Modular Construction of Dependability Models from System Architecture Models:
A Tool-supported Approach

M. Magyar and I. Majzik
Dept. of Measurement and Information Systems
Budapest University of Technology and Economics
Budapest, Hungary
[mmelinda,majzik]@mit.bme.hu

Abstract—Model based dependability analysis can be used to evaluate the effects of architectural choices on system level availability and reliability. In component based systems the dependability model is built typically from sub-models that are assigned to components or subsystems and represent the local fault occurrences and error propagation. We describe the design and application of a tool that is able to construct the system level dependability model from these sub-models automatically, on the basis of the architecture model of the system. In our tool currently UML architecture model and Stochastic Activity Networks dependability model formalism are supported.

Keywords—dependability evaluation; UML based modeling; modular model construction; Stochastic Activity Networks

I. INTRODUCTION

Evaluation of the effects of architectural choices on the system level reliability, availability and safety is a typical analysis task in the architecture design phase of dependable systems. The proper selection of components, the degree of redundancy, and the applied architectural fault tolerance techniques can be evaluated effectively by applying model based dependability analysis. In this case a system level dependability model is constructed that represents fault occurrence, error propagation, and error handling mechanisms.

In component based systems several components (e.g., hardware devices, software modules) and relations can be characterized from dependability point of view in a similar way: although their parameters may be different, the fault occurrence and error propagation processes are similar. The components and relations of the same type can be assigned dependability sub-models that represent the local fault occurrences and error propagation. Similarly, specific components responsible for redundancy management, error detection, error handling can be assigned (non-trivial) sub-models. The system level dependability model is constructed from these sub-models in a modular way, on the basis of the architecture design as introduced in [3].

Model based design allows a tool-supported construction of the system level dependability model: the system architecture is available in a machine-readable model format, and a tool can implement the systematic procedure of composing the sub-models on the basis of predefined composition rules. This tool can be integrated into the design process in the following way: Dependability experts construct re-usable sub-models (belonging to components, relations and mechanisms used in a specific domain) together with the interfaces of their composition. In the case of a specific design the system architects focus on the architecture, supported by the experts regarding the selection of the dependability mechanisms, and their (joint) decisions are evaluated by (1) constructing the system level dependability model using the automated tool and (2) invoking the solution of the model. The results of this evaluation may show that the selected components and mechanisms satisfy the dependability requirements; otherwise sensitivity analysis can be performed to identify the bottlenecks and weak points that need re-design.

In this paper we describe our Dependability Model Construction Tool that supports this modular modeling approach and is able to build the system level dependability model from the sub-models automatically, on the basis of the architecture model.

II. THE MODEL FORMALISMS

In the tool currently UML class (object) diagrams are supported as input models and Stochastic Activity Networks (SAN [1]) is the formalism of the dependability model. In UML, the components are the classes (objects) and the relations between them are the associations (links). The UML model needs to be extended by stereotypes (identifying the types of the components and relations from dependability point of view) and related parameters. These domain-specific extensions are defined in a UML profile. The UML profile and the constructed SAN sub-models are consistent in a sense that each stereotype identifies a dependability sub-model, and the parameters can be used to instantiate this sub-model in the system level dependability model. For example, the type of a component can be a software task, while the parameters can be fault occurrence rate and error detection delay.

SAN was selected as the formalism of the dependability model and the sub-models since it allows specifying complex conditions and actions, general time distributions and reward functions. The editor of Möbius [2] is used to construct the sub-models of the components, modeling their internal fault occurrences and error detection capabilities. Interface places are used to model specific failure states of the components which can be propagated to other
components. The generated system level dependability model constructed by the tool can be solved using the solvers of Möbius. The solution of the dependability model depends on the firing distributions of the used activities. Simulation of the model is always an option, but several analytical or numerical solutions are also offered by Möbius if the firing distributions are exponential or deterministic.

III. THE ARCHITECTURE OF THE TOOL

The tool implements a model transformation in two steps. First an intermediate model (IM) is constructed which is the abstract dependability model in the form of a hypergraph with typed nodes and edges. The nodes represent system components while the edges represent error propagation dependencies. In the second step the SAN output is generated loading the sub-models belonging to IM nodes and edges, instantiating them using the parameters and connecting them through the predefined interface places according to the composition rules. The resulting SAN model is exported in the format of Möbius.

The sub-models used by the transformation are stored in a sub-model library with additional XML based descriptors that specify the interface places and the used variables in order to be able to connect these sub-models to each other and to initialize the parameters of the SAN model. The library is easily extensible: SAN sub-models (created in Möbius), as well as sub-models generated by external tools from fault trees and PNML can be included.

The first version of the tool was a standalone implementation [4]. Now the tool is available as a Java plug-in in the Eclipse IDE [6]. It is integrated with the Eclipse Modeling Framework [7] to (1) build the Ecore metamodels of the input, output and internal models and (2) to generate parts of the model manipulation program code automatically.

The tool is also integrated with the UML2Tools [8] Eclipse plug-in in which both the domain-specific UML profile and the system architecture model can be constructed. In general, however, any UML modeling environment can be used which provides the architecture model in standard XMI format (v2.1). A related feature of the tool is the validation of the architecture diagram on the basis of the applied UML profile.

Fig. 1 presents the architecture of the tool. The transformation is implemented in the UML2SAN module, while the used metamodels are implemented in the IM metamodel and SAN metamodel plug-ins. The thin arrows represent a “depend on” relation, while the thick arrows represent information flow (processing of models).

The architecture of the tool allows supporting other input languages as well. As the composition of sub-models is implemented at the level of the IM, only new import modules towards the IM are required. Additional output syntax can also be supported since the output model is based on the EMF metamodel of SAN.

IV. EXPERIENCES WITH THE TOOL

Our Dependability Model Construction Tool [9] was successfully used recently in the design and evaluation phase of a safety critical system consisting of 29 components and 109 relations [5]. The UML profile contained 6 class and 6 association stereotypes with a maximum of 6 parameters each. The corresponding domain-specific sub-models were constructed that represented the hardware devices, the software tasks and the dependencies among them with different local error detection capabilities. The system level SAN model composed by our tool consisted of approximately 3500 model elements (including more than 300 places, 1100 gates, 400 instantaneous and 150 timed activities with 470 variables used in the firing distributions).

The transformation was relatively fast, the generation of the system level dependability model took a few minutes only. The dependability model was used to calculate system reliability and the rate of undetected errors (for the computation of the tolerable hazard rate). Due to the non-exponential firing time distributions the model was solved by simulation. In general, the length of the simulation depends on the confidence interval and the defined parameters, our simulation took about 20 minutes. Based on the possibility to change the model parameters in Möbius, we performed sensitivity analyses to investigate the effect of changes in parameters like the period of on-line self tests and coverage of alternative error detection techniques.

REFERENCES