COMPLEX SYSTEMS DESIGN AUTOMATION IN THE PRESENCE OF BOUNDED AND STATISTICAL UNCERTAINTIES

Horst Salzwedel, TU Ilmenau

Abstract

Complex technical and organizational systems are designed by groups of people departments and/or companies. Because of product uncertainties during development, each of the individual engineers make individual assumptions in their employed design methodologies. Because of the complexity of the system they do not understand the impact of their decisions on the overall systems. Since current design approaches do not consider the impact of this uncertainty on the integrated system, the design specifications are error prone and not validated[1]. The likelihood that a system design does not have critical errors is less than 4%[2].

In this paper design methodologies of individual engineers are modeled and connected by a design process graph to make them executable. Bounded and statistical uncertainties of early design stages are included and bounds on integrated designs based on uncertainties at different design stages are determined using MLDesigner 3.0 [9] Simulation Set capabilities. The new design methodology permits treating uncertainties in early design stages, validate design specifications, and optimize the design flow for different criteria, like quality in design and design cost and verify systems in the presence of this uncertainty.

Introduction

Complex systems like networked systems in vehicles (aircraft, spacecraft, automobiles, ships, trains, autonomous systems), between vehicles, IT systems, communication or organizational systems can only be developed by groups of people, departments and/or companies. To meet time to market requirements, the system design is partitioned into subsystems and subsystem design is distributed to specialist teams before uncertainties about individual subsystems and interactions between subsystems has been resolved. The individual engineers make assumptions, many based on experience with less complex systems. It is not possible for these engineers to understand the impact of their decisions on these systems. Since the models passed on for system integration do not include the model uncertainties, the uncertainty of the integrated system is not determined. The integrated system model or design specification is therefore not validated. The likelihood that this design approach will work without critical problems is less than 4% [2]. Very time consuming and expensive integration processes try to resolve conflicts and critical performance issues.

Figure 1 [3] depicts the probability of critical problems as a function of phases of the design process. The probability of critical problems is very high in early design stages and is low in late design stages. These failure rates often result in huge cost overruns for correcting mistakes in complex systems design. Several billion $/€ projects were terminated due to these failures.

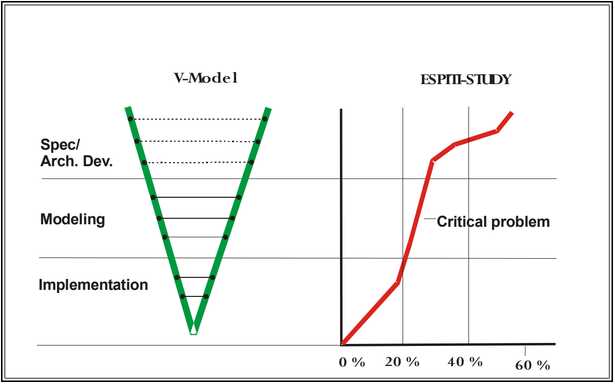

Figure 1: Critical problems in design of complex systems [3]

The analysis of designs shows that most causes for critical design flaws come from poor design specifications. The main reason is that most specifications are on paper only and therefore cannot be validated. The introduction of executable specifications [4] has improved the quality of specifications and significantly reduced the number of design iterations and the cost of design of electronics, software and organizational systems [5]. However, for complex systems, like large IT systems, complex systems design flows and aerospace systems the likelihood that the first design will be right is very low. In addition, in silicon systems, cost for organizational processes for design and development is increasing exponentially and is already more than 30% for complex chip designs [6].

Development of executable development and organizational process models can catch critical problems in development processes [7] or result in significant improvements and cost savings in organizational processes [8]. However, in most projects these available technologies are not yet employed.

Even so uncertainty plays a major role in early design stages, it is only treated at a subcomponent level and not at the system level. The reason is that as the number of designers/design teams increases, the number of design alternatives increases. For example, 2 different design points of each designer in a 10-person design team, where each team member generates two alternatives, results in 2^{10} different designs. Current manual integration methods cannot cope with this size of design space exploration and verification.

In this paper a system design methodology is developed based on executable design flows. Bounded and statistical uncertainties of early design stages are included and bounds on integrated designs based on uncertainties at different design stages are determined. Examples will demonstrate the new design methodology.

The new design methodology not only permits to treat uncertainties in early design stages, but permit optimizing the design, as well as the design flow for different criteria like quality in design and design cost. Design changes can easily be implemented and design tracking automated.

Product Uncertainty

Information, knowledge and uncertainty about a product changes significantly during product life time. This knowledge about the product includes knowledge about the product itself, knowledge about the intended use, and knowledge about the environment in which the product is to be used. This uncertainty is bounded and may sometimes be described by bounded parameter sets, bounded statistical distributions, and/or bounded functional behavior described by e.g. H^{∞} norms for continuous systems.

Figure 2 shows a typical behavior of the mean of a product uncertainty. At the start of a development project the product uncertainty, U_0, is a function on whether the development team has developed similar type of products before. During the development phase the knowledge about the product increases with the learning rate, l, reducing product uncertainty. This learning curve can be described by an exponential function,

$$U_D(t) = \begin{Bmatrix} U_0 e^{-lt} & \forall \ t < t_D \\ U_0 e^{-it_D} & \forall \ t \geq t_D \end{Bmatrix},$$

where

$$l = \begin{Bmatrix} l_P & for & 0 \leq t < t_P & on\ paper \\ l_M & for & t_P \leq t < t_M & model\ based \end{Bmatrix}$$

for initial slow development on paper and rapid development with executable models, respectively.

At the end of the development phase, the product is delivered to customers, who will find additional insufficiencies that are reported to support and corrected by the development team. The learning effect due to customer testing and support can be described by,

$$L_S(t) = \begin{Bmatrix} 0 & \forall \ t \leq t_D \\ L_{S0}(1 - e^{-l(t-t_D)}) & \forall \ t > t_D \end{Bmatrix}$$

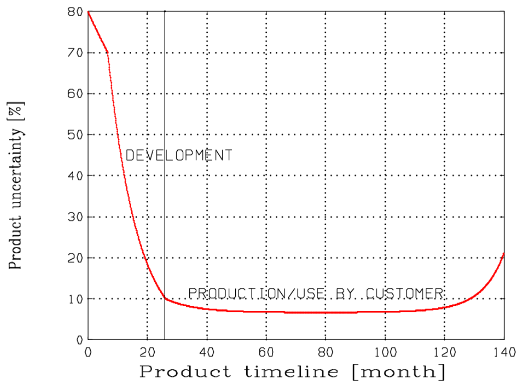

Figure 2: Product uncertainty

However information about a product is also lost during lifetime of a product. The reasons are that people leave development teams, are reassigned, or just not remembering all the decisions which went into the design of a product. Additionally, as time progresses, products are often used for applications that were not considered during the development phase. These effects impose additional uncertainty on the product, which can be approximated by,

$$U_L = U_{L0} e^{Lt^2}$$

As development of products with long life times like the Space Shuttle showed, people even have to be called back from retirement in order to overcome this loss of product information.

The uncertainty about a product can then be expressed by,

$$U_P(t) = U_D(t) + U_L(t) - L_S(t)$$

Comparing Figure 1 with Figure 2, we observe that observed occurrence of critical design errors and product uncertainty are highly correlated. We may conclude that critical design errors are primarily due to information uncertainty about a product and design processes that do not consider this uncertainty in the right way and do not validate and verify the system considering this uncertainty.

In the following sections it is shown how uncertainty in use cases, environment and design may be used to determine the system uncertainty. Making the design process itself executable enables its validation, harmonization, and may result in significant savings in development time and risk.

Complex System Design

Complex systems are always designed and developed by groups of people, departments and/or companies. The system is partitioned into subsystems and subsystems are assigned to different groups. Each group typically has unique knowledge in a particular field and will use their own methodology, models and software for the design of their assigned component. Some will use different software systems.

For design, validation and verification design groups will consider a range of possible parameters and use approximate models for analysis. Different designers/developers will exchange information about their components to other designers according to design or process graphs, Figure 3. The exchanged information typically will not include all assumptions and ranges of uncertainty considered in the design of components. The reason is that some of the uncertainties are very specific to the field of knowledge used for a component and other designers will not understand this information nor will they be able to properly evaluate potential interactions between components in different subsystems. Other reasons include that the variability of a design may be too large to be considered for manual system integration.

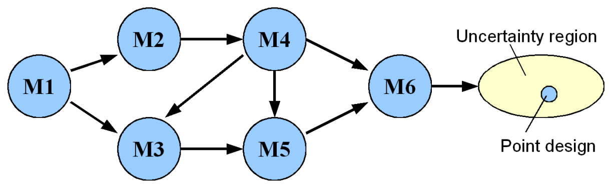

Figure 3: Design process graph

Ten designers doing a conceptual design and each is passing on a minimum and a maximum of one component specific parameter to the integration team, the integration team would require to integrate 2^{10} different system designs for analysis and verification. This would be too time consuming when done manually. It is therefore not done. For system integration, each team will pass on their "best" design to the integration team.

The integrated system is therefore not verified for variability due to uncertainties in subsystems and interactions between components located in different subsystems. As a result, the probability that this coupling is causing critical errors in the design is very high. Testing (expensive and time consuming) should eliminate most critical errors, but customers increasingly find unresolved critical errors after delivery of products. Additionally, different designs based on the same platform design will exhibit very different probability of failures.

Design Automation

To integrate, validate, and verify different designs (e.g. 2^{10}), and find the "best" solution between them is an impossible task if done manually. The design must be automated, in order to overcome this problem. 2^{10} iterations of a simulation are not very unusual, dependent on the complexity of the simulation. With distributed simulations and multi core processors even complex simulations can be run for, e.g., $\geq 2^{10}$ iterations.

For an automated design, a simulation must be developed that meets the following requirement,

1) modules, that are complete independent simulations of design methodologies

2) the simulations may be executed by the same or different simulation tools

3) a design process graph connects the design methodologies

4) Monte Carlo simulation capabilities, in order to be able to analyse sets or ranges of parameters

5) simulation model generation, in order to iterate over different architectures

6) optimization methodologies that can optimize a design with respect to system level objectives

7) distributed simulation to overcome potentially large processing requirements

8) the simulation must be connected to a data base, in order to keep track of the large volume of information

9) models and data must be stored in a standardized way, in order to ease comparison and analysis

Figure 4 depicts such an automated design process simulation. Each designer has to develop an executable model of her/his design methodology. Process engineers design the process design graph that connect the executable design methodologies.

The simulation generates a set of feasible designs and maps component uncertainties into system uncertainties of coupled designs and can determine which designs meet system requirements for all uncertainties in parameters, architecture, missions/use cases, and environment. If the system performance falls within the permissible performances of the design specifications, the design is verified.

The system uncertainty determined by the range of design variations also depicts the level of uncertainty at a given stage of development. As the different members of the design team get up the learning curve, Figure 2, this uncertainty is going to decrease. Unacceptable levels of system uncertainty can be analyzed, causes determined and eliminated.

The automated design process can be connected to a data base and automatically track design changes and their impact on overall system design criteria, including performance, cost, quality of design.

The design process itself can be modeled and optimized. In [7] the development process for a railway switching system was modeled in MLDesigner and optimized for team selection, team load. Critical components of the development process were identified and eliminated that would have doubled the development time. In [10] the design and development process for automotive electronic control units was optimized using genetic algorithms for finding the optimal combination of design methodologies (team selection), performance (timing, BER, cost and quality. Areas for significant improvements were identified.

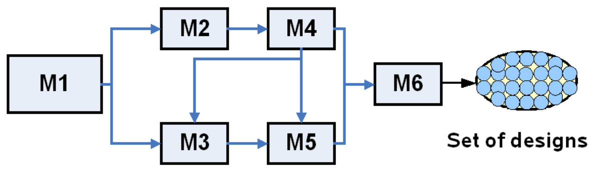

Figure 4: Executable design methodologies, connected by a design process graph

Implementation of Design Automation

The proposed design automation technology is being implemented and analyzed in the multi domain simulator MLDesigner [9], which already includes multiple execution domains, reducing the need of interfaces to other tools. When necessary, MLDesigner provides hooks that make it easy to interface to tools with similar execution domains. The Monte Carlo simulation capabilities of MLDesigner automatically distribute simulations to other computers that are registered to MLDesigner. MLDesigner 2.7 includes an SQL library for easy data base access.

All MLDesigner models are stored in XML, supporting mapping though XSLT scripts. Baumann [11] is developing network building blocks and XSLT based mapping functions to automatically map functional level models into architecture and combined architectural and functional models into implementations. Fischer

[12] is developing system level architecture optimization technology which generates the information about the optimal architecture, that can then be read by the mapping functions of Baumann. Riehmer [13] is developing a standardized XML based data format for MLDesigner and Octave that is compatible with the Open Office Document Standard and can be read and generated by spreadsheets from Open Office and Microsoft Office. Genetic algorithm optimization methods have been developed to optimize MLDesigner models of design processes for performance and quality. These algorithms will be extended for optimization in automated design processes.

The next version of MLDesigner (3.0) will provide new simulation set capabilities which have been developed to make it possible to connect independent simulations, using UML activity diagram syntax that describes control flow, design flow, work flow and/or analysis flow connecting the elements of the simulation set. Parallel and sequential execution will support automated distributed simulation of independent simulations. The design process graph can be defined by the standard graphical model editor of MLDesigner, Figure 5.

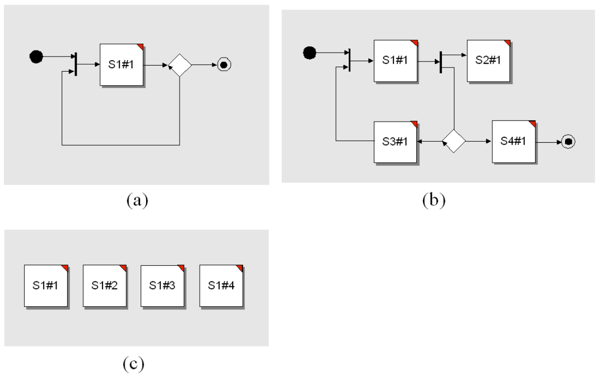

Figure 5: Sequential executions with termination conditions (a,b) and parallel executions (c) of MLDesigner simulation sets

Examples

This sections show two very simple examples using the simulation set capabilities of MLDesigner 3.0 and using two external tools --- SatLab and Octave --- for analysis. These low complexity examples can be verified directly in MLDesigner or by SatLab and Octave.

Example 1: The accuracy of a least squares signal processing algorithm is analyzed for the example of fitting a fourth order polynomial

$$y_i = p_1 + p_2 x_i + p_3 x_i^2 + p_4 x_i^3 + \nu_i, \quad \nu = N(0, R_\nu)$$

through a data set with normally distributed measurement errors. 30 different data sets are generated by MLDesigner and 30 to 1400 measurements are taken of these data for parameter estimation. The parameter estimation is performed by SatLab. Figure 6 shows how the parameter estimation error change as a function of number of measurements taken.

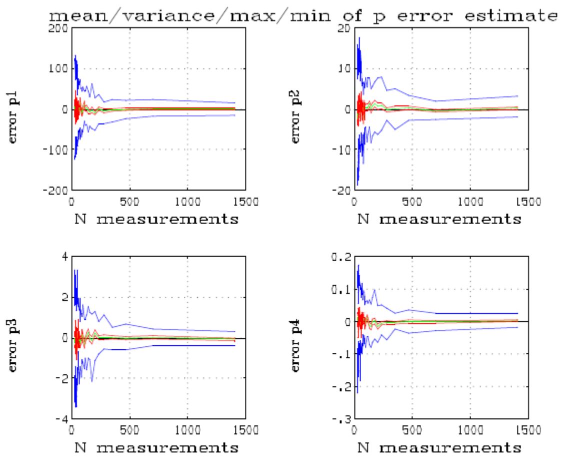

Figure 6: Analysis results for accuracy of least squares parameter estimation algorithms

Example 2: Chapter 10.3 of Reference 14 shows a design methodology for a design specification of a longitudinal controller for an aircraft (Boeing 747). Variations in loading and fuel consumption will affect aircraft dynamics parameters and flight behavior. These parameters changes are bounded by minimum and maximum weight and the permissible minimum and maximum center of mass location for the aircraft. Using maximum and minimum values instead of nominal values for seven longitudinal aircraft dynamics parameters, results in $2^7 = 128$ different dynamic models of the aircraft pitch behavior. Figure 7 shows minimum, mean and maximum behavior of an altitude change of the aircraft are affected by 40% changes in pitch parameters $Xu, Xw, Zu, Zw, Mu, Mw,$ and Mq . The results show that the chosen closed loop eigenvalues for the autopilot design is very robust against parameter changes and passengers will hardly notice the changes in behavior.

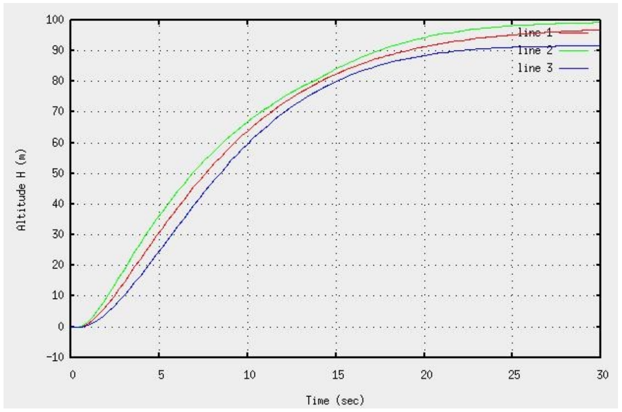

Figure 7: Aircraft altitude hold autopilot response verification for bounded aircraft parameter changes of ± 40 %

Conclusions

Uncertainty of information in design of complex systems is the primary reason for critical problems and expensive corrective actions. A design automation methodology based on coupled models of design methodologies of participating design groups and developers is proposed that overcomes this critical design problem and enables large design space exploration and verification of uncertain systems. The developed methodology permits verification of system level functional and performance behavior in the presence of system uncertainty. Digital as well as analog systems may be verified by this methodology. A methodology for implementing this methodology by the simulation set capabilities of the development tool MLDesigner is shown. Two simple examples show how statistical and bounded system uncertainties may be mapped into bounded system behavior by automating design and analysis using an executable design flow.

REFERENCES

[1] John Hines, We Don't Do Design Correctly!, Keynote speech at IEEE 9th *International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, MASCOTS 2001, Cincinati, Ohio, Aug. 15-18, 2001.

[2] Bruno Schienmann, *Kontinuierliches Anforderungsmanagement*, Addison-Wesley, 2002. – ISBN 3-8273-17-87-8

[3] Horst Salzwedel, Mission Level Design of Avionics, *AIAA-IEEE DASC 04 - The 23rd Digital Avionics Systems Conference 2004*, Salt Lake City, Utah, USA, October 24-28, 2004.

[4] Holger Rath, Gunar Schorcht, Horst Salzwedel, Simulationsumgebung für Bordnetze - Bordnetz-Spezifikationen modellieren und validieren, *Hanser automotive*, 5-6.2006, S. 34-38, Carl Hanser Verlag München, ISSN 1860-5699.

[5] Horst Salzwedel, Frank Richter, Matthias Kühn. Standardized Modeling and Simulation of Hospital Processes - Optimization of Cancer Treatment Center, *International Conference on Health Sciences Simulation, HSS '07,* San Diego, California, January 14-18, 2007.

[6] Ralf Pferdmenges, Presentation on current challenges in chip design, *EDACentrum Workshop: System Planning*, Hannover, Germany, November 30, 2006.

[7] Manuela Tröbs, *Analyse, Optimierung und Simulation des Simis IS Entwicklungsprozesses*, Thesis, Siemens/Technical University Ilmenau, June 2006.

[8] Matthias Kühn, Frank Richter, Horst Salzwedel, Process simulation for significant efficiency gains in clinical departments – a practical example of a cancer clinic, to be presented at *52th International Scientific Colloquium*, Ilmenau, September 10-12, 2007.

[9] MLDesigner® Manual v3.0, http://www.mldesigner.com

[10] Tom Dengler, Host Salzwedel, Optimierung von komplexen Enwicklungsprozessen mittels simulationsgestützter Prozessanalyse, to be presented at *52th International Scientific Colloquium*, Ilmenau, September 10-12, 2007.

[11] Tommy Baumann, Horst Salzwedel, Mapping of Electronic System Level (ESL) Models Into Implementation, *DATE'07*, Acropolis, Nice, France, April 16-20, 2007.

[12] Nils Fischer, *Design of a Plug-and-Play Development Environment for Optimizing Avionics Systems Architectures*, Thesis, Airbus/Technical University Ilmenau, July 2007.

[13] Stefan Riehmer, *Standardizing Data Storage for MLDesigner and Octave using the Open Office Document Standard*, Thesis, Technical University Ilmenau, August 2007.

[14] Gene Franklin, David Powell, Abbas Emami-Naeini, *Feedback Control of Dynamic Systems*, Pearson Prentice Hall, New Jersey 2006.

AUTHOR:

Horst Salzwedel, Technical University Ilmenau, horst.salzwedel@tu-ilmenau.de