Proceedings
of the 4th International Modelica Conference,
Hamburg, March 7-8, 2005,
Gerhard Schmitz (editor)

B. Johansson, P. Krus
Linköping University, Sweden
Probabilistic Analysis and Design Optimization of Modelica Models
pp. 247-254

Paper presented at the 4th International Modelica Conference, March 7-8, 2005,
Hamburg University of Technology, Hamburg-Harburg, Germany,
organized by The Modelica Association and the Department of Thermodynamics, Hamburg University of Technology

Probabilistic Analysis and Design Optimization of Modelica Models

Björn Johansson Petter Krus
Department of Mechanical Engineering
Linköping University, Sweden

Abstract

In this paper, the system simulation model is discussed from an engineering design perspective. Special emphasis will be given Modelica models, and it is exemplified how computational design methods operate on the simulation model in order to evaluate different concepts. Model based design optimization and probabilistic analysis are discussed as examples of such computational methods.

An XML-based information system for representation and management of design data for use together with the Modelica model is further proposed in order to simplify the use of computational design methods.

Finally, an example is presented, where probabilistic analysis is carried out on a Modelica model of an aircraft actuation system using the proposed and implemented tools and methods.

1 Introduction

In the area of engineering design, a substantial part of the process consists of manual design work involving the inspiration and creativity of the designer. However, a large part of the design process can be formalized, and by applying formal design methods, these can be implemented in computer software as *computational design methods*. By employing computational methods in early stages of the design process, it is possible to acquire valuable information. Such methods could for example include *model based design optimization* or *probabilistic analysis*. These computational methods will be described in more detail throughout the paper, but common for the methods is that they operate on *simulation models* in an automatic, iterative way. This implies new requirements on the simulation tools as well as on the representation and management of data related to the computational methods.

2 Computational design methods

As indicated in the introduction, a computational design method the uses the simulation model as the primary source of information.

The principal similarities between different computational design methods and how they operate on the simulation model are illustrated in Figure 1. With this view, the computational methods either operate on the inputs to the model (design synthesis), or on the outputs from the model (design evaluation). Both probabilistic analysis and design optimization can be seen as automatic methods that repeatedly execute and evaluate the simulation model.

This way of automatic execution adds specific demands to the simulation environment. From the design perspective, it is not of interest exactly how the model is executed, but it must be valid and must not ‘fail’ or get ‘stuck’. It also calls for separation between the actual simulation model and information related to perform a design task using computational

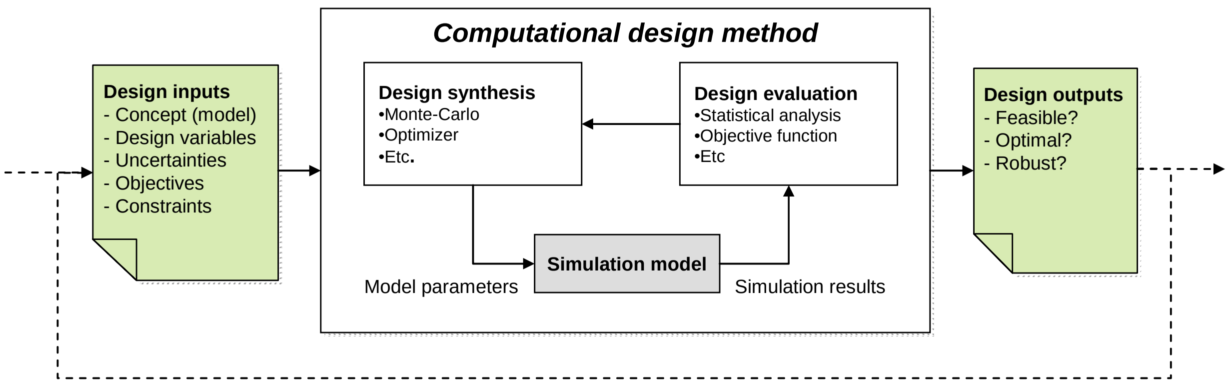

Figure 1. Computational design methods operating on a system simulation model.

methods. This is because the same simulation model could be used in a wide range of design tasks.

2.1 Model based design optimization

A typical example of a computational design method is *design optimization based on system simulation*, as described by Krus et al. [4].

By formulating requirements and desirables as a mathematical objective function, design optimization can be employed. Parameterized simulation models of the system enable an optimization algorithm to be used to find the system parameters that maximize the objective function while meeting the constraints. The optimization algorithm repeatedly modifies specific design variables (model parameters), executes the model, and evaluates an objective function, see Figure 2.

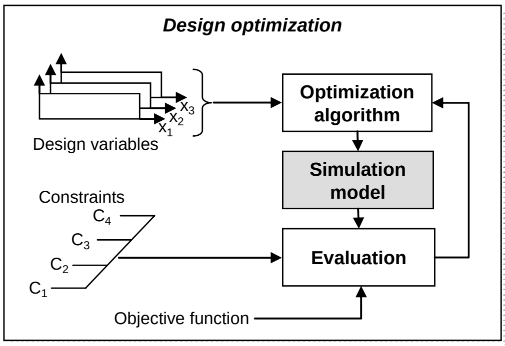

Figure 2. Process for model based design optimization.

A non-gradient method is specifically appropriate for optimization of simulation models since the objective function is defined from simulation results and derivatives of the objective function can not be defined. One example is the *Complex* optimization algorithm, presented by Box [6], which has been used very successfully over a wide range of problems and is characterized by simplicity and robustness.

2.2 Model based probabilistic analysis

Other important examples of computational design methods are based on *probabilistic analysis*. These methods are used not only to assure a technically feasible concept, but also to find a robust design point by including uncertainty in the models.

In all stages of the engineering design process, and especially in early stages, most available information suffers from uncertainty. By using methods for probabilistic analysis, this uncertainty is brought into the design process through the use of simulation models. This is highly desired since important knowledge about the uncertainty is otherwise omitted.

For example, by taking uncertainty into account, the following information can be extracted:

- The probability of meeting a set of constraints and achieving a technically feasible design with in the ranges of the design variables, the *probability of feasibility*.
- How much it will be necessary to relax a specific constraint in order to have a sufficiently high probability of feasibility.
- The effect of uncertainty in system parameter values, i.e. the robustness of the design

The information above can not be achieved using deterministic simulation models with fixed parameter values. Therefore, it is necessary to use *probability distributions* to represent uncertain values on model parameters.

A *feasible design* is defined as a design that satisfies all imposed technical constraints [5]. The examination of the concept's feasibility could be seen as a probabilistic methodology where the probability of finding feasible design alternatives within the design space is investigated. This so-called probability of feasibility, P_{feas}, is an important figure of merit in the early phases of design since it indicates whether the concept is promising for further analysis such as design optimization.

Figure 3 illustrates the process of *concept feasibility assessment*. By assigning normal distributions for the design variables and using a sampling-based method such as the Monte Carlo simulation together with the simulation model, the P_{feas} can be calculated given the settings of the design variables and the constraints.

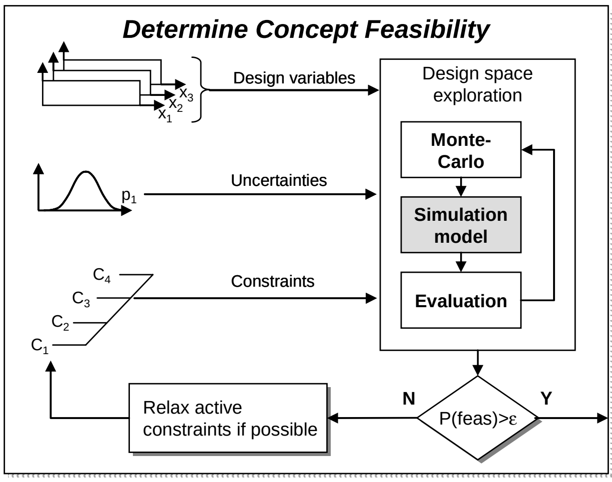

Figure 3. The process of concept feasibility assessment [5]. The model code is evaluated repeatedly where the design variables are varied within the design range using a sampling based method such as Monte-Carlo simulation.

If the total probability of feasibility is too low, the constraints must be investigated individually and either the active constraints relaxed or the concept modified, for example by infusing new technologies to the concept and thereby improving its characteristics. Mathematically, the probability of feasibility P_{feas} for a system with m constraints is defined as [5]:

$$P_{feas} = \prod_{i=1}^{m} P_i \quad (1)$$

$$P_i = P(C_i \leq 0) \quad (2)$$

where P_i is the probability that one specific constraint C_i is met. For another formulation using information content as the figure of merit, see the theory of Axiomatic Design [8]

The Monte-Carlo simulation used to simulate uncertainty or variability is a rather simple algorithm that randomly samples values according to a probability distribution. However, more sophisticated methods with improved search efficiency can be used as well such as Adaptive Importance Sampling (AIS) as described by Wu in [11].

2.3 Computational design data

As indicated in the previous sections, computational design methods include a wide range of data that is not primarily associated the model of the system. As can be concluded from Figure 2 and Figure 3, a wide range of *design related* data is required such as

- *Design variables* – A subset of the system parameters that are modified during the design iteration.
- *Uncertainties* – Many model parameters are uncertain, which must be handled.
- *Constraints* – Measures that must be met in order for the design to be feasible.
- *Objective functions* – A mathematical function used by an optimization algorithm in order to define a figure of merit.
- *Process model* – In order to accomplish full system simulation and optimization involving several types of models and codes, it is necessary to be able to represent and execute a computational sequence.

The data above is normally not possible to represent inside simulation models. It is also the fact that a computational design task often includes more than one model represented using one specific approach. In order to accomplish for example system optimization, it is often necessary to include several types of models, such as CAD, CFD, financial models, etc. Typical is also that integration of already existing, so-called legacy codes is necessary.

3 Modelica and computational data

The Modelica modelling language is developed in an international effort by the Modelica Association [6] consisting of members from both industry and the academic world with the intention of establishing a de-facto standard for system simulation. The Modelica language contains a large number of features with extensive support for advanced modelling of systems from different engineering domains. The modelling principle is object-oriented and equation based where different types of equations are supported. Modelica also enables representation of general data as so-called annotations.

It has been shown several times that Modelica is very well suited for modelling of physical systems. However, representation of design related data as exemplified in previous section is not directly supported. Even if it would be possible to represent design data as annotations this is not an attractive solution since it still not would be generally supported in tools available for Modelica.

One important argument why a separate representation of design data would be necessary is:

A design project often contains several models, and several types of models. In order to fully assess the properties of a certain design, this could include both technical domains and others, such as financial models. A general representation of design data that is simple to use together with different model implementations is therefore necessary.

The approach taken in this work is to represent the data as XML outside the simulation model as illustrated in Figure 4. This approach will be further described in the next section of this paper.

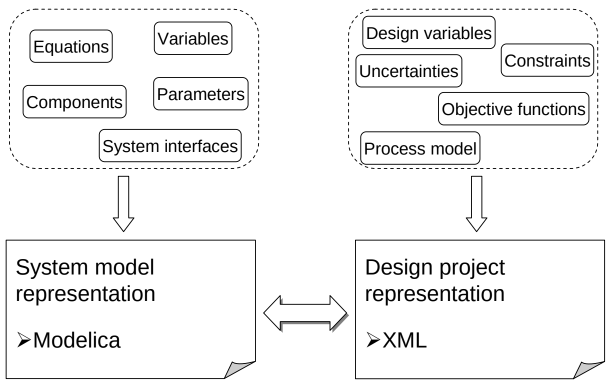

Figure 4. The system simulation model is represented in Modelica, while data regarding the design task is represented in XML.

3.1 XML-based data repository

In order to facilitate the use of computational design methods using models implemented for example as Modelica, a design data repository has been created where the system data can be represented in a general way using XML. An XML document is however not very usable without an accompanying XML schema [10]. Just as the XML can effectively describe data, the XML schema defines the structure of the XML document. It defines each allowed element in a document, the allowed attributes and possibly the acceptable attribute values for each element. It also defines the occurrences, sequence, and nesting of each element.

The information model developed for this purpose has a hierarchical and object-oriented structure in order to organize the data in a way that is close to the physical system. In order for the information model to be as general as possible, generic elements are defined such as *system*, *subsystem*, *variables* and *native data*. A top level structure of the data can be seen in Figure 5, and the different parts of the data model are described in more detail below.

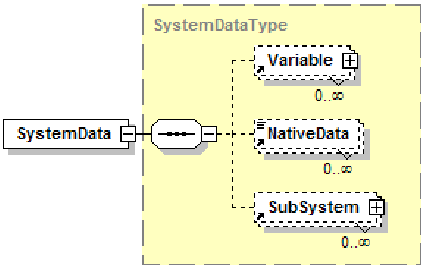

Figure 5. An object oriented and hierarchical structure in order to organize the design data.

The *variable* element is the important building block in the repository. This element is used as a neutral representation of both system parameters and design variables, see Figure 6. Besides name and default value, which are required attributes, the variable contains optional information such as unit, description, and data type. With a *variable type* attribute, it is also possible to define whether the variable is controllable, non controllable, or a so-called technology factor (described in more detail in [3]). As illustrated in Figure 6, the variable element also has sub-elements that contain additional information such as probability distribution and settings if the variable is generated by a design algorithm such as Design of Experiment (DOE) or is a design variable in an optimization algorithm. It is possible to attach these sub-elements to all variables in a generic way.

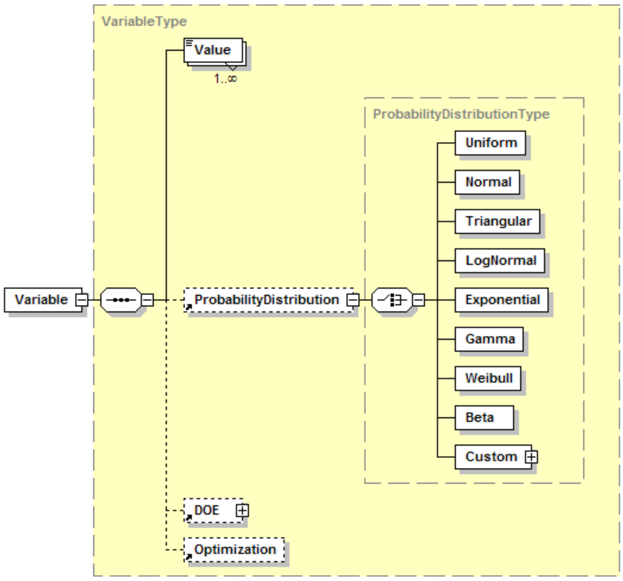

Figure 6. XML Schema representation of the variable element used to represent various kinds of system parameters with extensive information such as probability distributions.

The idea is that probability distributions are defined and stored parametrically. It is possible to select from typical standard distributions such as uniform distribution, normal distribution, triangular distribution, etc. Custom distributions could also be defined as interval values or single values. This means that no mathematical functions for the distributions are stored in the repository. For example, in the case of a normal distribution, the mean value and the standard deviation are stored and not the mathematical function describing the relation between these metrics and the probability density function, PDF.

In Figure 7, some example XML code is visualized as represented using the XML editor XML Spy. For visualization of the actual XML code, see the example in section 5.

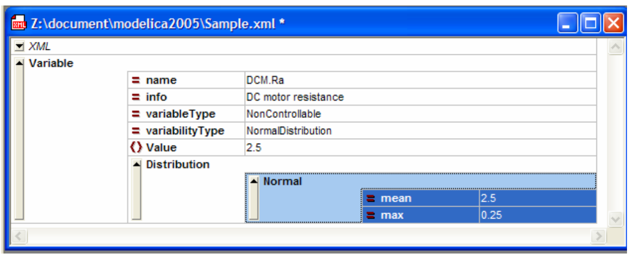

Figure 7. Design variable visualized in the XML editor *XMLSpy*.

4 Integration framework

A software prototype for collaborative system simulation and computational design has been developed in projects prior to the work presented in this paper; see for example [2].

The framework is based on a *Service Oriented Architecture* [7] which means that models and methods communicate using so-called *web service* standards such as SOAP and WSDL, see [9]. The standards are used to define interfaces between the models and to represent the data being exchanged between the models, methods and users clients. The framework enables different kinds of models to be encapsulated as simulation modules without exposing the actual content of the model. Only a published interface is visible to the outside. The models can also be executed in a distributed fashion which enables models and methods to be executed from their original location. With this approach, both models and methods are managed as generic simulation modules which are integrated and executed as illustrated in Figure 8.

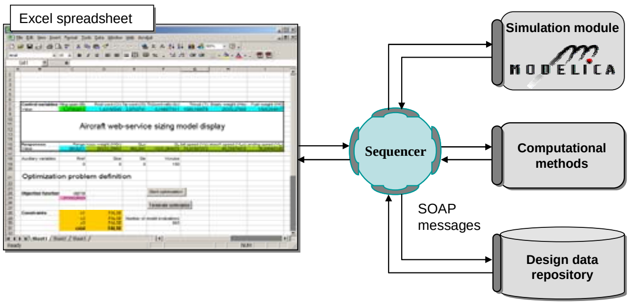

Figure 8. Integration framework where a simulation model implemented in Modelica is integrated with computational methods and a design data repository. Inputs and outputs are here managed using an Excel spreadsheet.

A wrapper is created around the simulation model in order to publish the model as a simulation module as illustrated in Figure 9.

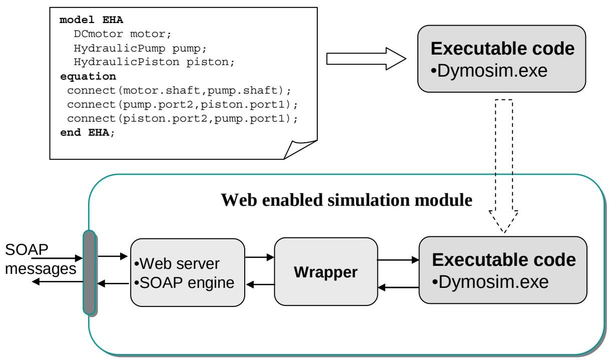

Figure 9. The Modelica system model is translated and compiled using Dymola. The executable code is wrapped as web service simulation module.

In the work presented here, a prototype has been implemented where Matlab constitutes the wrapper that communicates with both the simulation model, and the web service interface. A more permanent solution is however intended where XML technology is used to dynamically create and parse the input and output files to and from the Modelica simulation directly. This is a very flexible approach which has been implemented in previous projects, see [3].

Important to note is that this for model integration is not intended for high-speed data exchange between tightly coupled models. Rather, it is intended for automation of sequential (or parallel) computational design tasks involving several distributed model. An XML-based process model has also been developed which can be automatically executed by a so-called sequencer. Further details about this framework are presented in [2].

5 Example – Probabilistic analysis of aircraft actuation system

In this section an example will be presented where a probabilistic analysis is carried out using the presented framework and a simulation model developed in Modelica.

5.1 Electro-hydrostatic actuation system

The system is an electro-hydraulic system, principally illustrated in Figure 10. The intention is to mount the system inside the aircraft wing in order to move the control surfaces of the aircraft.

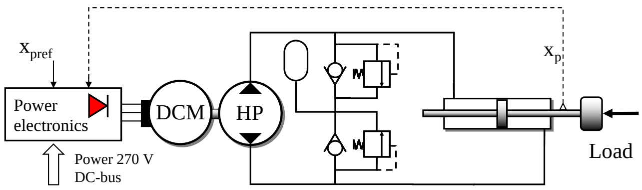

Figure 10. A schematic model of an electro-hydrostatic Actuation system (EHA) implemented in Modelica.

Due to the compact design of the system and the high power density, the system generates heat that can lead to high temperatures and cause damage to the system. It is therefore of interest to analyze the thermal behaviour of the system during missions of the aircraft. In order to accomplish this, a model of both the dynamic performance and the thermal properties of the EHA as well as load forces from authentic missions have been modelled in the Modelica language.

5.2 Simulation model in Modelica

There are different aspects that are of interest when studying actuation systems such as dynamic performance, how the system responds to a control signal, or how sensitive the system is to disturbances.

The models of the system were designed in an object-oriented way where all the components were modelled using the Modelica modelling language. In each component, equations for both dynamic behaviour and thermal properties are included and thermal properties such as temperature and heat flow are represented in the connectors.

The electric motor and the power electronics are also designed to include dynamic as well as thermal properties. Both hydraulic and electric components have equations for thermal properties. Pure thermal components have also been added to the model. In Figure 11, a graphical representation of the model as implemented in Dymola is visualized.

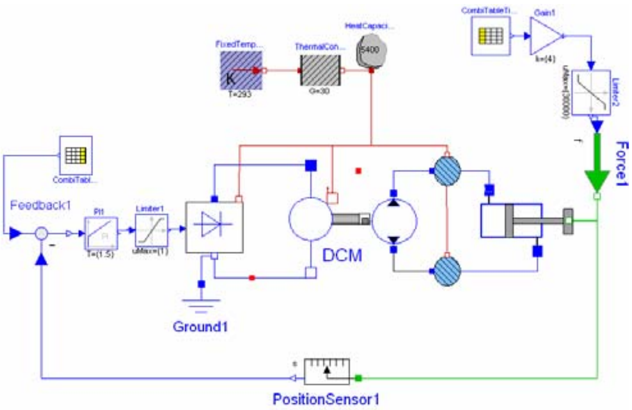

Figure 11. The simulation model as implemented in the Dymola simulation tool.

The system has been simulated in mission of 50 minutes. In Figure 12, results from simulation can be seen. The system was simulated with load and control signals from authentic mission data. The simulation show that high temperatures will occur both in the hydraulic fluid as well as in the motor windings during a so-called extreme mission.

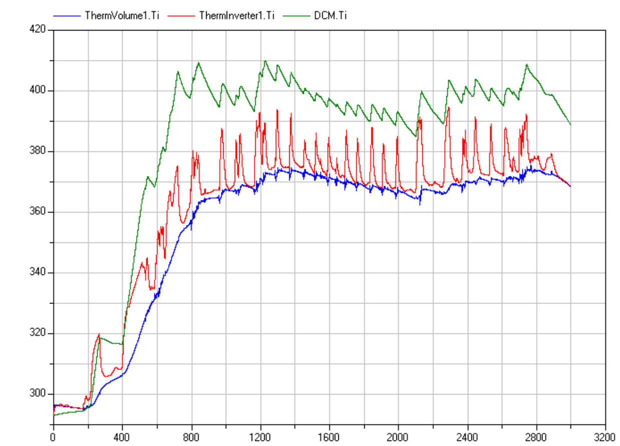

Figure 12. Temperatures [K] in the system during a heavy mission. Simulation of the Modelica model using Dymola. The mission is simulated for 50 minutes (3000 sec).

5.3 The uncertainties

From a design point of view, the system includes several uncertain parameters that could affect the thermal properties in the components. In order to keep the example simple, only three parameters in the model is selected to illustrate uncertainties in the system.

Normal distributions are selected for the resistance in the DC motor and in the power electronics. A normal distribution is also set for at speed dependent thermal parameter in the motor.

Table 1. Definition of uncertain parameters.

System parameter	Mean value	Standard dev
Inverter resistance [Ω]	0.35	0.1
Speed dependent thermal constant [rad^{-1}]	0.5	0.1
Motor resistance [Ω]	2.5	0.25

As an example, the representation of the motor resistance is visualized below. Both graphically, and as XML code.

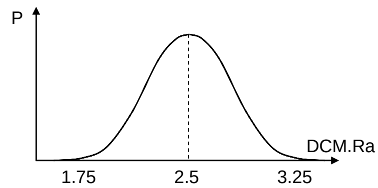

Figure 13. A normal probability distribution defines the resistance of the electric DC motor.

```
<Variable name="DCM.Ra" info="DC motor resistance"
          variableType="NonControllable"
          variabilityType="NormalDistribution">
   <Value>2.5</Value>
   <Distribution>
         <Normal mean="2.5" stdDev="0.25"/>
   </Distribution>
</Variable>
```

5.4 The constraints

A few example constraints are here presented regarding the temperatures in different parts of the system.

- The temperature of the hydraulic oil should not exceed 90°C,
  - $C_1 = \text{Oil.Ti} \leq 90°\text{C}$
- The temperature of the DC motor windings should not exceed 100°C
  - $C_2 = \text{DCM.Ti} \leq 100°\text{C}$

The constraints are evaluated in each simulation in order to evaluate the probability of feasibility described below.

5.5 Evaluating probability of feasibility

In the application example, probabilistic analysis is employed on the system in order to investigate the probabilities of meeting the constraints.

The framework illustrated in Figure 8 is here used for the simulations. The simulations are controlled from an Excel document, where inputs to the model can be entered as well as results from the model monitored.

In each execution of the model, the max temperature in the different parts of the system at each simulation is stored. By modifying the inputs according to the probability distributions of the uncertain parameters, variability in the responses is obtained as well.

The results are investigated by computing a Cumulative Density Function (CDF) for the response of interest. By fitting a standard distribution to the values of the responses, the probability of achieving responses that meet the constraints can be computed, see Figure 14.

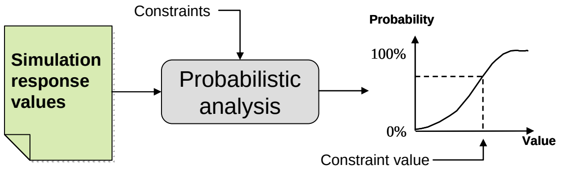

Figure 14. The simulation results are extracted from the XML repository for analysis.

Below, the results for the temperatures of the hydraulic fluid as well as the DC motor temperature are visualized.

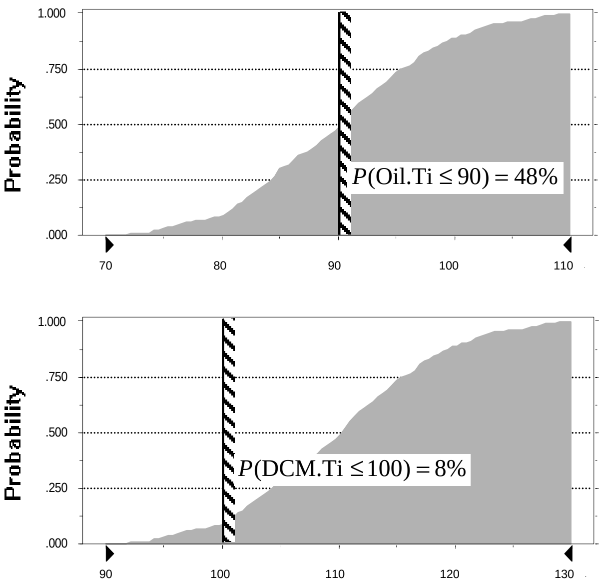

Figure 15. The probability of meeting constraints on oil temperature and DC motor temperature with uncertainty in some system parameters.

For the uncertainties and constraints used in this example, the results are the following probabilities:

- The temperature of the hydraulic oil should not exceed 90°C,
  - $P(\mathrm{C}_1) = 48\%$
- The temperature of the DC motor windings should not exceed 100°C
  - $P(\mathrm{C}_2) = 8\%$

This implies that the total probability of meeting the constraints (probability of feasibility) is:

$$P(\text{feas}) = P\{(\text{Oil.Ti} \le 90) \cup (\text{DCM.Ti} \le 100)\}$$
$$= 4\%$$

It is obvious that this is too low probability for the system to be robust and we must investigate if the constraint can be relaxed or else we make some change to our design. For the purpose of this example, we now assume that the constraints cannot be relaxed.

Now assume that we infuse technologies to our concept that increases the ventilation of the EHA mounting area and the increases the transportation of heat from the EHA surface. This means that we can assume a *technology factor* that should affect the probability of meeting the constraints.

By modifying our model we can now re-evaluate the probabilistic analysis in the same way as above.

The results in Figure 16 show that the probability of meeting the constraints has increased.

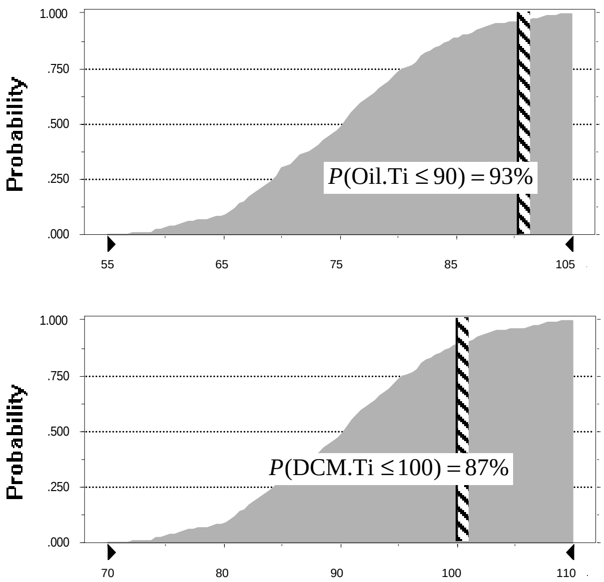

Figure 16. The probability of meeting constraints on oil temperature and DC motor temperature with a modified concept.

The total probability if feasibility for the modified concept is now:

$$P(\text{feas}) = P\{(\text{Oil.Ti} \leq 90) \bigcup (\text{DCM.Ti} \leq 100)\}$$
$$= 81\%$$

We can now accept the current concept and move on to the next step in the design process, which includes further simulation and optimization to achieve an optimal design point with respect to both performance and robustness. This is however beyond the scope of this paper.

6 Discussion and conclusions

It is important to realize that in a computational design task, the system simulation model is not the top-level integrator that accesses and integrates different types of data. It is rather a component that is *being accessed* from a design framework at a higher level including some computational method. The information that the simulation model delivers is then evaluated and integrated with results from several types of models.

Simulation models in industry exists in a wide range of representations ranging from old legacy codes represented in Fortran code to modern object-oriented modelling languages such as the Modelica language implemented in simulation tools such as Dymola. It is important that the computational design methods can interact with the models regardless of implementation. With a design data repository implemented in a format that is simple to access by a wide range of tools, this interaction is highly facilitated.

The approach presented in this paper uses XML for representation of the design data in a format that is general and not associated with existing representations of system simulation models. The advantage is that XML is widely supported by a wide range of software tools, and that it is simple to access and manage the XML data.

The example presented in this paper is only one simple illustration of how the simulation model can be used in a computational design task. Increased demands for the product developing industry regarding faster time to market will make design automation more and more important. It is therefore very important to continue to define interfaces between the area of engineering design and the area of system modelling and simulation.

References

[1] BOX M. J., “A new method of constrained optimization and a comparison with other methods,” *Computer Journal*, 8:42-52, 1965.

[2] JOHANSSON B. AND KRUS P., “A Web Service Approach for Model Integration in Computational Design”, in *Proceedings of ASME Computers and Information in Engineering Conference*, Chicago, USA, September 2-6, 2003.

[3] JOHANSSON B., DELAURENTIS D. A. AND MAVRIS D. N., “Managing Design Data for Probabilistic Evaluation of Aircraft Concepts”, in *Proceedings of International Conference on Engineering Design*, ICED'03, Stockholm, Sweden, August 19-21, 2003.

[4] KRUS P., JANSSON A. AND PALMBERG J-O, “Optimisation for Concept Selection in Hydraulic Systems”, in *Proceedings of 4th Bath International Fluid Power Workshop*, Bath, UK, 1991.

[5] MAVRIS D. N. AND DELAURENTIS D. A., “A probabilistic approach for examining aircraft concept feasibility and viability”, *Aircraft_Design*, 3 pp.79-101, 2000.

[6] MODELICA ASSOCIATION, “Modelica A Unified Object-Oriented Language for Physical Systems Modeling, Language Specification, http://www.modelica.org, 1999.

[7] PAPAZOGLOU, M.P.: “Service-Oriented Computing: Concepts, Characteristics and Directions” in Proceedings of the 4th International Conference on Web Information Systems Engineering (WISE 2003), December 10-12, 2003

[8] SUH N. P., *Axiomatic Design: Advances and Applications*, Oxford University Press, New York; Oxford, 2001.

[9] TSALGATIDOU A. AND PILIOURA T., "An Overview of Standards and Related Technology in Web Services," *Distributed and Parallel Databases*, 12, pp. 135-162, 2002.

[10] WORLD WIDE WEB CONSORTIUM, http://www.w3c.org.

[11] WU Y.-T., “Computational methods for efficient structural reliability and reliability sensistivity analysis” *AIAA Journal*, Vol. 32(8), pp 1717-23, 1994.