49. Internationales Wissenschaftliches Kolloquium
Technische Universität Ilmenau
27.-30. September 2004

Horst Salzwedel

DESIGN TECHNOLOGY DEVELOPMENT TOWARDS MISSION LEVEL DESIGN

ABSTRACT

Complex systems are characterized by architectural complexity, dynamic interaction between subsystems, and complex functionality, understood only by teams from different disciplines. 20 years ago the major challenge was the multidisciplinary design of avionics. Over the past 20 years, design methods and tools have been developed to cope with these challenges. Today the complexity of networked electronics and the interaction of hardware and software impose similar complexity and design challenges. According to Moore's Law, closely followed by industry, the complexity of electronics increases by a factor 100 every 10 years, requiring to increase abstraction in the design methodology, in order to cope with this increase of complexity. This paper shows the move towards performance and mission level design and its advantages over functional level design approaches.

THE CHALLENGE OF COMPLEXITY

Aerospace systems are characterized by architectural complexity, dynamic interaction between subsystems, and complex interdisciplinary functionality. The challenges of designing these complex system with more than 1000 electronic control units (ECUs) include not only the architectural and functional complexity of the avionics systems themselves, but also the complexity of the organizational structure of the design teams from mission specification, design, implementation, test, training, and operation.

The introduction of stability augmentation systems in the 1960th and 1970th coupled different areas of engineering developments and caused interdisciplinary problems in the designs. Every aircraft prototype tested exhibited aero-servo-elasticity problems. The analysis of these problems showed that the main cause for these problems was flawed specifications resulting from insufficient communication between design engineers of different areas, and use of incompatible modeling techniques and tools.

Multidisciplinary research led to modeling and design methodologies that considered engineering expertise and the limit of it for the design flow, e.g., for the development of integrated flight propulsion control systems. Multidisciplinary research sponsored by AFWAL [2] let to the development of generic software tools like Ctrl-C®, MatriX® and their derivatives Matlab® and Octave™, that permitted to combine functional level models from different disciplines to reduce these problems.

An example of successful multidisciplinary modeling can be found in the design of a transfer alignment filter. This filter is responsible for the transfer of navigational information from an aircraft navigation system to that of a missile under a wing. Early developments for this filter considered a rigid body connection between the aircraft and the missile, and treated all other effects as white noise, causing large alignment errors and large alignment times. Large research efforts and tests could not solve the problem. The availability of Ctrl-C permitted to easily combine models from structural dynamics, aerodynamics and flight control into a unified model. This permitted to identify structural flexing as the major problem for filter accuracy and alignment time. The inclusion of these effects improved the accuracy by more than a factor 100 and reduced the alignment time by more than a factor 100 [2].

With the proliferation of electronics in nearly all type of engineering systems, and the rapid increase of the complexity of electronics, the major challenge of system development has become the gap between system design and networked electronics implementation.

In the 1970s chip masks were designed with CAD systems. In the 1980s chip complexity had increased by a factor of 100. Engineers could no longer handle this complexity. Languages like Verilog and VHDL were introduced to design chips at the logical level and translate designs into masks. At the beginning of the 1990s, the complexity had again increased by a factor of 100. Software tools like SPW® and COSSAP® were introduced to raise the abstraction of electronics design from the logical level to the functional level. This permitted e.g. the design of modems in 3 month. The problem remained to translate behavioral models into RTL level models and validation of the resulting RTL level models against design specifications at the behavioral level.

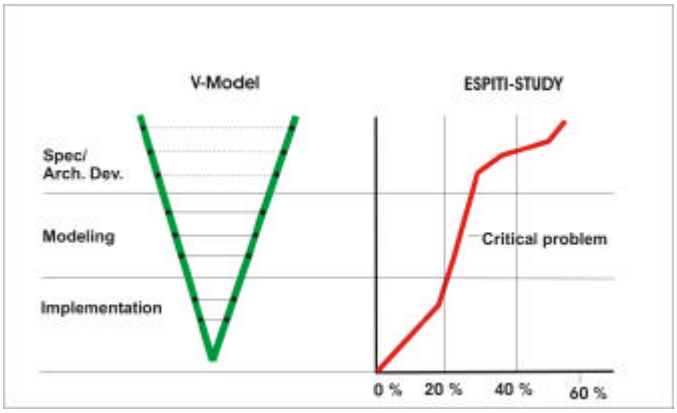
Today (2000s) the complexity of electronics has increased by a factor of 10000 since integrated functional level design tools have been introduced for system design, and by a factor of 100 since design of complex electronic moved from logical level to functional level. Additionally, chips have become systems and complex systems like aircraft, spacecraft, automobiles and communication systems are dominated by networked electronics. This compounds the problem of the gap between design and implementation. This increase in complexity has caused major problems throughout the industry, including,

- Flight control systems failures such as the Saab 39 and Osprey
- The failure of the first Ariane 5 rocket because of a value overflow. The implementation was not tested against the mission
- The development of the Teledesic satellite system was discontinued after it was found that major design specifications had to be revised late in the design
- In 1999 2 spacecraft to Mars failed because of a mix-up between units used by different design teams
- The Boeing 702 series of spacecraft exhibit major problems. The development team cannot predict when the problem will be fixed
- Electronic control units in automobiles exhibit failure rates of up to 3000ppm (required <10ppm)
- Luxury class automobiles have to be recalled because of unwanted interactions between large number of networked electronic subsystems

- The German automated toll collection system has been delayed because very little worked when independently developed subcomponents were put together. The development team cannot even predict when the problems will be solved

All these examples exhibit similar problems in the design flow. The subsystems are designed from written specifications. When they are put together, the system does not work. The problems are fixed on a local level. Despite very high validation and test cost critical problems remain.

A recent ESPITI study shows which stage of a design flow causes critical problems in system design. The probability is nearly 60% that the specifications cause critical problems. The probability for critical problems caused by modeling and design is about 25% and less than 20% by implementation. These critical problems could be in hardware or software, but often in the coupling between them. A major contributor to this problem is that the design is done at the functional level. However, complex systems can no longer be simulated as a whole at functional level. Specifications can therefore not be validated at this level of design abstraction.



MISSION LEVEL DESIGN

For the development of complex planetary and interplanetary space systems, detailed mission analysis precedes the design phase and implementation phase, in order to get validated, executable specifications for the overall system. During the design phase, all components of the design are validated against these specifications. Hardware in the loop test are performed to test the implementation against the design and the mission level specifications. This design approach led to highly reliable systems that roam the solar system up to the outermost planets.

Today components and subcomponents like telecommunication systems, operational infrastructure, embedded systems, and processors have complexities that far exceed those of early spacecraft to the outer planets. Even with our improved computational capabilities, it is no longer possible to simulate the HW/SW implementation of components and subcomponents against the mission requirements. This led to failures like the first Ariane 5 and the 1999 missions to Mars.

MISSION LEVEL DESIGN

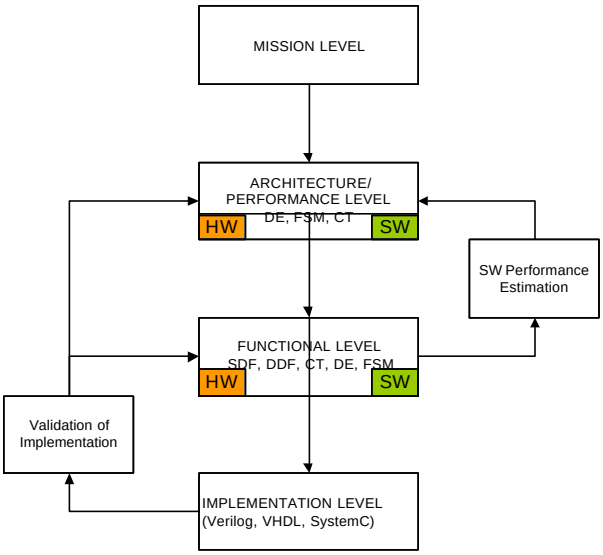
Mission Level Design (MLD) [4] is a hierarchical approach that generalizes the design approach for deep space missions:

1. A validated and executable mission is the behavior of the system that uses a component to be designed
2. Validated specifications of the functional behavior are generated by validating the high level architecture and performance of the component against the mission level requirements

3. The functional behavior of HW and SW is verified and validated separately and in combination against the specifications stemming from the architectural/performance model

Experience shows that 80-90% of the design decisions can be made at stage 2 of this approach.

In a Mission Level Design flow, the top-level architecture and performance requirements are modeled using discrete event (DE) models and finite state machines (FSM). Initial HW models abstract resources such as memory, buses, CPU cycles, etc into quantity and server resources. Software is modeled by their execution time on a target HW (this is later updated with estimated values from the SW performance estimator.) The mission model may be diverse and is typically modeled by DE, FSM and continuous time (CT) models. Challenges that have to be addressed at the architectural/performance level include [5,6],



- Dealing with complex architectures, with complex functionality in each subsystem und a high degree of concurrent processing,
- Dealing with dynamic events with complex interactions between subsystems,
- Dealing with data, tasks & architecture dependent interactions, and
- Dealing with use cases and mission scenarios

Design iterations at this level of abstraction reduces the risk of design errors by testing the design early in the design process, where errors are easy to fix. Experts suggest that design at the performance/architectural level can determine as much as 80-90% of a system's total cost, performance and time to market.

Functional level models in electronic design for signal processing and control applications typically also require Synchronous Data Flow (SDF) and Dynamic Data Flow (DDF) execution models for descriptions of signal processing algorithms. Descriptions by FSMs permit to apply formal methods for verification, and significantly reduce the risk of construction errors.

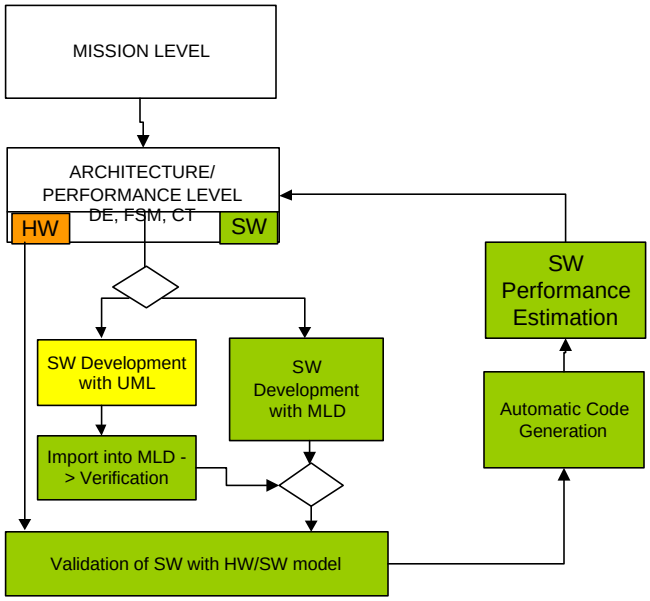
The SW performance estimator [7] determines the number of cycles, an embedded SW takes to execute on a processor as a function of compiler optimization. This updated value is used in the architecture/performance model to check the design against the specifications of the overall system.

The design and analysis software system MLDesigner [8], has been developed to implement the Mission Level Design flow. Besides the execution models mentioned above, it also includes a SystemC execution model, in order to validate RTL level implementations against behavioral models of the design. Conversion utilities translate models from 1st generation products, such as Ptolemy, BONeS and COSSAP into XML model descriptions of MLDesigner. Interfaces permit co-simulation with the "children" Matlab, SatLab and Octave of the matrix language tool Ctrl-C and with Mathematica. Code-generators generate code for HW and embedded SW [9]. Socket interfaces provide the ability to dynamically interface to other design tools, to internet resident applications and to hardware. Applications include embedded system design, processor and computer architecture

performance analysis, SOC co-design, wireless chip, handset and system architectural performance, and production, workflow and design process design and analysis.

SOFTWARE DEVELOPMENT

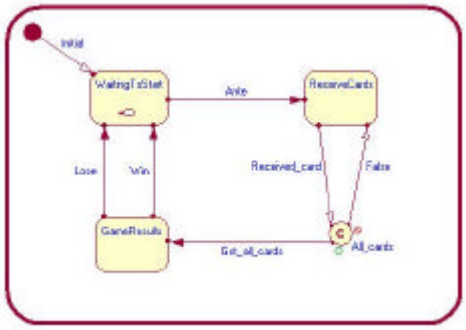
Software development plays an increasing role in the design of avionics systems. The development of UML significantly increased the design of software from given specifications. The major challenges are that the high probability of critical problems due to specifications, Figure 1, and the developed software is not validated against the use of it together with the hardware. To overcome this problem, software development can be integrated in a mission level design flow for software development, verification and alidation

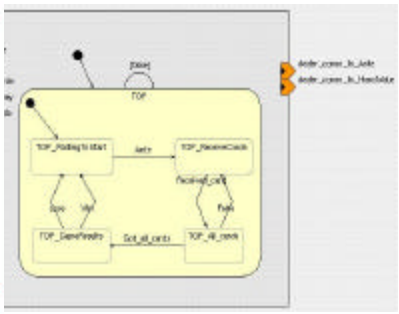


An executable and validated model of the architectural/performance level specifications is developed from mission level use cases, environmental models and other specifications. This model includes a consistent data transport model, that typically takes 80% of the development time when performed at the functional level. Moving the data transport to the front end of the design significantly reduces the development time. We compared this design flow with an UML design flow for the development of an embedded software system for the ESA packet utilization standard (PUS). The UML design flow took 6 month; the mission-level design flow took 10 days.

In the MLDesigner (right hand side) design flow [9], a FSM model of the embedded SW is developed in MLDesigner. It uses the verifier of the MLDesigner FSM for verification and validates the FSM model by simulation in a virtual test environment of the target system. The ANSI C-code generator generates the code for the verified and validated software model for the target system. The execution time of the generated code is determined by the software performance estimator and compared with the assumptions in the architecture/performance model.

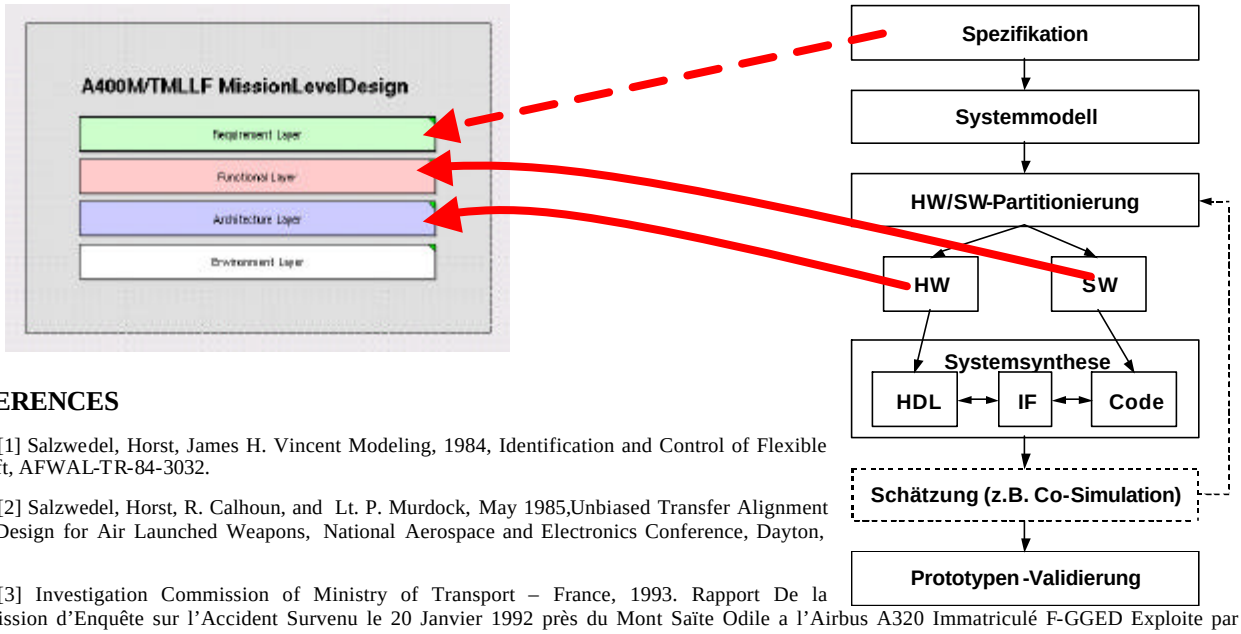
In the MLDesigner/UML design flow [11], a FSM model of the SW is developed in the UML tool Real Time Rational Rose®. The RTRR FSM model, shown on the left, is translated into an MLDesigner FSM model, shown right. The model is verified with the MLDesigner verifier, in order to check for construction errors. The validation is done the same way as that in the MLDesigner only design flow.





Applications

In [10] a new Mission Level Design approach for the HW/SW co-design for the terrain following system of an aircraft. All components are modeled at the architectural/performance level. The MLDesigner module for the overall model includes architectural components and functional components. The architectural module includes elements like power generation, terrain flight control, TMLLFC, and full duplex Ethernet bus, AFDX. The functional module shows the components of the TMLLFC, and the FSM model shows the Guidance mode of the TMLLFC. Additionally an MLDesigner operating system kernel model has been developed to perform a performance analysis of the overall system, consisting of hardware, software and operating system.



REFERENCES

[1] Salzwedel, Horst, James H. Vincent Modeling, 1984, Identification and Control of Flexible Aircraft, AFWAL-TR-84-3032.

[2] Salzwedel, Horst, R. Calhoun, and Lt. P. Murdock, May 1985,Unbiased Transfer Alignment Filter Design for Air Launched Weapons, National Aerospace and Electronics Conference, Dayton, Ohio.

[3] Investigation Commission of Ministry of Transport – France, 1993. Rapport De la Commission d'Enquête sur l'Accident Survenu le 20 Janvier 1992 près du Mont Saïte Odile a l'Airbus A320 Immatriculé F-GGED Exploite par la Compagne Air Inter.

[4] Schorcht, Gunar, July 2000, Design of Integrated Mobile Communication Systems at Mission Level, Dissertation at Ilmenau Technical University.

[5] Schorcht, Gunar, Ian Troxel, Keyvan Frahangian, Peter Unger, Daniel Zinn, Colin Mick, Alan George, Horst Salzwedel, MASCOTS 2003 Conference, System-Level Modeling with MLDesigner.

[6] Ian Troxel, W. Chris Catoe, Ramesh Balasubramanian, Alan D George, November 2004, HCS Research Lab., Jeremy D. Wills, Sung J. Kim, Gregory A. Arundale, Rockwell Collins, John L. Meier, Boeing, Colin K. Mick, MLDesign, LION: Virtual Prototyping for Advanced Optical Military Networks, MILCOM, Monterey.

[7] Lohfelder, Thomas, Holger Rath, Horst Salzwedel, 27-30 September 2004, Software Performance Estimation for a Mission Level Design Flow, 49th International Scientific Colloquium, Ilmenau.

[8] MLDesigner® Manual v2.4, http://www.mldesigner.com

[9] Rath, Holger ,Horst Salzwedel, 27-30 September 2004, ANSI C Code Generation for MLDesigner Finite State Machines, 49th International Scientific Colloquium, Ilmenau.

[10] Paluch, Nils, Achim Schönhoff, EADS, 27-30 September 2004, Anwendung von C-Design für verteilte Echtzeitsysteme, 49th International Scientific Colloquium, Ilmenau.

[11] Baumann, Tommy, Horst Salzwedel, 27-30 September 2004, Integrating UML Software Models in an Integrated Mission Level Design Flow for the Design of Hardware and Software, 49th International Scientific Colloquium, Ilmenau.

Autorenangabe:

Prof. Horst Salzwedel. TU Ilmenau, PF 10 05 65, 98684 Ilmenau, Tel.: 03677/691316, Fax: 03677/69128