Aerospace Science and Technology 54 (2016) 183–191

Contents lists available at ScienceDirect

Aerospace Science and Technology

www.elsevier.com/locate/aescte

# Geometry and simulation modeling in design languages [☆]

Johannes Gross [*], Stephan Rudolph

*Institute for Statics and Dynamics of Aerospace Structures, University of Stuttgart, Pfaffenwaldring 27, 70596 Stuttgart, Germany*

A R T I C L E  I N F O

*Article history:*
Received 5 June 2014
Received in revised form 11 January 2016
Accepted 4 March 2016
Available online 22 April 2016

*Keywords:*
Design languages
Model based engineering
UML

A B S T R A C T

In this third paper of the series [1,2] about satellite design languages, the detailed geometry and simulation modeling is described. The definition of the FireSat subsystems in different design languages is presented in the first paper. In the second part, novel analytical design evaluations resulting from these means are shown. This paper focuses on the detailed simulation models that can be generated out of the design languages. These simulation models provide the means to solve the field problems based on differential equations in the specialized engineering applications. From an abstract geometry description the design languages can be exported in different computer aided design tools (CATIA, OpenCascade, etc.). All models shown in this paper are entirely generated out of a description within the UML. The same geometry description is used for a thermal simulation of the satellite in the ESATAN thermal design suite. The third usage of the abstract geometry description is for the automated routing of the harness cables that is presented. As a non-geometric example for a simulation model, the behavioral simulation of the attitude control in a generated Matlab Simulink model is shown. All the information required to generate these models is organized in the different design languages.

## 1. Introduction

This paper is the third in a series of three papers describing recent progress in satellite design language research. The papers are using the FireSat Mission example which is first described in [3] and further explored by [4]. Delp [5,6] uses the example for demonstrating language standards and the authors of this publication show methodology aspects in [7–11].

The design language methodology and some subsystems design languages are explained in the first part [1]. Analytical evaluations of the spacecraft are shown in the second part [2]. This last part shows the integration of detailed geometry and simulation model descriptions within the design languages. The design language compiler generates the model descriptions for the established engineering applications from the design languages.

## 2. Engineering complex systems

Engineering complex systems, such as a satellite, includes solving a large number of problems among different engineering domains. These problems resolve to specific questions, that are answered from the engineer by building domain specific models of the system. This aspect of the design process can thus be seen as a sequence of models used to answer the design questions.

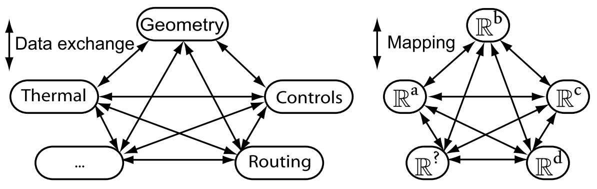

**Fig. 1.** Schema of data exchange problem [12,13].

Since the different models describe one system, they show significant overlap. The overlapping information in the different models has to be kept consistent along the design process. Additionally, the creation of the overlapping information in these models is redundant work. To reduce the redundant work, the data exchange between the engineering domain models is a common practice nowadays. In Fig. 1 some exemplary domain models and the interfaces between them are shown.

The federated data exchange shown in Fig. 1 has two problems. Firstly, the maximal number of interfaces for $n$ models grows with the factor of $n(n-1)$ quadratic for larger $n$. Secondly, the models cannot be generated entirely from one another. Thus the data exchange between the models is always missing aspects. Consequently the processes are uni-directional and require manual adjustments.

[☆] This work was conducted for the PhD Thesis of the corresponding author at the University of Stuttgart.
[*] Corresponding author.
*E-mail address:* johannes.gross@jpl.nasa.gov (J. Gross).

http://dx.doi.org/10.1016/j.ast.2016.03.003
1270-9638/© 2016 Elsevier Masson SAS. All rights reserved.

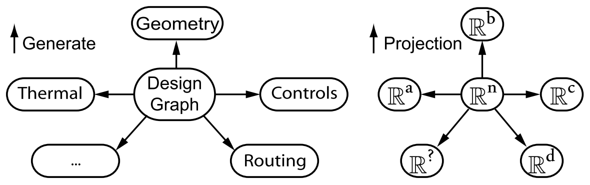

**Fig. 2.** Central data model as solution to data exchange problem [12].

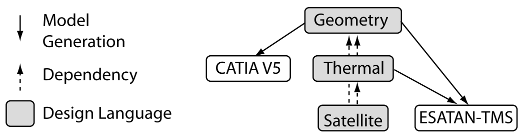

**Fig. 3.** Hierarchy of engineering domains.

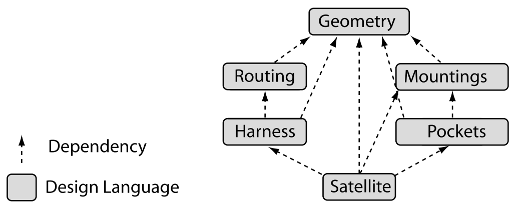

**Fig. 4.** Hierarchy of engineering problems.

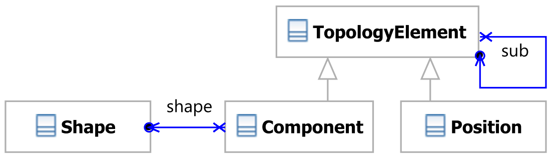

**Fig. 5.** Top level classes for the geometry description.

Mathematically, the situation can be described as shown on the right side of Fig. 1. Each parameter within the design domains is understood as a mathematical dimension. The problem statement is consequently: “Between spaces of different dimensionality exists no bijective mapping”.

One straightforward solution to this problem is shown in this paper by the use of a central model called “Design Graph” incorporating all relevant aspects of the different problem domains. Using the design language methodology, the domain specific models are generated from the central design graph. The model generation is in a mathematical view a projection from the higher dimensional design graph to the lower dimensional domain model. Fig. 2 shows the design graph as a central artefact in the data exchange problem.

To account for modularity, the different layers of abstraction that are required along the design process are realized within the central model by a hierarchy of domains. In Fig. 3 an exemplary hierarchy of the geometry, the thermal model and a satellite system is shown. The most general aspect, in this case the geometry, is on the top. The thermal modeling relies on a geometry and thus depends on it. The satellite incorporates both geometric and thermal issues and depends on both domains. Other engineering domains as for example a structural model would also depend on the geometry model but have a different group of physical parameters. To incorporate that into the graph above, a materials language would have to be defined that is above the thermal and the structural model.

The categories in the hierarchy can be either of an engineering domain like geometry, thermal or structural engineering but it can be also of a systems domain e.g. the satellite system or subsystems thereof. To provide a methodology capable of implementing these fundamental considerations of engineering, the so-called design languages are developed [14].

## 3. Design languages

The design language methodology allows for the flexible creation of the design graph in Fig. 2 by capturing the engineering knowledge in vocabularies that are combined by formal rules. For a detailed description of the design language methodology please refer to [1, Sec. 2]. Other example applications of design languages can be found in works by Arnold [15], Beilstein [16], Landes [17, 18] and Vogel [19,20].

## 4. Geometry and simulation model description

Many of the detailed simulation models mentioned above (e.g. thermal, structural, controls) depend on geometry. The geometry model is described using an abstract geometry design language. This means that geometric information is treated as a first class citizen within a design language and can be exported to different CAD and simulation tools. In Fig. 4 the hierarchy of the different design languages using the abstract geometry design language is shown.

On the top of Fig. 4 is the abstract geometry design language. In this design language for example the general notion of a point or a component is defined. The harness design language provides for a separate definition of the cable and connector types. The routing of the cables is done using a routing design language from Eheim [21]. This routing design language is defined based on the same abstract geometry representation as the satellite design languages and the harness design language, thus it can be seamlessly and efficiently integrated. The pockets design language on the right hand side takes effect on the satellites structural plates. But since the pockets are created in respect to the mountings of the boxes, this design language is also depending on the design language for creating mountings of the different subsystem boxes.

With this hierarchical approach the data exchange between the satellite and its boxes can be guaranteed still supporting the flexible exchange of any subsystem or component by another. In the following, the outcomes of such a flexible architecture for creating engineering products for late design phases are demonstrated. All models are generated fully automatically out of the design languages with only marginal adjustments before running the detailed high fidelity simulations.

### 4.1. Modular geometry definition

The abstract geometry is defined on the basis of a geometry design language developed by Schmidt [22]. The design language allows for the definition of a hierarchical order of parts similar to any product tree in Computer Aided Design (CAD) tools. In Fig. 5 the top level classes of that design language are shown.

A *TopologyElement* can either be a *Position* or a *Component*. The *TopologyElements* itself can have sub-Elements. If a *Component* has another *TopologyElement* as a child, it is a product, if it has a *Shape* as child, it is a part. With these four classes the hierarchy of a product tree is defined.

The *Shape* can for example be a *Cuboid*. In Fig. 6 the definition of three components of the satellite that have a *Cuboid* as geometric representation is shown.

The relationships in Fig. 6 define the geometric representation of the satellite components. The classes of the components can define the dimensions of the geometry model using these relations. The geometry model hierarchy is independent of the hierarchy of the component classes. It can be defined separately in rules.