Engineering with Computers (1996) 12: 23–33

Engineering with Computers

An Integrated System for Computer Aided Design and Analysis of Multibody Systems

C. Hardell

Division of Computer Aided Design, Department of Mechanical Engineering, Luleå University of Technology, S-971 87 Luleå, Sweden

Abstract. *An integrated system for design and analysis of multibody systems has been developed and is described in this paper. The use of the system is demonstrated through the example provided. The system consists of a commercially available CAD program and a multibody system analysis code developed at the Royal Institute of Technology in Stockholm, These tools are integrated using a relational database, the structure of which was developed at Luleå University of Technology, in order to get a complete system for design and analysis of multibody systems. The main gains from the integrated system are the possibilities of using calculated component data like mass and moment of inertia from the CAD program in the simulation models, the automatic formulation of input files for the analysis code, and finally the visualization of simulation results using the surface data of the solid models. The interactive structured query language (ISQL) of the database management system provides the possibility of examining the components of the multibody system during the design work and before any simulation is performed.*

Keywords. CAD; Database; Multibody system analysis

1. Introduction

Software tools used in the design of components of multibody systems are well developed and widely used [1–4]. This is also the case for software for the analysis of the behaviour of multibody systems [5]. In industry these tools have, so far with few exceptions, been used in a way that leads to time-consuming and unnecessary work for the analysts. The specification of the models used for multibody system analysis has been so complicated and time-consuming that analysis has been mainly used to validate existing designs.

In a modern design environment, designers use CAD programs to create solid models of components of the multibody systems. Two-dimensional drawings can be created by viewing these solid models from three orthogonal directions. In order to transfer the information to the analysts, the drawings are often distributed in a 'throw over the wall' manner to the analysts who create models of multibody systems for their multibody system analysis (MSA) software.

Analysts who perform dynamic multibody system analysis need the mass, moments of inertia and the location of the centre of gravity of each moving component . In reality, few components have geometries for which it is a trial task to compute these properties. Therefore, the 'throw over the wall' manner, where the simulation model specification starts from drawings, will be tedious and error-prone. Moreover, if the analyst wishes to create an animation, in order to gain an overall idea of the behaviour of the multibody system, an additional model including a surface description for visualization has also to be created.

To speed up the engineering process and reduce sources of error, a computer aided system that centres on solid models is needed. Solid models contain all the physical characteristics of the product, making them much more accurate than wireframes or drawings. They contain information about the geometry of the product (points, curves, surfaces and volumes) as well as volumetric properties like mass and inertia. After solid models have been created, their properties should be accessible to all engineers who need them, making it possible for all team members to visualize a product from the earliest stages of design. The benefits for multibody system analysts are obvious: they can extract all the properties needed for dynamic analysis and visualization as soon as the solid models have been created.

The goal of on-going work at Luleå University of Technology is to develop a computer aided concurrent engineering environment in which analysis drives

Correspondence and offprint requests to: C. Hardell, Division of Computer Aided Design, Department of Mechanical Engineering, Luleå University of Technology, S-971 87 Luleå, Sweden.

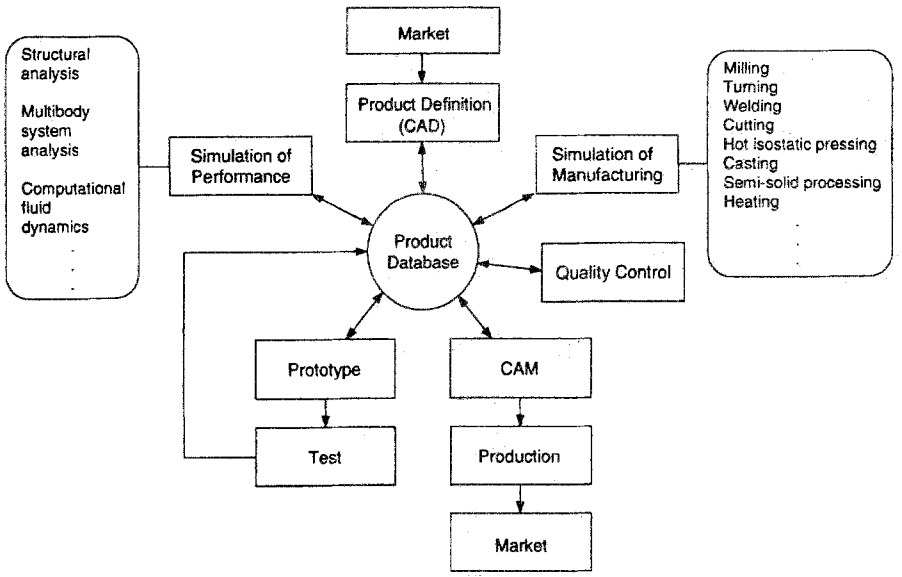


Fig. 1. Data flow of a computer aided concurrent engineering environment.

the design, as opposed to validating it. A design team should be provided with early and continuous feeback on many aspects of the design, including such attributes as performance, survivability, reliability and manufacturability. Inconsistencies, design flaws and so on should be trapped early in the design phase which greatly reduces expensive and time-consuming redesign late in the development process. The data flow of such an environment is shown in Fig. 1.

The result of this work is an integrated system for design and analysis of multibody systems. It consists of a CAD program and a code for kinematic and dynamic analysis, integrated using a neutral database, as shown in Fig. 2.

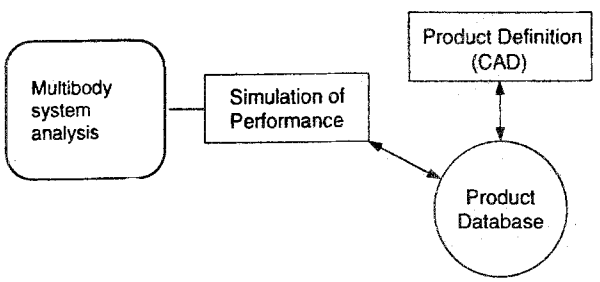


Fig. 2. The part discussed in this paper.

Most of the work related to the work described in this paper has been performed with commercial and numerical systems. An example is the direct interface between DADS [5] and PRO/Engineer [4] modelling system which was announced in November 1991. This interface allows users to transfer PRO/Engineer data into DADS. Solid model body data (mass, inertia, centres of gravity) are automatically extracted for use in the DADS analysis. This type of integrated system has some of the benefits of our system. The data needed for kinematic and dynamic analysis are determined from the solid models. The major disadvantage is that the product data are stored in the specific database of PRO/Engineer. The use of the database of PRO/Engineer makes this system dependent on a specific application software. This is not wanted in the work described in this paper.

The research group at Institute B of Mechanics at the University of Stuttgart, Germany [6], has extracted multibody system data from a CAD program [1], stored it in a database and generated system equations using NEWEUL [5], a software for generation of equations of motion using Newton–Euler formalism. However, the detailed method of integration is not presented in [6].

The research group at the NSF–ARMY–NASA Industry/University Cooperative Research Center for Simulation and Design Optimization at the University of Iowa City, USA, is developing an integration of different engineering tools including the CAD software Unigraphics and the analysis code DADS [7]. However, details concerning the interface between CAD software and the analysis code are not found in [7].

2. Integration of Codes for Design and Analysis of Mechanical Systems

The product development process consists of many tasks that are dependent on computer based tools that operate on, more or less, the same set of data. The different activities involved in the product development of a mechanical component may be geometry definition, simulation of performance and simulation of manufacturing. One of the most important research

topics in the current developments of computer aided engineering systems is the integration of different computer-based tools used in product development. This has been achieved to some extent in commercial products. However, *there is in general no single commercial system that provides solutions to all aspects of the product development process of a specific industrial product.* A large part of the profit arising from the use of computer-based systems comes from integration, giving the possibility to transfer data from one tool to another efficiently.

In Fig. 3 the data flow of an integrated system with program-specific interfaces is shown. Here each arrow corresponds to one interface program that has to be created. In this integrated environment, a change of CAD program will require three new interface programs.

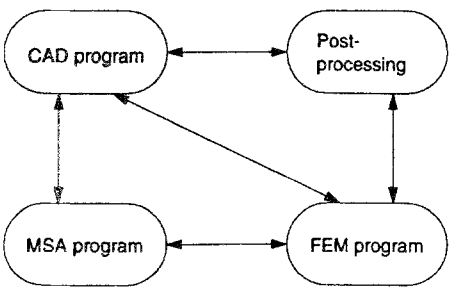


Fig. 3. Integrated system with program-specific interfaces.

To reduce the number of interfaces needed, an integrated computer aided system with a central database can be implemented, as shown in Fig. 4. When the CAD program is changed in this environment, just one new interface program has to be created.

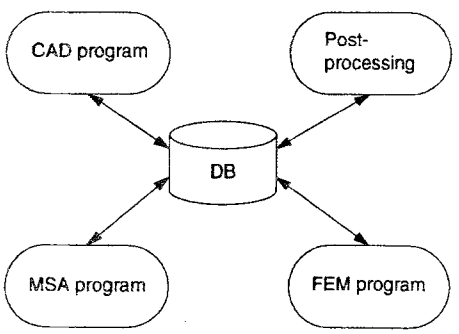


Fig. 4. Definition of multibody system data using the CAD program.

In general, the integrated computer aided engineering environment must consist of software products from different suppliers. It is important that the integration be achieved by use of protocols and tools that conform to present standards where possible, so that one program can be replaced by another with the same functionality without affecting the complete environment.

The integrated system described here consists of a CAD program, a code for kinematic and dynamic analysis, a database and interface programs, which means it is a system of the kind shown in Fig. 4. These parts are described in some detail in sections 2.1–2.3.

In section 2.1.1 the CAD program used is described with its solid modeling, multibody system modeling and data dumping capabilities. In section 2.2.1 the MSA code used is described. It provides explicit expressions for the system equations.

In section 2.3.1 the integration of the CAD program and the MSA code is described. Programs have been developed that read the multibody system data dump of the CAD program and write the data to a neutral product database. The structure of this database has been developed at Luleå University of Technology. A program that reads multibody system data from the product database and writes input files for the MSA code has also been developed. The integrated system is suitable for design and analysis of moderately complex multibody systems, especially when parameter studies and optimization are to be performed.

2.1. Software for Computer Aided Design of Multibody Systems

There are several computer programs available for computer aided design, which differ greatly in functionality. Some are just computerized drawing-boards, while others are sophisticated modular tools supporting a variety of engineering activities like solid modelling, finite element modelling and analysis, drafting and manufacturing.

Software suitable for computer aided design of multibody systems includes:

1. *Solid modelling capabilities*, making it possible to define complete and accurate properties of the components of the multibody system. The solid models contain all the rigid body data needed for dynamic analysis and all the surface data needed for visualization.

   In the CAD programs, global properties like volume, surface area, moments of inertia and centre of gravity are computed by evaluation of triple integrals such as

$$\phi_{\text{Solid}} = \int_{\text{Solid}} \mathrm{f}(\mathbf{p})\, \mathrm{d}V \tag{1}$$

   where ϕ is the property required, $\mathrm{f}(\mathbf{p})$ is a vector function describing ϕ, and integration is performed over the entire volume of the solid, see [8]. The

integral in (1) is used to calculate the properties of the primitives in the Constructive Solid Geometry (CSG) scheme. as presented on eqns (2) and (3), see [8]:

$$\phi_{\text{Solid}_1 - \text{Solid}_2} = \phi_{\text{Solid}_1} - \phi_{\text{Solid}_1 \cap \text{Solid}_2} \quad (2)$$

$$\phi_{\text{Solid}_1 \cup \text{Solid}_2} = \phi_{\text{Solid}_1} + \phi_{\text{Solid}_2} - \phi_{\text{Solid}_1 \cap \text{Solid}_2} \quad (3)$$

The global properties can also be calculated using boundary representations where surface integrals are adopted.

The results of the mass property calculation do not depend on the solid modeling scheme used (CSG or B-rep) and they can be related directly to the input entities needed for the rigid bodies modeling a multibody system.

2. *Mechanism design capabilities* where joints, springs, dampers, applied forces and applied motions of the multibody system are defined. In some cases, it may be necessary to add some components to the system after it had been transferred to the analysis code.
3. At least *one way to extract a complete set of data describing the multibody system.* Normally the data describing the multibody system can be dumped from the CAD program to ASCII files or temporary databases specific for the CAD program adopted. These databases can then be accessed with user-writtem programs.

With a CAD program including the above utilities, initial multibody system data can be created for later use in analysis.

2.1.1. The CAD Software of the Integrated System

The CAD software [2] used in the work described in this paper includes the utilities described above. It supports solid modelling and mechanism design including an internal solver for kinematic analysis. The data describing the multibody system can be dumped either to ASCII files, of which there are two different formats (universal and archive), or to temporary databases specific for the CAD program adopted. These databases can be accessed through specialized subroutines and are relatively simple to interface with. The data are well structured in tables, making it easy to find what is needed. The same sets of data can be defined in the ASCII files mentioned above, but the automation of the search for data in these files demands more programming work. Therefore the database dumping approach has been chosen in this work.

When defining a multibody system in this program, the four steps shown in Fig. 5 have to be completed:

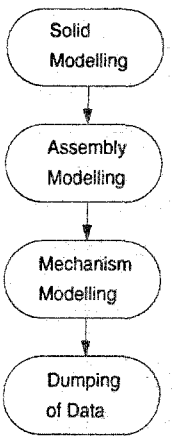


Fig. 5. Definition of multibody system data using the CAD program.

- definition of the properties of the different bodies of the multibody system using *solid modeling*;
- collection of the solid models of the system into an *assembly* model;
- creation of a *multibody system model* through definition of joints and grounds – the forces and motions applied are also defined;
- *dumping of data* of the multibody system to a temporary database which is specific for the CAD program adopted.

2.2. Software for Analysis of Multibody Systems

The determination of the motion and internal and external forces associated with interconnected multibody systems is often the very object of design. Choosing properties to obtain desired force and position characteristics is a challenging task which, for progress to be possible, must be viewed at several levels.

The complete static and dynamic response of individual components subjected to known loads can be accurately deduced using numerical methods like FEM. On the other hand, if we wish to understand inertial effects involving the motion of the entire mechanism, it is more useful to model it as made up of interconnected rigid body components. This allows the use of the techniques developed for this class of problems.

Models of multibody systems may arbitrarily be divided into three classes, all of which have importance for the purpose of design.

1. *Simple* models which might either represent a simple system or the gross behaviour of a complex system; for example, we might represent a wheel suspension of an automobile as a single mass connected to a spring and a viscous damper.

2. Models consisting of a small number of interconnected bodies. We are thinking of something more complex than a four bar linkage, but not a complete assembly like an internal combustion engine or a complete robot. We will refer to these as *moderately complex* mechanisms.
3. Models involving more complete assemblies, such as engines, complex robots and so forth. We will refer to these as *very complex*.

It should be emphasized that, depending on our purpose, *any* system can be represented as either simple, moderate or very complex.

The *software* used for simulating the behaviour of multibody systems can be divided into two groups:

- Multibody system analysis codes producing *exclusively numerical data*. Well-known examples are ADAMS and DADS [5]. These codes are suitable for simulation of the behaviour of rigid body systems of all three classes mentioned above.
- Multibody system analysis codes providing *explicit symbolic expressions* for the system equations. Well-known examples are AUTOLEV, NEWEUL and SD-FAST [5]. These codes are suitable for simulation of the behaviour of rigid body systems of the first and the second class above. The greatest advantage is the possibility of searching the parameter space for optimal design solutions without having to derive new equations of motion. Their disadvantage is that the complexity of the symbolic expressions increases rapidly when the number of interconnected bodies is increased.

2.2.1. Analysis Software of the Integrated System

The code used in the system described in this paper is called Sophia2 [9] and belongs to the second group above, since it produces explicit symbolic expressions for the system equations. Sophia2 is a set of routines, coded in Maple [10] by Martin Lesser at the Royal Institute of Technology in Stockholm, Sweden. It is named after Sophia Kovelefskia, who was professor of mechanics at the University of Stockholm at the beginning of this century. Sophia2 runs on all platforms that Maple runs on, i.e. Macintosh, PC and various workstations.

The formalism used in Sophia2 follows closely the one developed by Kane for the study of rigid body systems and explained in the book by Kane and Levinson [11]. The geometric language and notation used are based on Lesser [12].

Sophia2 is suitable for the analysis of moderately complex systems, especially when parameter studies are to be performed, since it provides explicit symbolic expressions for the system equations. When these system equations describing the behaviour of the multibody systems have been evaluated, the parameters of the system (masses, lengths, etc.) can be varied and the performance simulated for the different versions of the system.

An example of an input file for Sophia2 is given in Appendix B. The resulting equations can be studied using the solver of Maple or external solver.

2.3. Integration of Software for Design and Analysis of Multibody System

Integration of software can be of two kinds:

- *Open integration*, which usually means communication between different software using files. Program-specific formats are mainly used since standards for product data exchange, like the STandard for the Exchange of Product model data (STEP), are not well developed for multibody systems.

  Examples: I-DEAS ⇔ ADAMS (complete) and
  I-DEAS ⇔ DADS.

  Open integrations makes it possible to exchange an application program for another program, and also to manipulate data in a neutral (application–independent) environment.
- *Closed integration*, which means that the transfer of data is performed without user interaction and control.

  Examples: PRO/Engineer ⇔ DADS and
  I-DEAS ⇔ ADAMS (kinematics).

  This type of integration gives a user-friendly environment because the user interface is not changed when moving from one application to another. In the example with PRO/Engineer and DADS, the user will stay within the graphical user interface of PRO/Engineer while simulating the behaviour of the multibody system using DADS.

2.3.1. Open Integration of the Design and Analysis Software using a Neutral Database

The integrated system that has been developed and is described in this paper is an open system with a product database.

Requirements of this integrated system are:

1. it should be possible to combine several methods of analysis in the product design process with a minimum amount of human effort and with a minimum system response time;
2. it should be easy to include new methods and tools to the integrated design and analysis system, and

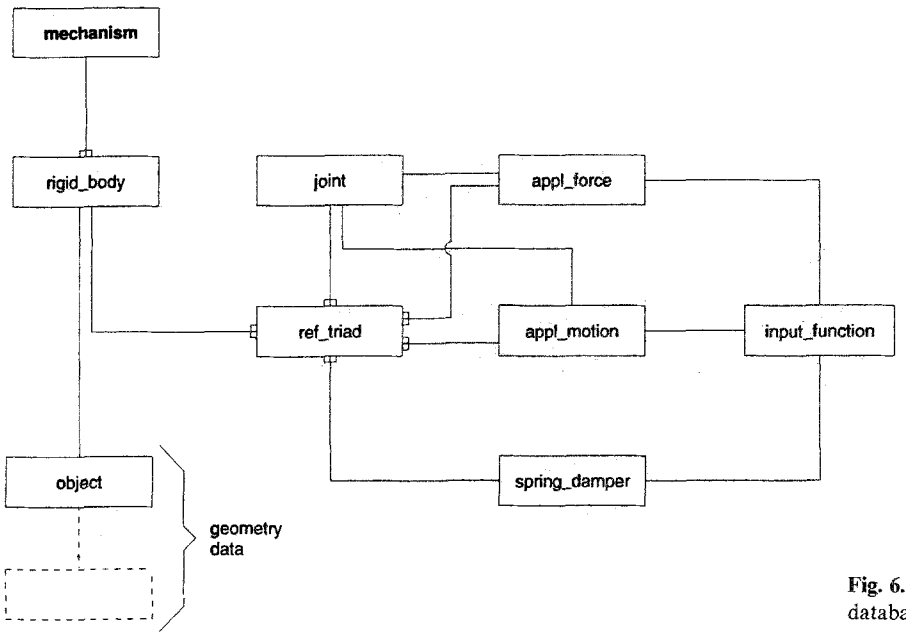


Fig. 6. Tables of multibody system database.

the system should be designed in a way that makes it supplier independent;

3. it should be possible to make design and analysis oriented queries to the system.

To satisfy the first requirement, it is important that data created in one application can be used in another. In the multibody system case, the properties of the solid models must be accessible for the multibody system analysts. To satisfy the second requirement, some sort of neutral data storage is needed. To satisfy the third requirement, a query language is needed.

One way to satisfy all three of the requirements above is to use an application-independent database, with a management system featuring a sophisticated query language. Our integrated system is centred around a product database, the structure of which is developed using a commercial database management system (DBMS) [13]. This DBMS supports distributed processing, which allows simultaneous users across a network to access the same database and examine the components of the multibody system. The structured query language (SQL) of the DBMS allows non-expert programmers to query and update the database. The query language also allow advanced filtering of data relevant for an application before the data are shipped to the application program.

The DBMS has been used to create a relational database structure in which data of multibody systems created in the CAD program are stored. This database consists of the tables shown in Fig. 6 and is described in detail in [14]. In the database structure, a mechanism is defined as a collection of rigid bodies that can move relative to each other, with joints that limit relative motion between pairs of bodies. A reference triad is a local coordinate system that consists of an origin and three axes, x, y and z. Each reference triad is attached to a rigid body at a selected location. The location and orientation of a joint are defined by a pair of reference triads associated with two rigid bodies. The location of a force is defined by a reference triad. Stiffnesses and dampings between rigid bodies are defined between pairs of reference triads. The motion of a mechanism is driven by loads. The loading may be gravity, applied forces, applied motion or initial conditions. The variations of applied forces and applied motions can be described by input functions. The geometry of each rigid body is defined in the object table and its subtables. The initial multibody system data are created using the CAD software that was described in section 2.1.1. Computer aided analysis of multibody systems is performed using the code Sophia2, which was described in section 2.1.2.

The interface codes that have been developed for the integrated system are the following:

- A program that reads the database dump created by CAD program and writes multibody system data to the product database. This program uses the specialized subroutines delivered together with the CAD software to read the database dump, and also embedded SQL (ESQL) to manipulate the product database. Since the multibody system is defined in a similar (physical) way in the CAD database and the product database, the data handling is straight-

forward. The data that are interesting for multibody system analysis are simply extracted from the CAD program and put into the tables of the product database.

- A program that reads multibody system data from the product database, manipulates the data and writes Sophia2 input. Here the rather physical description of the product database is transformed into mathematical expressions and subroutine calls, as shown in the example of Appendix B. In some cases the transformation from the product database is straightforward and in the other cases complicated; for example, the generalized coordinates declaration routine depends only on the joint table, while data from the joint, ref_triad and rigid_body tables are needed for the direction cosine matrix routines of Sophia2.

3. Using the System – An Example

In this section the use of the integrated system is demonstrated through an example. A multibody system is modeled in the CAD program, the multibody system data are stored in the product database, and the system equations are derived and time integrated. For the visualization a simplified data transfer has been adopted. The position results are transferred directly from Sophia2 to the CAD program, where an animated sequence is created.

To make the example easy to follow a simple system, a double pendulum, consisting of three bodies was chosen. This system is a simple planar chain system, but the analysis code can handle spatial systems with or without loops. The steps that are completed are shown in Fig. 7, where *CAD modeling* includes the four steps of Fig. 5.

In the mechanism design module of the CAD program, the multibody system model of Fig. 8 is created: it consists of two moving bodies, one ground and two revolute joints. The moving bodies are modeled as cylinders made of different materials. The longer one, which is connected to the ground through a revolute joint, has the density of aluminium (2800 kg/m^3) while the shorter, which is connected through a revolute joint to the aluminium cylinder, has the density of iron (7800 kg/m^3). The forces applied are caused by gravity in the positive y-direction of the coordinate system of the ground. These data of the multibody system are dumped in the tables of the temporary database created by the CAD program.

Mechanism storage refers to the transfer of multibody system data from the dump created by the CAD program to the independent product database. The dump created by the CAD program is read by execution of an interface routine which uses the specialized subroutines delivered together with the CAD software. The multibody system data are transferred to the tables of the product database using an interface routine which, when excuted, write data into databases using ESQL (Embedded SQL). The contents of the multibody system tables used in this example are given in Appendix A.

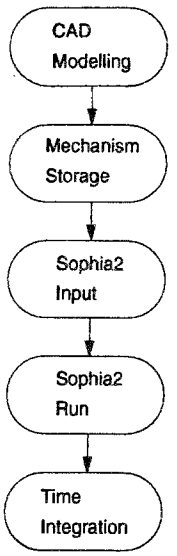


Fig. 7. Steps presented in the example.

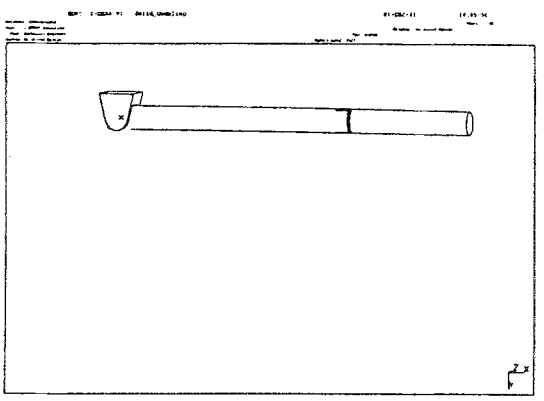


Fig. 8. CAD model of example system.

The Sophia2 input file is generated automatically by the interface code which reads multibody system data from the product database, manipulates it and writes Sophia2 input files. The input file is shown in Appendix B. Just the symbolic parameters of the multibody system are put into Sophia2 in order to make it possible to carry out parameter studies.

The resulting system expressions produced by Sophia2 are shown in Appendix C. These expressions

are put equal to zero in order to obtain the system differential equations. The current parameter values from the product database were substituted into the system equations and the system equations were solved numerically using the Maple routine, dsolve, which uses a Fehlberg fourth–fifth-order Runge–Kutta method.

The positions of the bodies were transferred to the CAD program where an animated sequence was created. Three of the configurations of the sequence are shown in Appendix D.

4. Discussion and Conclusions

An integrated system for design and analysis of multibody systems has been developed. This system consists of a commercially available CAD program and a multibody system analysis code developed at the Royal Institute of Technology in Stockholm. The design tool and analysis tool are integrated using an application-independent database. The database was developed using a relational DBMS, since its sophisticated query language makes database query and update easy. It was found that the formulation presented is very suitable for both the formation of system equations and for visualization of simulations.

The main gains of the work presented are the possibilities of using calculated component data like mass and moment of inertia from the CAD program, the automatic formulation of input files for Sophia2 and finally the visualization of simulation results. The interactive structured query language (ISQL) of the DBMS provides the possibility of examining the components of the multibody system during the design work and before any simulation is performed.

During the last ten years much work has been done on the specification of standards for product data exchange. This work has resulted in a standard, ISO 10 303, informally called STEP (STandard for the Exchange of Product data). So far, the standard for multibody system data is not complete, but when it is it will be used in our future work. The structure of our product database can easily be changed to make it accord with the STEP standard.

The fairly simple representation method of relational databases is not particularly well suited for complex engineering structures. This fact has driven the development of object-oriented database management systems (OODBMS). The recent development of the second generation of OODBMSs with query languages will make it possible for non-expert database programmers to use the benefits of object-oriented databases [15]. Future work will be aimed towards integrated systems with object-oriented databases.

Acknowledgments

The author would like to thank Martin Lesser of the Royal Institute of Technology, Stockholm, for providing Sophia2; also Lennart Karlsson, Peter Jeppsson and Greger Bergman of the Luleå University of Technology and Anders Lennartsson at the Royal Institute of Technology, Stockholm, for valuable discussions. The author also wishes to thank the Institute of CIM at Luleå University of Technology and the Swedish National Board for Industrial Development for financial support.

References

1. ARIES Conceptstation, Aries Technology Inc. Lowell, MA
2. SDRCI-DEAS VI, Structural Dynamics Research Corporation, 200 Eastman Drive, Milford, Ohio 45150–2789
3. Intergraph, Intergraph Corporation, Huntsville, Alabama 35894–001
4. PRO/Engineer, Parametric Technology Corporation, 128 Technology Drive, Waltham, MA 02154
5. Schielen, W. (Editor) (1989) Multibody Systems Handbook, Springer-Verlag, New York
6. Daberkow, A.; Kreuzer, E.; Leister, G.; Schielen, W. (1993) CAD modeling, multibody system formalisms and visualization – an integrated approach, advanced multibody system dynamics, W. Schielen (Editor), Kluwer Academic Publishers, Dordrecht, The Netherlands
7. Haug, E.J. (1993) Integrated tools and technologies for concurrent engineering of mechanical systems, Concurrent Engineering Tools and Technologies for Mechanical System Design, E.J. Haug (Editor), Springer-Verlag, Heidelberg
8. Mortenson, M.E. (1985) Geometric Modeling, Wiley, New York
9. Lesser, M. (1993) Sophia2 – Reference Manual, Department of Mechanics, Royal Institute of Technology, S-100 44 Stockholm, Sweden
10. MAPLE, Waterloo Maple Software, 160 Columbia Street West, Waterloo, Ontario, Canada N2L 3L3
11. Kane, T.R.; Levinson, D.A. (1985) Dynamics: Theory and Applications, McGraw-Hill, New York
12. Lesser, M. (1992) A geometrical intepretation of Kane's equations, Proc. Royal Society London, 436, 69–87.
13. INFORMIX, Informix Software Inc., 4100 Bohannon Drive, Menlo Park, CA 94025
14. Hardell, C.; Stensson, A.; Jeppsson, P. (1993) A relational database for general mechanical systems, presented at the NATO Advanced Study Institute on Computer Aided Analysis of Rigid and Flexible Mechanical Systems, Troia, Portugal, 27 June–10 July 1993. To be publlished in NATO ASI Series, Kluwer Academic Publishers, Dordrecht, The Netherlands
15. Bertino, E.; Negri, M.; Pelagatti, G.; Sbattella, L. (1992) Object-oriented query languages: the notion and the issues, IEEE Trans. Knowledge Data Engr., 4, 223–237

Appendix A: Database Tables used in the Example

mechanism	
mech_no	1
mech_descr	Double Pendulum
create_date	25-NOV-1993 16:04:35
solve_date	
model_mech_no	0
solver_id	0
id_set_no	1
md_user_flag	
workng_units	1
units_factrs_1	1.0000000000000
units_factrs_2	1.0000000000000
units_factrs_3	1.0000000000000
units_factrs_4	273.15000000000
grav_const	1.0000000000000
grav_vec_1	0.
grav_vec_2	9.8066501617432
grav_vec_3	0.

joint		
joint_no	1	2
joint_nam	JOINT1	JOINT2
ref_triad_1	1	3
ref_triad_2	2	4
joint_type	1	1
jt_user_flag		

rigid_body			
mech_no	1	1	1
rgd_body_no	1	2	3
rgd_body_nam	RB1_GROUND	RB2_AL_CYL	RB3_FE_CYL
object_id	1	2	3
cg_ref_triad	1001	1002	1003
ground_flag	1	0	0
rb_flag_1			
rb_flag_2			
rb_user_flag			
bcs_origin_1	0.	0.2500000000000	0.65000000000000
bcs_origin_2	0.	0.	0.
bcs_origin_3	0.	0.	0.
bcs_xpoint_1	1.0000000000000	1.2500000000000	1.65000000000000
bcs_xpoint_2	0.	0.	0.
bcs_xpoint_3	0.	0.	0.
bcs_zpoint_1	0.	0.2500000000000	0.65000000000000
bcs_zpoint_2	0.	0.	0.
bcs_zpoint_3	1.0000000000000	1.0000000000000	1.00000000000000
mass	1.3010000000000	2.5600000000000	4.20800000000000
inertias_1	0.0010000000000	7.6187244849280D-04	0.00100000000000
inertias_2	8.9373037917539D-04	0.0510000000000	0.02900000000000
inertias_3	0.0010000000000	0.0510000000000	0.02900000000000
inertias_4	0.	0.	0.
inertias_5	0.	0.	0.
inertias_6	0.	0.	0.

ref_triad				
ref_triad_no	1	2	3	4
ref_triad_nam	rf1	rf2	rf3	rf4
owner_body	1	2	2	3
rt_flag_1				
rt_user_flag				
location_1	0.	-0.2500000000000	0.2500000000000	-0.1500000000000
location_2	-0.3000000000000	0.	0.	0.
location_3	0.	0.	0.	0.
rtcs_xpoint_1	1.0000000000000	0.7500000000000	1.2500000000000	0.8500000000000
rtcs_xpoint_2	-0.3000000000000	0.	0.	0.
rtcs_xpoint_3	0.	0.	0.	0.
rtcs_zpoint_1	0.	-0.2500000000000	0.2500000000000	-0.1500000000000
rtcs_zpoint_2	-0.3000000000000	0.	0.	0.
rtcs_zpoint_3	1.0000000000000	1.0000000000000	1.0000000000000	1.0000000000000

ref_triad			
ref_triad_no	1001	1002	1003
ref_triad_nam	BODY1 CG TRIAD	BODY2 CG TRIAD	BODY3 CG TRIAD
owner_body	1	2	3
rt_flag_1			
rt_user_flag			
location_1	0.	0.	0.
location_2	-0.3000000000000	0.	0.
location_3	0.	0.	0.
rtcs_xpoint_1	1.0000000000000	1.0000000000000	1.0000000000000
rtcs_xpoint_2	-0.3000000000000	0.	0.
rtcs_xpoint_3	0.	0.	0.
rtcs_zpoint_1	0.	0.	0.
rtcs_zpoint_2	-0.3000000000000	0.	0.
rtcs_zpoint_3	1.0000000000000	1.0000000000000	1.0000000000000

Appendix B: Input File for Sophia2

```
# Declare generalised coordinates and speeds
declareList([q1,q2,u1,u2]);
# Formulate kinematic differential equations
kde:={q1t=u1,q2t=u2};
# Compute direction cosine matrices
chainSimpRot([[N,B1,3,q1],[B1,B2,3,q2]]);
# Set up Euclidian vectors describing the geometry of the system
# The word «star» indicates centre of mass positions where it is appended
rB1star:=Evector(L1OQ,0,0,B1);
rOP2:=Evector(L1,0,0,B1);
rP2Q2:=Evector(L2/2,0,0,B2);
rB2star:=EaddMixed(1,rOP2,1,rP2Q2,B1);
# Compute all needed velocities and angular velocities
vB1star:=Esimplify(express(diffFrameTime(rB1star,N),B1));
vB2star:=Esimplify(express(diffFrameTime(rB2star,N),B1));
wNB1:=Esimplify(express(angularVelocity(B1,N),B1));
wNB2:=Esimplify(express(angularVelocity(B2,N),B1));
# Substitute the explicit time derivatives of the generalised coordinates q_i
# with the generalised speeds u_i in the velocity expressions.
vB1star:=subs(kde,vB1star);
vB2star:=subs(kde,vB2star);
wNB1:=subs(kde,wNB1);
wNB2:=subs(kde,wNB2);
# Build velocity K-vector
vK:=buildKvec([vB1star,wNB1,vB2star,wNB2]);
# Extract tangent vectors from velocity information
extractPartials(vK,u,2);
# Compute linear inertia forces
ptB1:=express(linInertiaForce(mB1,vB1star,N),B1);
ptB2:=express(linInertiaForce(mB2,vB2star,N),B1);
# Put in moment of inertia about centre of mass
IB1star:=EinertiaDyad(iB11,iB12,iB13,0,0,0,B1);
IB2star:=EinertiaDyad(iB21,iB22,iB23,0,0,0,B1);
# Compute rotary inertia forces
htB1:=Esimplify(express(rotInertiaForce(wNB1,IB1star,B1,N),B1));
htB2:=Esimplify(express(rotInertiaForce(wNB2,IB2star,B2,N),B1));
# Build inertial force K vector
ptK:=buildKvec([ptB1,htB1,ptB2,htB2]);
ptK:=subs(kde,op(ptK));
# Put in applied forces
RB1star:=express(Evector(0,mB1*g,0,N),B1);
RB2star:=express(Evector(0,mB2*g,0,N),B1);
# Put in applied torques
TB1star:=express(Evector(0,0,0,N),B1);
TB2star:=express(Evector(0,0,0,N),B1);
# Build applied force K-vector
rK:=buildKvec([RB1star,TB1star,RB2star,TB2star]);
# Form Kane's equations
FormKane(ptK,rK,[vKP1,vKP2]);
```

Appendix C: System Equations Generated by Sophia2

System Equation 1:

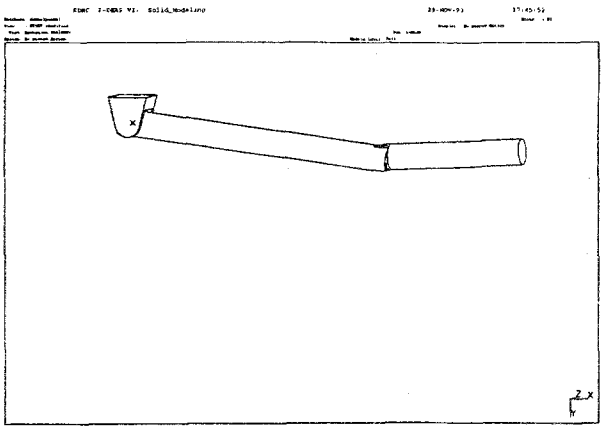
$$\frac{1}{4} L1^2\, mB1\, u1t + iB13\, u1t + \frac{1}{4} L2^2\, mB2\, u2t + \frac{1}{4} L2^2\, mB2\, u1t$$
$$- L1\, u1\, mB2 \sin(q2)\, u2\, L2 + mB2\, u1t\, L1^2$$
$$+ L1\, mB2\, u1t \cos(q2)\, L2 - \frac{1}{2} L1\, mB2 \sin(q2)\, u2^2\, L2$$
$$+ \frac{1}{2} L1\, mB2 \cos(q2)\, u2t\, L2 + iB23\, u1t + iB23\, u2t$$
$$- \frac{1}{2} L1 \cos(q1)\, mB1\, g + \frac{1}{2} \sin(q2)\, L2 \sin(q1)\, mB2\, g$$
$$- \cos(q1)\, mB2\, g\, L1 - \frac{1}{2} \cos(q1)\, mB2\, g \cos(q2)\, L2$$

System Equation 2:

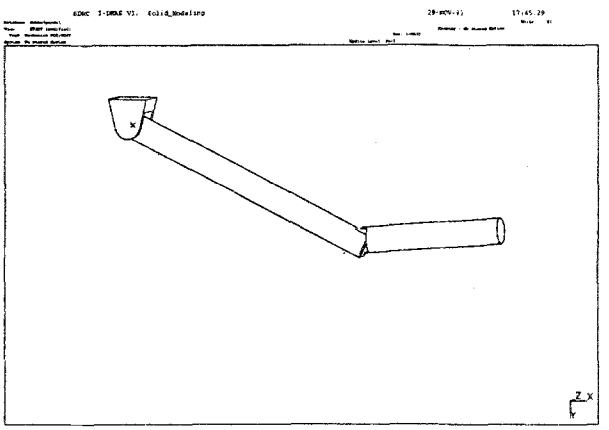
$$\frac{1}{4} L2^2\, mB2\, u2t + \frac{1}{4} L2^2\, mB2\, u1t + \frac{1}{2} \sin(q2)\, L2\, u1^2\, mB2\, L1$$
$$+ \frac{1}{2} L1\, mB2\, u1t \cos(q2)\, L2 + iB23\, u1t + iB23\, u2t$$
$$+ \frac{1}{2} \sin(q2)\, L2 \sin(q1)\, mB2\, g$$
$$- \frac{1}{2} \cos(q1)\, mB2\, g \cos(q2)\, L2$$

Appendix D: Configurations of Example System

Body positions at t=0.1s



Body positions at t=0.2 s



Body positions at t=0.3 s