Mechanical CAD with Multibody Dynamic Analysis Based on Modelica Simulation

Vadim Engelson, Peter Bunus, Lucian Popescu, Peter Fritzson
PELAB,Linköping University, S-58183, Sweden,
Contact e-mail: vaden@ida.liu.se

Abstract

In a comprehensive modeling and simulation environment, it is desirable to integrate models specified in different modeling formalisms. A complete integrated environment combines the powerful mechanical model design of CAD package with the structuring mechanisms of object oriented modeling language including a mathematical and logical behavior representation. We present a system for simulation of multi-domain models, which has been implemented using Modelica as a standard model representation. The user can work with mechanical models designed with AutoDesk's Mechanical Desktop, extend it in various ways, and analyze the simulation results in a high performance interactive visualization environment.

Introduction

Modelica is a new language for hierarchical object-oriented physical modeling. The multi-domain capability of Modelica gives the user the possibility to combine mechanical and other domain model components within the same application model. Interaction between system components, which are often complex and difficult to analyze, can be easily studied.

In order to automate the design of mechanical models, CAD tools can be utilized. We have focused on adding the ability to easily interface the Modelica simulation environment with a wider variety of CAD systems. At the first stage we have focused on achieving full integration with widely accepted mechanical CAD solutions like SolidWorks and Mechanical Desktop. Our SolidWorks to Modelica translator described in [Engelson 2000] is available commercially. In this report we present a translator implementation from AutoDesk's Mechanical Desktop to Modelica, which extracts geometric and structure information from an existing designed mechanical model and produces a corresponding set of Modelica components (class instances) with connections between them.

AutoDesk's Mechanical Desktop 4 is an integrated package of advanced 3D modeling tools and 2D drafting and drawing capabilities. The proposed integrated environment consists of a CAD tool, a translator from CAD to Modelica, a simulation environment, like Dymola or MathModelica, and a visualizer that provides online dynamic display of the assembly (during simulation) or offline (based on saved state information for each time step). Either stand-alone tool or the same CAD tool (Mechanical Desktop) is used for result visualization.

This integrated environment allows designers and engineers to build very quickly a virtual prototype, which enables them to choose the optimal design. In Modelica it is possible to model and simulate both control and mechanical aspects of the desired mechanical application. The dynamics of mechanical and/or control systems is also documented by plots of system variables.

Initially user creates a static model without any dynamic capabilities. Our plug-in to the CAD package extracts from the drawing the geometry, mass, inertia and constraints information, translating them to a simulation language source code. This code is combined with other code fragments (e.g. control systems), simulated, and the output can be visualized as a data plot of the system variables and/or as a 3D or 2D dynamic model animation. The 3D visualizations are scenes that display the geometry of the parts in motions prescribed by the simulation results.

In this paper, we first give a brief introduction to the overall design of the developed simulation environment. At the same time, we examine the implementation details of the developed translator. A brief overview of the Modelica language is also given with an emphasis on the modular and hierarchical facilities of the language. The Modelica Multi Body System Library (MBS) is briefly presented together with a simple modeling and simulation example. We will also present some principles of the developed translator implementation. The use of the translator is demonstrated on an industrial robot examples. We conclude with an overview concerning further development based on the integrated design and simulation environment.

The Modelica Modeling Language

Modelica is a new language for hierarchical object-oriented physical modeling, which is developed through an international effort [Fritzson and Engelson 1998; Elmqvist et al. 1999].

Compared to other modeling languages available today, Modelica supports:

- Acausal modeling based on ODE and differential algebraic equations (DAE).

- Multi-domain modeling capability, which provides the user with the possibility to combine electrical, mechanical, thermodynamic, hydraulic etc., model components within the same application model.
- A general type system that unifies object-orientation, multiple inheritance, and templates within a single class construct. This facilitates reuse of components and evolution of models.

The reader of the report is referred to [Modelica Association 2000] and [Modelica Association 2002] for a complete description of the language. Those interested in shorter overviews of the language may wish to consult [Fritzson and Engelson 1998] or [Elmqvist et al. 1999].

The dynamic simulation capabilities of the language has been demonstrated many times in the literature by modeling and simulating heat exchangers [Mattsson 1997], automatic gear boxes [Otter et al 1997] or hydraulic systems [Ferreira et al 1999], [Tummescheit and Eborn 1998].

Modelica models can be specified in textual notation or as connection diagrams. In this paper we use the diagram notation for simplicity

Related Work

In this part of the paper, we briefly survey some of the commercial virtual prototyping packages available which are most closely related to our developed environment.

VisualNastran 4D from MSC Working Knowledge provides an integrated environment for motion and FEA (Finite Element Analysis) simulation and complete suite of tools for the development and communication of physics-based virtual prototypes. Constraints and drivers can be defined by numeric or equation input in the formula editor, or with tabular data.

ADAMS, developed by Mechanical Dynamics Inc., provides a fully integrated virtual prototyping environment. In addition to the powerful modeling and visualization capabilities includes an analysis engine called ADAMS/Solver, which converts an ADAMS model to equations of motion, and then solves the equations, typically in the time domain. ADAMS/Solver can resolve redundant constraints, handle unlimited degrees of freedom, and perform static equilibrium, kinematic, and dynamic analyses.

Dynamic Designer/Motion and **Simply Motion**, two other products from Mechanical Dynamics Inc., provide a full integration with AutoCAD Mechanical Desktop. Simply Motion written also in AutoDesk's ARX development language extends the design automation capabilities of Mechanical Desktop to include realistic 3D dynamic motion simulation. Simply Motion anticipates the mechanical designer needs and automatically updates the motion data. Through a browser called IntelliMotion Browser, the user can add joints, springs and input motion to the mechanical model.

DADS, standing for Dynamic Analysis and Design System available from LMS International Inc. (CADSI), performs assembly, kinematic, dynamic, inverse dynamic and preload analysis. It incorporates advanced numerical methods to solve Differential Algebraic Equations (DAE) using both implicit and explicit solvers.

The primary limitation of these environments is the difficulty of integrating multi-domain simulation in the same environment. Usually an interface to other popular simulation tools, like **MATLAB** and **Simulink**, is provided, but this solution does not offer too much flexibility. We have identified two major needs for a virtual prototyping system:

- The need to integrate multi-domain simulation in the same environment.
- The generation of quality documentation coupled to the design and code.

In the following pages, we detail our proposed procedure for avoiding the current limitations of the software in this area.

MBS (Multi Body System) Library in Modelica

The MBS (Multi Body System) library has been developed in [Otter 1995], and an overview can be found in [Otter et al. 1996]. We briefly present the multi-body system library together with a simple modeling example.

Multi body system modeling then is not just concerned with placement of bodies; it requires a description of element connectivity, i.e. the constraints controlling the degree of freedom. Such parameters can then be investigated by attached simulation code used to subject the model to various forms of dynamic behavior. A distinguishing feature of mechanical multi-body systems is the presence of joints, which impose different types of kinematic constrains between the various bodies of the system. Kinematic constraints are enforced between the kinematic variables of two bodies. These constraints express the conditions for relative translation or rotation of the two bodies along or around a body fixed axis.

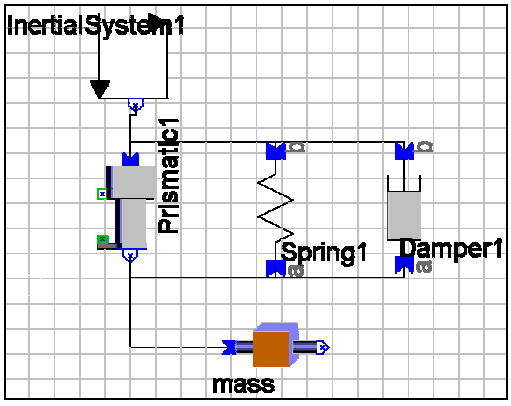
The MBS library contains bodies, joints, coordinate system transformations, forces, torques, and a class representing the inertial system. In order to correctly simulate a mechanism, it is necessary to have a kinematic chain with one link fixed. Such link is chosen as a frame of reference for all other links. The CAD environment has the possibility of specifying which part of the kinematic chain will be fixed. Later, in the translation phase, this fixed part will be connected with an instance of the `InertialSystem` class. Every basic mechanical component from the MBS library has at

least one or two interfaces (ports, connectors, flanges) to connect the element to other mechanical elements. All objects are in some way connected to the inertial system, either directly or through other objects.

The MBS library usage is shown by the following modeling **example**:

The system consists of a mass hanging on a spring and damper in a gravity field.

A fragment of Modelica code for the model is shown below (MBS library classes and their components are written in *italics*):




```
model Hanging
  Parts.InertialSystem     InertialSysrem1
"must be in any MBS system";
  Joints.Prismatic  Prismatic1(n={0,-1,0})
"directed vertically down, along y-axis";
  Parts.Body2  mass(r={0,-1,0},m=1)  "position
for center of mass; weigth 1 kg";
  Forces.Spring       Spring1(c=300,s0=0.5)
"ideal spring coefficient, and unstretched string length" ;
  Forces.Damper  Damper1(d=2)  "ideal damper
coefficient";
```

connect(…,…) describe links between components

Mates in Mechanical Desktop

AutoDesk's Mechanical Desktop, like other typical CAD/CAM systems supports modeling the geometry of parts. In an assembly model, these parts are put together in order to form a complete model.

The assembly document defines the mobility between the parts of an assembly. After parts have been created, constraints are applied to position them relative to one another. Each time when a constraint is applied to a part, some degrees of freedom are eliminated. Simultaneously Mechanical Desktop checks that the assembly is structurally sound.

The Mechanical Desktop offers the **Mate** menu option. When two elements (vertices, straight or circular edges, plain or curved faces) on different parts are selected the following constraint types can be specified:

- Two planes coplanar with their normals aligned in opposite directions (facing each other).
- An axis *planar* with a plane (belonging to the plane).
- Two axes that share the same direction and slope (collinear).
- A point that lies on an axis.
- Two coincident points.
- A sphere, cylinder, or cone tangent to a plane or to other spheres, cylinders, and cones.
- A point that lies on a plane.

These constrains are automatically mapped to one of Mechanical Desktop kernel commands:

- **AMMATE** - Mate constraint. Causes a plane, axis or point on one part to be coincident with a plane, axis or point on another part in a specified direction. Removes a translational or rotational degree(s) of freedom.
- **AMFLUSH** - Flush constraint. Make two planes coplanar (i.e. parallel, but not necessary coincident) with their faces aligned in the same direction. The offset between the planes is fixed.
- **AMINSERT** - Insert constraint. Aligns center points and planes of two circles in a specified direction. Removes translational degree(s) of freedom. Used to constrain a bolt in a hole, for example.
- **AMMANGLE** – Angular Constraint. Specifies an angle between two planes, two vectors, or a combination of a plane and a vector.

Translation of mates into joints

The translator first gathers all pairs of parts that have constraints between them. After that for each such pair it gathers the information about the mate constraints applied between the two parts. This information is translated to a corresponding joint object from the MBS library.

Each valid combination of mates will be translated to a single MBS joint or a combination of MBS joints. Table 1 lists the names of the lower joints, together with the number of translational and rotational degrees of freedom [Shigley 1995], and their correspondents from the MBS library. All other joint types are called higher pairs. They are combinations of the basic joints and are not listed in Table 1.

Pair	Tr. and Rot. DOF	Total Nr. of DOF	Relative motion	MBS name
revolute	1 rot	1	circular	`Revolute`
prismatic	1 tr	1	linear	`Prismatic`
cylindrical	1rot+1 tr	2	cylindrical	`Cylindrical`
sphere	3 rot	3	spherical	`Spherical`
flat	1rot+2 tr	3	planar	`Planar`

Table 1 : Lower Joints

Revolute Joint: Removes 5 degrees of freedom, 3 translational and 2 rotational. It corresponds to the combination of one Mate Plane/Plane and one Mate

Line/Line constraint. Requires an axis of revolution on each part.

Cylindrical Joint: Removes 4 degrees of freedom, 2 translational and 2 rotational. Allows parts to rotate and translate along a common axis. Corresponds to Mate Line/Line constraint. Requires an axis of revolution/translation on each part.

Spherical Joint: Removes 3 degrees of freedom, all 3 translational. Allow parts to rotate about a single point. Corresponds to Mate Point/Point constraint.

Prismatic Joint: Removes 5 degrees of freedom, 2 translational and 3 rotational. Allows parts to only translate along an axis. It corresponds to a combination of Mate Line/Line and Mate Plane/Plane constraints (however, several different combinations can be used). Requires an axis of translation on each part.

Planar Joint: Removes 3 degrees of freedom, 1 translational and 2 rotational. Allows part to translate and rotate in a plane. It corresponds to a mate Plane/Plane constraint. Requires a plane on each part.

Universal Joint: Removes 4 degrees of freedom, 3 translational and 1 rotational. Allows parts to rotate about two orthogonal axes. Requires an axis of rotation on each part. The axes must be orthogonal when the parts are assembled.

Fixed Joint: Removes all 6 degrees of freedom and rigidly connects one part to another. Parts connected this way are treated with a single rigid part.

There are two alternatives in using AutoCAD information in order produce MBS joints. First is to access all details of mates through the API and analyze the combination of mates and their DOF constraints. This approach has been implemented in the SolidWorks to Modelica translator. Another alternative, which is much simpler is to request DOF information directly from the API (this option is missing in SolidWorks API). We analyze which pairs of bodies have constraints between them. Then we combine DOF-s of each pair of bodies and derive the matching MBS joint type.

Translator Implementation

The translator was implemented as a plug-in to AutoDesk's Mechanical Desktop by using a provided application development tool, AutoDesk Mechanical Application Programming Interface (MCAD API) [AutoDesk 1999b]. The MCAD API provide a direct and unified access mechanism for AutoCAD and Mechanical Desktop for extraction of geometrical data, mass, inertia and constraints. From the AutoCAD point of view, our translator is an ObjectARX application, a dynamic link library (DLL) that makes direct function calls to AutoCAD. It is possible to add new classes to the ObjectARX. The ObjectARX entities created are virtually indistinguishable from the built-in AutoCAD entities. The ObjectARX protocol can be extended by adding functions at runtime to existing AutoCAD classes [AutoDesk1999a].
Mechanical models are saved in the DWG format, which contains all the information related to the geometrical properties of the parts and information related to the mechanical assembly like mates and constraints, as well as arbitrary "user-defined" objects necessary for complete code generation and missing in the assembly (strings, motors, external forces etc.) This is a binary proprietary format, which cannot be analyzed without Mechanical Desktop software.

The geometry of each part is exported to the STL file format [3Dsystems, 2000]. This format describes part geometry as a list of triangles. At the same time, mass and inertia of the parts are extracted together with mates information from the mechanical assembly. The translator will use this information to generate a corresponding set of Modelica class instances with connections between them. This automatically generated Modelica file is processed by a simulation environment like Dymola [Elmqvist 1996] or MathModelica [www.mathcore.com]. In contrast to other virtual prototyping environments presented in the related work chapter, our environment creates readable Modelica code so the programmer can combine it with other code fragments and modify it if necessary. For instance, the simulation code can be enhanced by adding other components from other Modelica libraries or by adding externally defined C code. In that phase electrical, control or hydraulics components can be added to the generated mechanical model, providing in that way a multi-domain simulation.

By default, only the gravity force is applied to the translated model. The translated CAD model models a set of dynamic equations of motion. Therefore, the simulation can predict the mechanism response to a given set of initial conditions or force (or torque) load, which might be function of time. External load can be specified by adding an instance of `ExtForce` (or `ExtTorque`) class from the MBS library.

The results of the simulation can be visualized as 2D plotting of the simulation variables or as a 3D dynamic animation of the mechanical assembly with the MVIS (Modelica VISualizer) and OpenGL based interactive visualization tool [Engelson 2000]. Compared to the similar translator developed for SolidWorks [Larson 1999], the main improvement of the Mechanical Desktop to Modelica translator is the possibility of visualizing the simulation result *inside the CAD editor*. For this purpose the simulation results are fed back into AutoCAD plug-in which forces the parts in the assembly to move on the screen according to stored translation and rotation data. This is usually combined with visualization of variables in the 2D X-Y plot viewer incorporated in the simulation environment.

By using the "Interference" option the user can examine two parts to see if they interfere with each other. Then the system uses the simulation data to move

all the parts and check for interference between the parts selected. The output can be seen on the screen or saved as a text-file. If the user chooses to see the results on the screen then the tool displays specially enlightened object equal to the intersection of two bodies.
The overall system will produce a dynamic multi-body system simulation in time domain for the evaluation of the dynamic interaction between several parts of the mechanical system or between the mechanical assembly and the attached controller. This is very useful in optimizing the mechanical components geometry, and leads to a good deal of confidence when it comes the time to go from drawings to fabricating the equipment. Simulating a couple of scenarios with mechanical rigid body models and inspecting animations and numerical results has validated our approach.

A Simple Robot Translation and Simulation

A simulation example represents a simple 6-DOF robot. The robot model from Figure 1 is modeled and designed in Mechanical Desktop.

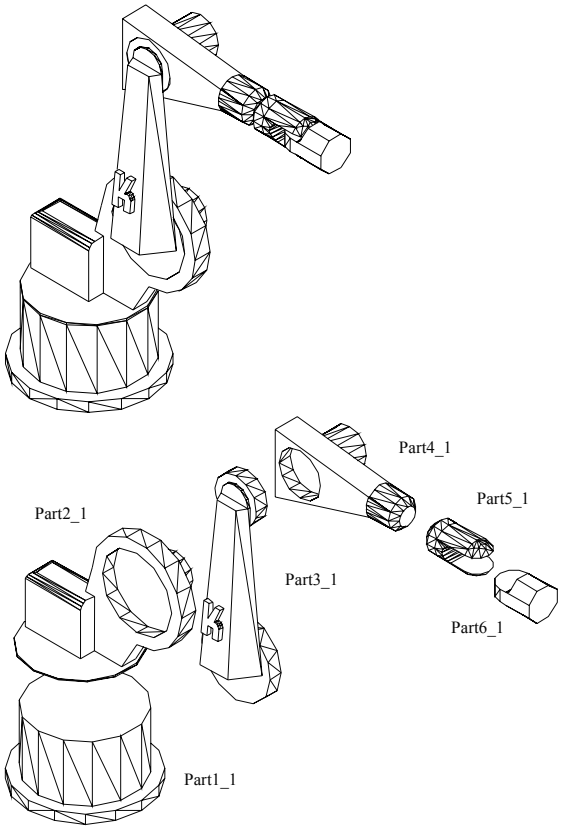


Figure 1 : Above: robot modeled and designed with AutoCAD Mechanical Desktop; below: exploded view of the robot model

The translator detects 6 parts in this assembly and determines that they are connected by a chain of revolute joints. A fragment of automatically generated code is shown below. Corresponding diagram appears to be too large in order to be inserted into this paper.

```
model ROBOT1
Interfaces.Frame_a aa;
Parts.FrameTranslation I(r={0,0,0});
Body PART1_1
 (r={0, 167.5, -4.10243e-014},
       I11=1343.67, I22=1128.16,
       I33=1343.67,I21=9.04166e-014,
       I31=-4.27363e-014,
       I32=-2.03526e-012, mass=179.594,
       r0={0, 0, 0},nx={1, 0, 0},
       ny={0, 1, 0},
       Material={0.5, 0.5, 0.5, 0.5},
       parmStlIndex=10);
Parts.FrameTranslation PART1_1_Bar0
       (r={0,8.24184,-9.79685e-016});
Joints.Revolute J0
       (startValueFixed=true,n={0,1,0});
(…here 5 other parts are inserted…)
equation connect(aa,I.frame_a);
connect(I.frame_b,PART1_1.frame_a);
connect(I.frame_b,PART1_1_Bar0.frame_a);
connect(PART1_1_Bar0.frame_b,J0.frame_a);
(…here all other connections are
insered…)
end ROBOT1;
```

Summary And Future Work

The objective of the work presented herein was to demonstrate that the integration of a typical CAD/CAM system and an equation based simulation environment could produce a feasible virtual prototyping environment with an enhanced flexibility compared to other traditional commercially available environments. The main advantage of our environment is that the multi-domain simulation is made possible in the same environment.

The improved CAD integration provides the users with a more intuitive and sound way of constructing and verifying large, moving assemblies. In that way designers can take into account the dynamic nature of the problem and simulate the entire mechanical assembly, rather than visualize a static part or a small subassembly, resulting in more accurate modeling and design solutions

Commercially available MBS simulation packages like ADAMS or Working Model 3D cannot be directly used for modeling the motion of mechanical systems with attached components from other application domains. To solve this problem we propose a methodology based on the utilization of possibilities available from both Mechanical Desktop and Modelica. The efficiency of the proposed methodology has been illustrated by the modeling and simulation of the motion of a robot. The combination of two environments, the CAD modeling environment and the simulation environment, in effect, collapse the phase of coding.

In the future, we shall concentrate on the following tasks:

- Better collision detection handling and visualization of forces and effects of collisions.

- Automatic output of the force data to finite element analysis (FEA) packages for structural analysis and other applications.

Acknowledgements

This paper is supported by VINNOVA (Verket för innovationssystem, Sweden) in the VISP (Virtuell Integrerad Simuleringsstödd Produktframtagning) project.

References

3Dsystems Inc. 2000. "Stereo Lithography Interface Specification."

AutoDesk Inc. 1999a. "AutoCAD 2000 ObjectARX Developer's Guide.

AutoDesk Inc. 1999b. "Mechanical Application Programming Interface [API] – Developers Guide.

Elmqvist, H.; S. E. Mattsson and M. Otter. 1999. "Modelica - A Language for Physical System Modeling, Visualization and Interaction." In Proceedings of the 1999 IEEE Symposium on Computer-Aided Control System Design (Hawaii, Aug. 22-27).

Elmqvist, H.; D. Bruck; M. Otter. 1996. *Dymola – User's Manual*. Dynasim AB, Research Park Ideon, Lund

Engelson, V. 2000. "Tools for Design, Interactive Simulation, and Visualization of Object-Oriented Models in Scientific Computing". Ph.D. Thesis, IDA, Linköping University, Sweden.

Engelson, V.; H. Larsson; P. Fritzson. 1999. "A Design, Simulation and Visualization Environment for Object-Oriented Mechanical and Multi-Domain Models in Modelica. " In *Proceedings of 1999 IEEE International Conference on Information Visualization* (London, July 14-16), 188-193.

Fritzson P. and V Engelson. 1998. "Modelica - A Unified Object-Oriented Language for System Modeling and Simulation." In *Proceedings of the 12th European Conference on Object-Oriented Programming* (ECOOP'98 , Brussels, Belgium, Jul. 20-24).

Ferreira, J.A.; J.E de Oliveira; V.A. Costa; 1999. "Modeling of Hydraulic Systems for Harware-in-the-loop Simulation: a Methodology Proposal." In Proceedings of *The International Mechanical Engineering Congress and Exposition.* (Nashville, USA, November 14-19).

Larson, H.,1999. Translation of 3D CAD models to Modelica. Master thesis. IDA, Linköping University, Sweden, April.

Mattsson, S. E. 1997. "On Modeling of Heat Exchangers in Modelica. " In *Proceedings of European Simulation Symposium* (Passau, Germany, October 19-22)

Modelica Association 2000. *Modelica – A Unified Object-Oriented Language for Physical Systems Modeling - Tutorial and Design Rationale Version 1.4,* www.modelica.org.

Modelica Association 2002. *Modelica – A Unified Object-Oriented Language for Physical Systems Modeling – Language Specification Version 2.0*, www.modelica.org

Otter, M.; C. Schlegel; H. Elmqvist. 1997 "Modeling and Realtime Simulation of Automatic Gearbox using Modelica." In *Proceedings of European Simulation Symposium* (Passau, Germany, October 19-22).

Otter, M. ; Elmqvist H.; F. E. Cellier. 1996. "Modeling of Multibody Systems with the Object-oriented Modeling Language Dymola, Nonlinear Dynamics, 9:91-112, Kluwer Academic Publishers.

Otter, M. 1995 Objektorientierte Modellierung mechatronischer Systeme am Beispiel geregelter Roboter, Dissertation, Fortshrittberichte VDI, Reihe 20, Nr 147.

Shigley, J. E. 1995. *Theory of Machines and Mechanisms.* McGraw-Hill, New York series in mechanical engineering.

Tummescheit H.; J.Eborn. 1998. "Design of Thermo-Hydraulic Library in Modelica." In Proceedings of *The 12th European Simulation Multiconference* (Machester, UK, June 16-19)