be depicted in the TTPN, including the temporal characteristics. TTPN modules are automatically generated for each block, each series diagram, and each parallel diagram of a given SD, and these modules are connected by drawing arcs and transitions between them. The TTPN modules of a block, a series diagram, and a parallel diagram are shown in Figure 5, Figure 6, and Figure 7.

Since each block is either failed or ok, only one of the two places \( p^\text{fail}_b \) and \( p^\text{ok}_b \) contains a token at any point in time. The timed transition \( t^\text{fail}_b \) models the change of the state ok to the state failed, and also represents the time to failure. The distribution function assigned to \( t^\text{fail}_b \) is the failure distribution function of the block. Accordingly, the timed transition \( t^\text{rep}_b \) models the time till the repair of a faulty block. \( t^\text{rep}_b \) is included only if the corresponding block is repairable; in this case, the repair distribution function of the block is assigned to \( t^\text{rep}_b \). The marking of the
A series diagram is considered ok if all of its children are ok. Therefore, a token is removed from the place $p^\text{ser}_\text{num}$ when a child is repaired (Figure 6). If only one child has been failed and is repaired, the series diagram changes to the ok state. In this case, the immediate transition $t^\text{ser}_\text{rep}$ fires, the token in $p^\text{ser}_\text{fail}$ is taken away, and a token is deposited in $p^\text{ser}_\text{ok}$. The inhibitor arc between $p^\text{ser}_\text{num}$ and $t^\text{ser}_\text{rep}$ prevents $t^\text{ser}_\text{rep}$ from being enabled as long as at least one child is failed. The arc from $t^\text{ser}_\text{fail}$ to $p^\text{ser}_\text{num}$ ensures that the number of tokens in $p^\text{ser}_\text{num}$ remains equal to the number of failed children.

The TTPN module of a parallel diagram is similar to the one of a series diagram. Since a parallel diagram is in the failed state if all of its children are failed, the arcs between $p^\text{par}_\text{num}$ and $t^\text{par}_\text{fail}$ as well as the inhibitor arc between $p^\text{par}_\text{num}$ and $t^\text{par}_\text{rep}$ have multiplicity $n$ (Figure 7).

### 3.2 Conversion of the interactions between AWM and FRM

If the FRM of a Conjoint Simulation model is defined as a SD, operations are assigned to the blocks, series diagrams, and parallel diagrams; these operations launch activities in the AWM when the corresponding unit fails or is repaired. Operations in the AWM can be triggered when a block in the SD (or FT) fails or is repaired. Additionally, the failure and repair of a series diagram or of a parallel diagram (or of an AND gate or OR gate respectively) may launch operations in the AWM; for instance, the failure of a parallel diagram may cause the start of specific recovery processes in the AWM, or the repair of a series diagram allows a workload application to be restarted on the processor objects whose failure states are represented by the blocks of the series diagram.

Since the evaluation of the Conjoint Simulation model is performed at the TTPN level, the assignment of the operations has to be translated to the TTPN representation. During conversion of the SD or FT representation, the operations are automatically assigned to the transitions of the generated TTPN, which model the failure or repair of the respective unit.

For instance, the operations to perform in the AWM when a parallel diagram fails are assigned to the set $C^F_{\text{fail(par)}}$ of the transition $t^\text{par}_\text{fail}$. Equivalently, the operations which are launched when the series diagram is repaired are assigned to the set $C^F_{\text{rep(par)}}$ of the transition $t^\text{par}_\text{rep}$.

### 4. Conclusion

In this paper, we have presented a modeling approach which enables the model designer to specify the failure-repair behavior in the FRM as well as the interactions between AWM and FRM at the very descriptive level of SD and FT models. The conversion of the SD and FT representations of the FRM to the TTPN representation is automati-
cally performed; the Petri net representation is immediately linked to the AWM during model evaluation, or the model designer may modify and extend the Petri net performing a gradual refinement of the FRM.

The modeling techniques of Conjoint Simulation including the combinatorial model types are implemented in the modeling environment SimPar [6] which has been used for the performance and dependability evaluation of various multiprocessor systems ([2], [5], [7], [8]).

**Abbreviations**

- **AWM**  Architecture-Workload Model
- **FRM**  Failure-Repair Model
- **FT**  Fault Tree
- **SD**  Series-Parallel Diagram
- **TTPN**  Timed-Transition Petri Net

**Literature**