Computers in Engineering – 1994
Volume 2
ASME 1994

CONCEPT, DEVELOPMENT AND IMPLEMENTATION OF DAMOS-C: THE OBJECT ORIENTED APPROACH TO MULTIBODY SYSTEMS

Andreas Daberkow and Werner Schiehlen
Institute B of Mechanics
University of Stuttgart
Stuttgart, Germany

ABSTRACT

The increasing use of computer aided design (CAD) and multibody dynamics software in engineering design and analysis requires an efficient communication management between different codes. As product cycle times are shrinking and companies are under competitive pressure, an automation of the model data exchange between CAD and multibody dynamics is urgently demanded.

In this paper the concept, development and implementation of the independent, object-oriented multibody modeling kernel DAMOS-C is presented. According to the different multibody formalisms, computer codes and applications in vehicle, machine and robot dynamics a general description and organization of multibody system data is introduced. A general data model is developed to fulfill the needs of a modular simulation system based upon numerical and symbolical formalisms. With respect to existing CAD interfaces, different solid model construction methods and visualization procedures, multibody system classes and methods are implemented in the multibody modeling kernel. The extreme versatility of this modeling kernel is shown by an integration in a commercially available CAD-system and by application to a conversion tool which processes input data for different multibody software codes.

1 INTRODUCTION

The integration of computer aided modeling and dynamic analysis in the product development process is shown in Fig. 1. Concurrent engineering is characterized by integrated communication structures, data handling and software standards. Computer aided design and dynamic analysis of mechanical systems or structural components of mechanical systems are key technologies in this development process.

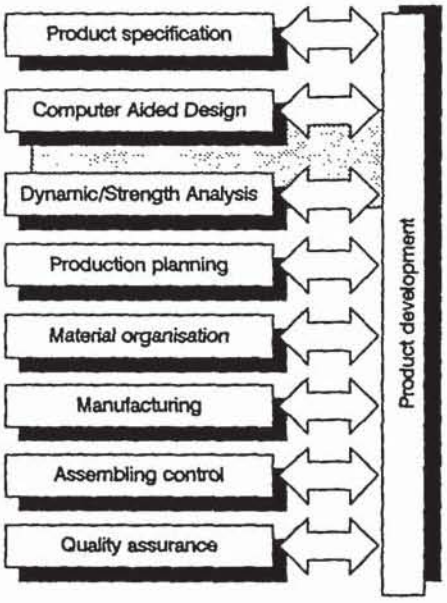

FIG. 1 PRODUCT DEVELOPMENT PROCESS BY CONCURRENT ENGINEERING PRINCIPLES

The modeling of a mechanical system by the method of multibody systems is characterized by a composition of rig-

id bodies, joints, springs, dampers, and servomotors, see Fig. 2. Joints with different properties connecting the various bodies constrain the motion of system's bodies, determine the degree of freedom of the multibody system, and result in constraint forces and torques. Force elements like springs, dampers, and servomotors acting in discrete node points result in applied forces and torques on the rigid bodies.

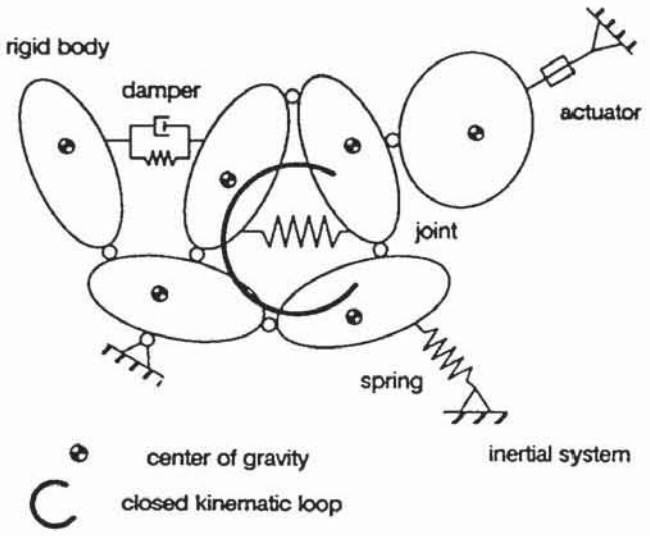

FIG. 2 MULTIBODY SYSTEM

A classification of multibody formalisms may be obtained according to the principles of mechanics applied, the number of equations to determine the system's motion, the system's topology, or the procedure the system's equations are generated, i.e. either in numerical or symbolical representation. A considerable number of computer codes has been developed for numerical equation generation, well known examples are ADAMS (Orlandea, 1973), and DADS (Haug, 1989). Computer programs like SD-FAST (Rosenthal and Sherman, 1986) and NEWEUL (Kreuzer,1979), (Kreuzer and Schiehlen, 1985), provide explicit analytical expressions for the system equations, including numerical or symbolical values for the parameters. A survey on different formalisms and computer codes in multibody dynamics can be found in Schiehlen (1990), further computational aspects are discussed in Roberson and Schwertassek (1988).

Nowadays two-dimensional CAD-systems are widely embedded in the industrial design and construction process, while a throughout application of three-dimensional CAD-systems is more rare. The description of an analytically and topologically complete model, an interference detection, tool path geometry or calculation of surface and volume properties, respectively, is closely related to the geometric representation of solid models, see Mortenson (1985) and Pahl (1990). Most of these solid modelers are based on one of the two well-known methods, constructive solid geometry (CSG) or boundary representation (B-Rep). The boundary representation model contains explicitly a topological consistent representation of the faces, edges and vertices bounding a solid. A large group of solid modeling CAD-systems uses the boundary representation, e.g by integrating the geometric modeler PARASOLID (1990).

Two approaches for the coupling of solid modelers with multibody simulation software are realized for the numerical computer code ADAMS. The graphic modeling tool P/Mechanism, included in the finite element design tool PATRAN (1990), supports the graphical modeling and the postprocessing of simulation results with ADAMS. A further interface to ADAMS is provided by the CAD system ARIES (1990). Conceptually, these approaches calculate the mass properties from the solid model designed in the CAD system. These interfaces are supplied with additional mechanical properties, such as the locations of applied forces, torques and joint attachment points. However, approaches independent of CAD-3D-systems have some advantages, see Thatch and Mykleburst (1988), and result in program packages like RASNA (Hollar and Rosenthal, 1991).

Further investigations have been made to improve the automated exchange of design data between different CAD-systems, e.g. by IGES or STEP. STEP (STandard for the Exchange of Product model data) endeavors the evaluation of a generalized design data model including the coupling to analysis tools for finite element and kinematic analysis (Weber, 1988). At present, there isn't any STEP product model available for dynamic analysis. A first effort to develop a vendor independent generalized data model for multibody systems including symbolical parameters and a preprocessing with CAD-systems is described by Otter, Hocke, Daberkow and Leister (1993).

2 MOTIVATION AND BASIC CONCEPT

A system dynamics analysis requires the basic parameters as mass, center of gravity, and moments of inertia without considering the geometry model and modeling method of the CAD system used. A high degree of modularization demands an exchange of complete or single object data of a multibody system. Additionally, passing simulation results of the dynamic analysis to a CAD or graphics system is necessary without concern about the geometry model representation. Therefore, a general interface to multibody computer codes is required serving as a compatible and comfortable post processor and takes different algorithms and implementations of multibody computer codes

into account. Commercially available multibody modeling software tools within CAD-systems are dedicated to one particular multibody dynamics computer code. Often, no options are supplied for a parametric multibody system description, or the modeling is restricted to either robot, mechanism or vehicle dynamics, respectively. The variety of systems each with different model data and the growing problems in the data exchange, see Fig. 3, counter the need to accelerate the production of cheaper and more reliable products.

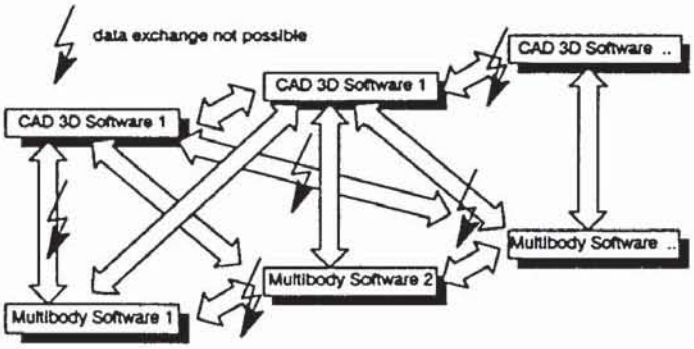

FIG. 3 CURRENT PROBLEM IN AUTOMATED MODEL CONVERSION

Consequently, this leads to the following key concept:

- Comprise the necessary data describing a multibody model for the different multibody programs.
- Examine the different geometry models of CAD-systems for solids and extract the relevant data for multibody systems.
- Define a geometry model for the representation of multibody elements.
- Construct a software interface for a system-independent modeling of multibody systems. Design data types and operations from these objectives.

3 BASIC MULTIBODY SYSTEM ELEMENTS AND PROPERTIES

The kinematics of spatial multibody systems are described by body-fixed reference frames, determining the rigid body location and orientation and the position of joint and force elements. Therefore, the main task is the definition of location and orientation of reference frames. In the following, the basic multibody system elements are described with respect to the Cartesian coordinates in numerical implementation on the one hand and the Lagrangian generalized or minimal coordinates in symbolical implementation on the other hand.

3.1 Cartesian Coordinate Approach

To derive the equations of motions for multibody systems by means of cartesian coordinates, the presentation follows the approach described by Haug (1989). First of all a set of cartesian coordinates, e.g.

$$\mathbf{x}_i = \left[r_{ix}\ r_{iy}\ r_{iz}\ \phi_i\ \theta_i\ \psi_i\right]^T . \tag{1}$$

is chosen for each body. In (1) the location is described by the 3×1 column vector $\mathbf{r}_i$ and the orientation is specified by the Euler angles $\phi_i\ \theta_i\ \psi_i$ with respect to the global inertial frame.

3.1.1 Kinematics. In practice, constraints restrict the relative position and orientation between the bodies. A multibody system with nb bodies and f degrees of freedom involves $q = nb - f$ constraints due to joints represented by two frames.

A body-fixed frame K_k on body i is defined by its vector ${}_i\mathbf{r}_{ik}$ and its rotation tensor $\mathbf{S}_{ik} = \left[{}_i\mathbf{e}_{k,1}\ {}_i\mathbf{e}_{k,2}\ {}_i\mathbf{e}_{k,3}\right]$ relative to the reference frame K_i of body i, Fig. 4.

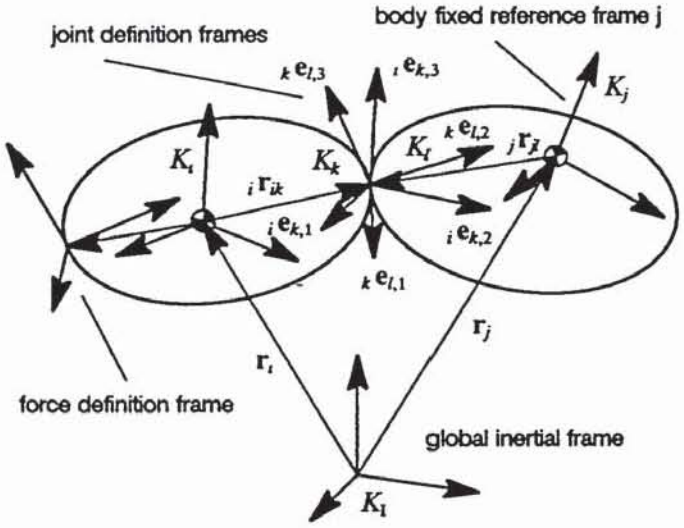

FIG. 4 BODY-FIXED REFERENCE FRAMES

Libraries for the constraint equations of joints are developed from basic conditions for the parallelism and orthogonality of the corresponding joint definition frame unit vectors. For a multibody system with nj joints, each with a $r \times 1$ constraint vector equation, the combined $q \times 1$ constraint vector equation reads as

$$\Phi(\mathbf{r}_i, \mathbf{S}_i, \mathbf{r}_j, \mathbf{S}_j) = \Phi\ (\mathbf{x}, t)\ , \tag{2}$$

where $\mathbf{S}_i$ and $\mathbf{S}_j$ denote the 3×3 rotation matrices of the body fixed reference frame of body i and j, respectively, related to the inertial system. Thus, the joint description library yields the overall constraints of the multibody system. From these basic information, the translational and angular velocity as well as the accelerations of each body is determined by differentiation.

3.1.2 Dynamics. The $6 \cdot nb$ nonlinear equations of motion for spatial dynamics in the Cartesian generalized coordinates approach are derived from Newton's and Euler's equations as

$$\overline{\mathbf{M}}\ \ddot{\mathbf{x}} + \overline{\mathbf{k}} + \Phi_{\mathbf{x}}^T \boldsymbol{\lambda} = \mathbf{f}^A \quad , \tag{3}$$

where

$$\overline{\mathbf{M}} = diag\ (m_1\mathbf{E}\ , \ldots\ , m_{nb}\mathbf{E}\ ,\ \mathbf{I}_1\ , \ldots\ \mathbf{I}_{nb}\ ) \tag{4}$$

is the $6 \cdot nb \times 6 \cdot nb$ inertia matrix with the mass m_i of each body i and the 3×3 inertia tensors of each body defined with respect to the body-fixed reference frame located in the center of gravity, $\boldsymbol{\lambda}$ is the $q \times 1$ vector of Lagrangian multipliers, $\overline{\mathbf{k}}$ the $6 \cdot nb \times 1$ vector of centrifugal forces and $\mathbf{f}^A$ denotes the $6 \cdot nb \times 1$ vector of the applied forces and torques, which in reality are complicated functions of $\mathbf{x}$ and further external signals.

Eqs. (2) and (3) also denoted as Lagrangian equations of the first kind, represent a system of mixed second order differential-algebraic equations.

3.2 Lagrangian generalized coordinate approach

Another approach to derive equations of motion of multibody systems is based on the choice of a set of minimal coordinates. A detailed description of a formalism to generate dynamics equations by means of Lagrangian coordinates can be found in Schiehlen (1986). Depending on the number of degrees of freedom f of the multibody system a set of state variables is determined from coordinates representing relative motions in the system.

3.2.1 Kinematics. For a chain or tree structure topology in a multibody system a $f \times 1$ vector of generalized coordinates $\mathbf{y}$ may be introduced summarizing the relative joint coordinates between pairs of rigid bodies. Assuming that the relative motion of rigid bodies is described by the relative position and orientation of joint definition frames K_k and K_l, the position vector

$$_k\mathbf{r}_{kl} = {}_k\mathbf{r}_{kl}\ (\mathbf{y}) \tag{5}$$

and the rotation tensor

$$\mathbf{S}_{kl} = \mathbf{S}_{kl}\ (\mathbf{y}) = \mathbf{S}_{kl}\ (\alpha_{kl}, \beta_{kl}, \gamma_{kl}) \tag{6}$$

are easily written down. The rotation tensor (6) is given by elementary rotations about consecutive joint definition frame axes with angles $\alpha_{kl}, \beta_{kl}, \gamma_{kl}$.

From these basic relations, the inertial location of the jth body is obtained recursively by

$$\mathbf{r}_j(\mathbf{y}) = \mathbf{r}_i(\mathbf{y}) + \mathbf{S}_i(\mathbf{y})\ {}_i\mathbf{r}_{ik} + \mathbf{S}_i(\mathbf{y})\ \mathbf{S}_{ik}\ {}_k\mathbf{r}_{kl}(\mathbf{y}) - \mathbf{S}_j(\mathbf{y})\ {}_j\mathbf{r}_{jl}\ ,$$

$$\mathbf{S}_j(\mathbf{y}) = \mathbf{S}_i(\mathbf{y})\ \mathbf{S}_{ik}\ \mathbf{S}_{kl}(\mathbf{y})\ \mathbf{S}_{jl}^T\ . \tag{7}$$

The translational and angular velocity $\mathbf{v}_j$ and $\boldsymbol{\omega}_j$ as well as the translational and rotational acceleration $\mathbf{a}_j$ and $\boldsymbol{\alpha}_j$ of each body is again determined by differentiation.

A systematic choice of the generalized coordinates vector $\mathbf{y}$ needs further information in case of closed kinematic loops, compare Fig. 1. Then the number of degrees of freedom is smaller than the number of relative joint coordinates. The problem to select a proper set of independent generalized coordinates may be solved numerically, see e.g. Wehage and Haug (1982). The method described by Leister and Bestle (1992) uses independent generalized coordinates in a linear combination of the dependent relative joint coordinates. This linear combination and the independent generalized coordinates are specified automatically during the numerical simulation.

3.2.2 Dynamics. The nonlinear equations of motion for a multibody system based on the Lagrangian coordinate approach are given by

$$\mathbf{M}\mathbf{y} + \mathbf{k} = \mathbf{q}^e\ , \tag{8}$$

$$\mathbf{g}(\mathbf{y}, \mathbf{h}, t) = \mathbf{0}\ , \tag{9}$$

where

$$\mathbf{M} = \sum_{i=1}^{nb} (\mathbf{J}_{Ti}^T\ m_i\ \mathbf{J}_{Ti} + \mathbf{J}_{Ri}^T\ \mathbf{I}_i\ \mathbf{J}_{Ri}) \tag{10}$$

is the $f \times f$ mass matrix with the mass m_i and the inertia tensor $\mathbf{I}_i$ of each body, $\mathbf{J}_{Ti}$ and $\mathbf{J}_{Ri}$ are the $f \times 3$ translational and rotational Jacobians resulting from the differentiation of the position vectors and orientation matrices. In (8), $\mathbf{k}$ denotes the $f \times 1$ vector of the centrifugal and

coriolis forces and $\mathbf{q}^e$ is the $f \times 1$ vector of the applied forces and torques. In case of closed loop systems, the Jacobians in (10) are functions of the joint coordinate vector $\mathbf{h}$ and its partial derivatives. Here, besides the set of ordinary differential equations the implicit algebraic equation (9) has to be solved numerically for each given state $\mathbf{y}$ and $\dot{\mathbf{y}}$. In case of tree structured systems, equation (9) vanishes and the vector of generalized coordinates $\mathbf{y}$ is equal to the vector $\mathbf{h}$ representing the degrees of freedom.

3.3 Summary and classification of basic multibody model data

A dynamic simulation environment for multibody systems represents a large, sophisticated software system. Therefore, an important preceding step is the development of an abstract data model on a conceptual level. A unified data model has to serve both coordinate approaches in the same context. For symbolical as well as numerical formalisms a generalized classification of multibody systems relies upon the basic modeling elements *frame*, *body*, *joint*, and *force*.

To specify the joint and force definition frames, the position and orientation of each frame is determined with respect to the reference frame of the body, which has its origin in the mass center of the body. This position vector ${}_i\mathbf{r}_k$ between the reference frame of body i and frame k is additionally supplied by symbolical variables for each vector coordinate.

According to the three degrees of freedom of rotation an angle representation of the rotation tensor is chosen to describe the frame orientation. By supplying the rotation sequence information and symbolic variables for the rotation angles, a full parametrization is achieved for later parameter studies of joint and force element attachment points.

The joint definition frames identify the connection between two bodies by one joint. With respect to the joint type, the directions of translation and axes of rotation yield the characteristic relative joint position vector (5) and joint rotation tensor (6). Additionally, an initial offset of the joint coordinates is supplied as initial values. Figure 5 shows the revolute and translational joint definition frames of a joint library as well as the joint position vector and rotation tensor. From this information, the implicit constraint equation (2) of the Cartesian coordinate approach and an initial set of Cartesian coordinates is determined. For tree-structured multibody systems and the Lagrangian coordinate approach, the explicit constraint

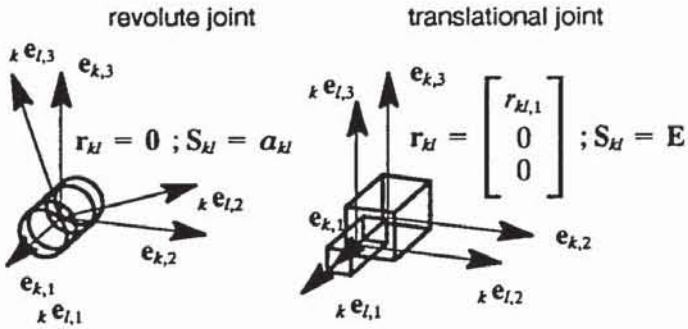

FIG. 5 JOINT DEFINITION FRAMES WITH POSITION VECTOR AND ROTATION TENSOR

formulation in the position vector $\mathbf{r}_j$ and orientation matrix $\mathbf{S}_j$ in (7) yields the location of each body.

For each body, the mass and the inertia tensor is needed. A time variant representation of the inertia tensor is obtained by

$$\mathbf{I}_i = \mathbf{S}_i \, {}_i\mathbf{I}_i \, \mathbf{S}_i^T . \tag{11}$$

The constant elements of the inertia tensor ${}_i\mathbf{I}_i$ with respect to the body fixed reference frame and the mass m_i again are supplied by a list of symbols in order to achieve a full parametrization.

The force definition frames for internal and external force elements serve to determine the actual lengths and velocities of spring, damper and actuator elements. Symbolic variables for force actuators support parameter studies of different force characteristics. In case of a general force law the desired force characteristics have to be supplied by the user as a function of the body's location, velocity and acceleration. A classification of force and torque laws in the Lagrangian coordinate approach is given by Schiehlen (1986).

4 GEOMETRIC MODELING OF SPATIAL BODIES IN CAD-SYSTEMS

The computer aided interactive construction and modification of spatial objects in industrial applications is characterized by different modeling methods. In practice, geometric modeling systems offer a graphic user interface, allowing an interactive design in a quasi threedimensional computer graphics workspace. With respect to the needs in multibody dynamics, the schemes and methods of the dominating constructive solid geometry and boundary representations are considered.

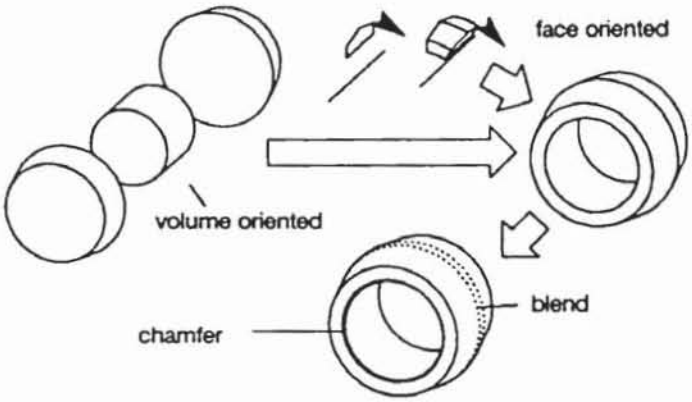

FIG. 6 GEOMETRIC ENTITIES AND MODELING TECHNIQUES

4.1 Geometry models

The basic process in solid model construction consists of the transformation of a desired shape and function in a spatial object composed of the geometric entities volume, face, edge, and vertex, Fig. 6. A first rough shape is created by combining volume-oriented or face-oriented techniques. In the first case, from predefined primitive objects like prism, cylinder or sphere with an internal Constructive Solid Geometry (CSG) or Boundary Representation (B-Rep), new shapes are constructed by Boolean operations. In the other case, the required face size and shape serves to create B-Rep solids by translational or rotational sweep operations. Such a first rough shape is often sufficient for the mass property calculation in a dynamic simulation. A final precise and detailed shape is achieved by local chamfer and blend operations, Fig. 6.

The Boolean combination of two or more primitive objects to a new solid object is the main characteristics of a CSG, Fig. 7. For two-dimensional projections of the CSG

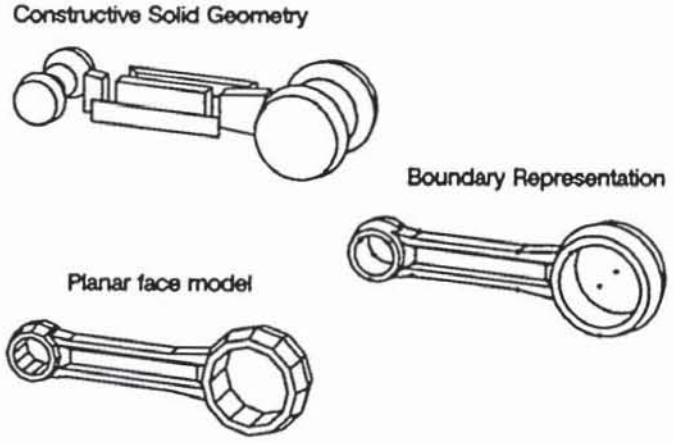

FIG. 7 CSG, B-REP AND PLANAR FACE MODEL

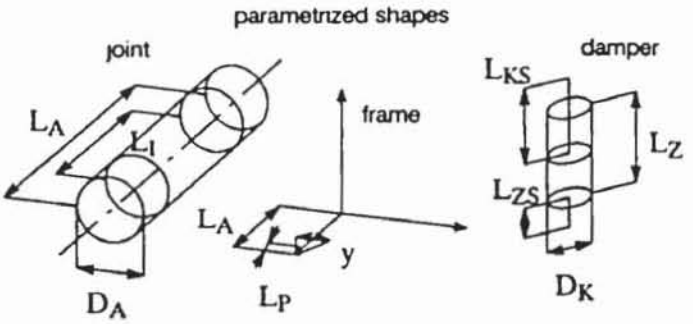

FIG. 8 PARAMETRIZED SHAPES

model, an equivalent wire or face model has to be derived from the binary tree of the primitives and their transformations.

The B-Rep model allows a Boolean combination of primitive objects, too. Each primitive object and the actual modeling state is described by a complete spatial boundary, of which the topological validity may be checked by application of the Euler operators to the enclosing faces, edges and vertices (Braid, 1974), see Fig. 7. The solid modeling tool PARASOLID uses a Boundary Representation and is commercially available in many CAD 3D-systems.

Besides the geometry models described above further models exist, which do not necessarily have volume properties. A simple *planar face model*, see Fig. 7, as a special case of the B-Rep, serves as a geometry model which is suitable for high-speed 3D- visualization, see Schiehlen and Daberkow (1988). Moreover, this model is implemented in graphic standards like PHIGS (Brown, 1985). A property of a solid can be derived from a face normal specifying the inner and outer parts of an object, while the coincidence of the vertices of adjoining faces is not guaranteed. The geometric modeling by *parametrized shapes* is appropriate for geometric objects, whose shape is uniquely defined by a restricted number of parameters. Examples of parametrized shapes with an equivalent wire representation are shown in Fig. 8.

4.2 Calculation of mass properties

The mass property calculation methods depend on the solid construction method. For the global properties volume, surface area, moment of inertia, and center of gravity, integral relations like

$$I = \int_{Solid} f^V dV \tag{12}$$

have to be evaluated (Mortenson, 1985), where $f^V = f^V(x,y,z)$ denotes a scalar property function. While

constructive solid geometry suggests the calculation of mass properties by the following recursively applied formulas

$$\int_{Solid1 \cup Solid2} f^V dV = \int_{Solid1} f^V dV + \int_{Solid1} f^V dV - \int_{Solid1 \cap Solid2} f^V dV,$$

$$\int_{Solid1 - Solid2} f^V dV = \int_{Solid1} f^V dV - \int_{Solid1 \cap Solid2} f^V dV, \quad (13)$$

where $\cup$ denotes a Boolean addition, $\cap$ a Boolean intersection and $-$ a Boolean subtraction. Boundary representations allow the evaluation via surface integrals. From the Gauss theorem it follows that (Mortenson, 1985)

$$\int_{Solid} f^V dV = \int_{Solid} div\ \mathbf{g}^V dV = \sum_{m=1}^{nf} \mathbf{g}^V \mathbf{n}_m F_m, \quad (14)$$

where F_m denotes the enclosing $m-th$ face of the solid with nf faces and unit normal vectors $\mathbf{n}_m$. In PARASOLID, the module *masspr* calculates the mass properties from the input of one or more solid objects, each supplied with a physical or unit density attribute. As a result one obtains the total surface, the total volume, the total mass of the objects, the center of gravity of the objects, and the 3×3 inertia tensor with respect to the center of gravity and axes parallel to the global CAD 3D inertial frame. Finally, the examination of different geometry models yield the following results:

- The results of the mass property calculation for multibody computer codes are not dependent on the model geometry (CSG or B-Rep).
- The results can be related directly with the input entities needed for the multibody modeling element *body*. From the center of gravity, the position vector $\mathbf{r}_i$ is determined automatically for an initial position of the body fixed reference frame, see Fig. 4. Choosing parallel axes of the reference frame to the global CAD 3D inertial frame, the components of the inertia tensor ${}_i\mathbf{I}_i$ and the mass m_i are calculated from PARASOLID.
- A planar face model derived from the geometric entities of the solid body yield the graphic data for the description of the body's shape necessary for visualization.
- The parametrized shapes are well suited to serve as a geometry model for multibody modeling elements like frame, joint and force.

5 DEVELOPMENT OF THE OBJECT-ORIENTED DATA MODEL

From the preceding sections it follows that an automated modeling and simulation requires a unique common description of multibody and CAD 3D modeling elements. The object-oriented data model, conceptually defined by Otter et al. (1993) represents nearly exclusively method-independent and nonredundant data to describe a multibody system.

5.1 Object-oriented software technique

Traditional concepts in multibody software are based on a sequential flow of program functions, see Fig 9. This functional approach has several drawbacks, as modifications and extensions of data and functions are difficult to perform. With the use of object-oriented software techniques, the requirements of reliable, extendable and interactive programs are satisfied. The uniform idea is a software architecture focussing on the physical data and their relationship rather than the program flow (Meyer, 1988).

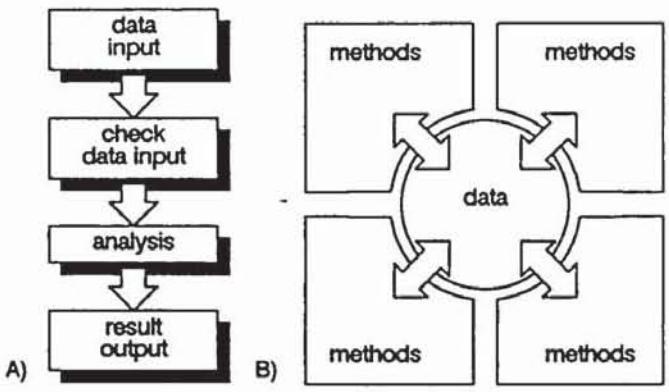

FIG. 9 TRADITIONAL (A) AND OBJECT-ORIENTED (B) DESIGN APPROACH

For the multibody system data model follows, that classes have to be defined for the elements *frame*, *body*, *joint*, and *force* and additional operations valid for these classes. For the proposed CAD 3D integration, the data model for rigid bodies is designed for the CAD 3D volume property calculation results and includes data for multibody formalisms in numerical and symbolical implementation.

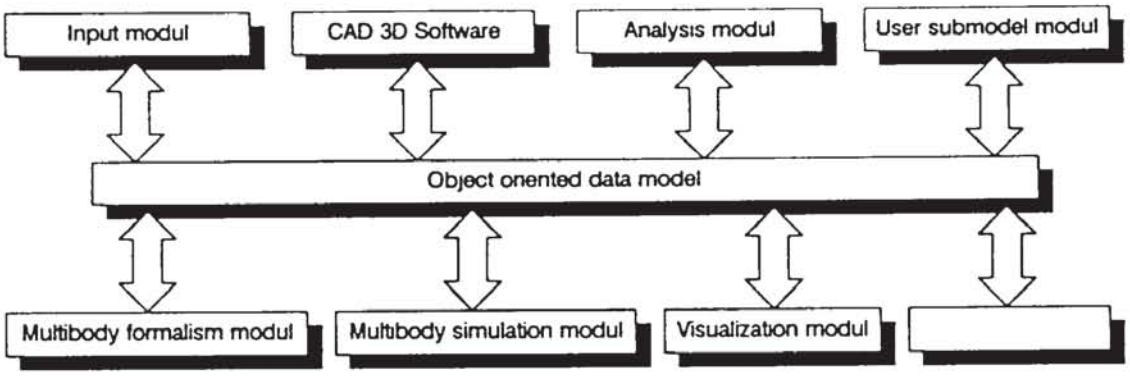

FIG. 10 MODULES IN AN OBJECT-ORIENTED MULTIBODY SYSTEM DATA MODEL

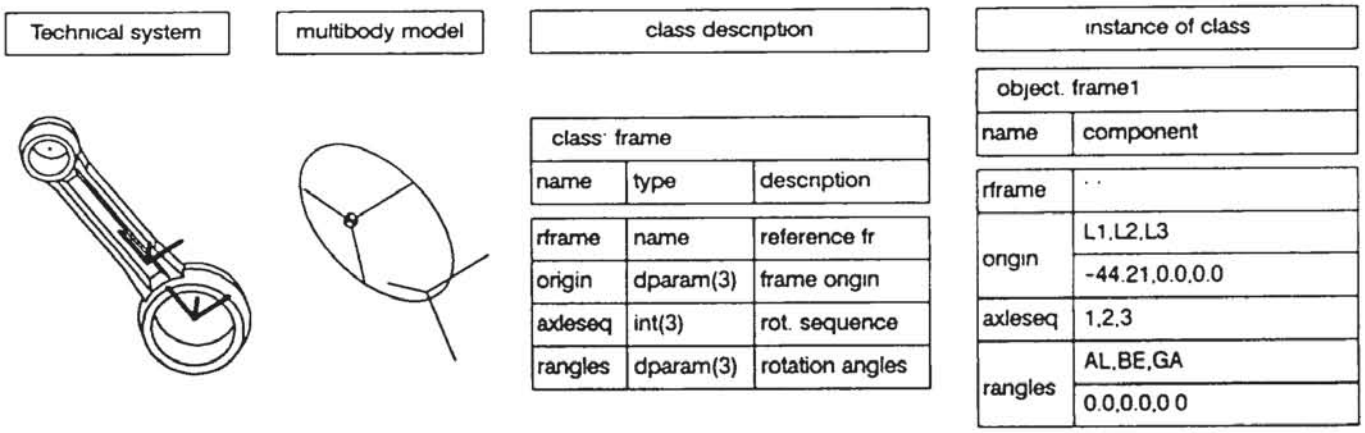

FIG. 11 OBJECT OF CLASS *FRAME*

Figure 10 shows the general data communication flow of modules in the object-oriented multibody approach. The modular design is most important concerning the solid model preprocessing. While one or several bodies of a multibody system are CAD 3D constructed solid models whose properties have been calculated, other kinematic and kinetic data may be preprocessed by means of user submoduls.

A strict separation of time invariant and time variant data allows the modeling and preprocessing of solid bodies with joint and force definition frame data from CAD 3D systems, while time variant interactions are comprised in joint and force definitions. A detailed description including numerical methods and user defined submodels is found in Otter et al. (1993).

5.2 Basic multibody classes and operations

The first step in object-oriented software design is the definition and implementation of abstract data types and operations, see Meyer (1988). By the definition of *classes*, a representation scheme (*type*) of properties of objects is determined. According to the characteristics of the class, each created *instance* (*object*) of a class is given by its *components*. For multibody modeling elements, the classes *frame*, *body*, *joint*, *force*, *interact*, *global*, and *param* are defined. An object of class *part* e.g. serves as a superior node for objects of class *frame*, Fig. 11, and *body*, Fig. 12, and comprises all time-invariant data in a multibody model.

Figure 11 shows the components of an object of class *frame* which are determined from the mechanical and mathematical frame properties. Besides the name tag of the corresponding reference frame, the origin and orientation is described by a component *dparam* comprising symbolical and numerical values. By the nonredundant specification of the rotation sequence, a variety of rotation descriptions is supported including Euler and Cardan angles. At least one object of class *frame* needs a definition with respect to the unique body fixed reference frame. A symbolical and numerical description is included for the class *body*, too. Figure 12 shows that the components of the inertia tensor and mass are supplied by their numerical values. A location of the center of gravity dif-

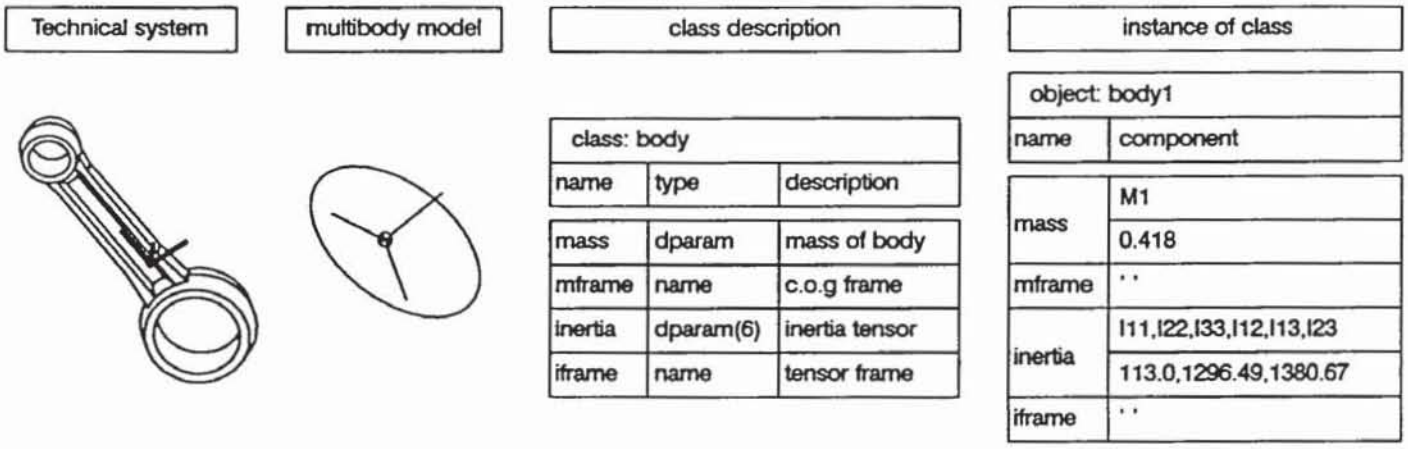

FIG. 12 OBJECT OF CLASS *BODY*

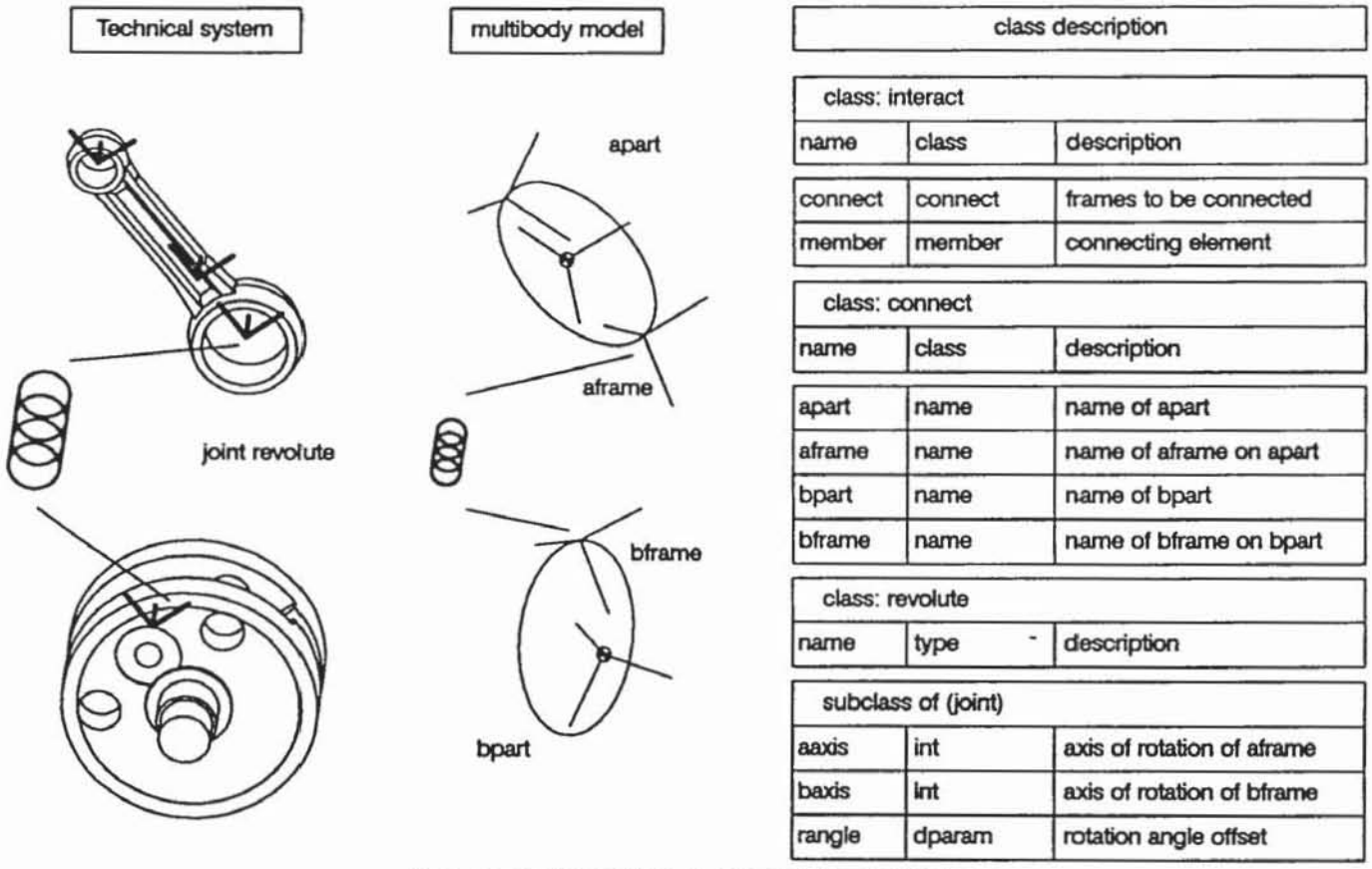

FIG. 13 COMPONENTS OF CLASS *INTERACT*

ferent from the body fixed reference frame is taken into consideration by a reference to an equivalent object of class *frame*.

Coupling elements of a multibody system are collected in a class *interact*. Interactions are valid between two objects of class *frame* on different objects of class *part*, either caused by an object of class *joint* or objects of class *force*. Again, the object components are designed according to the mechanical and mathematical joint and force properties. Different joint types are considered by equivalent subclasses, which *inherit* the properties of their superior class *joint* and *force*, respectively. Figure 13 shows the components of the classes *interact*, *connect*, and *revolute*.

A variety of rotation definitions is obvious from the *revolute* class. The rotation axes are chosen from the related

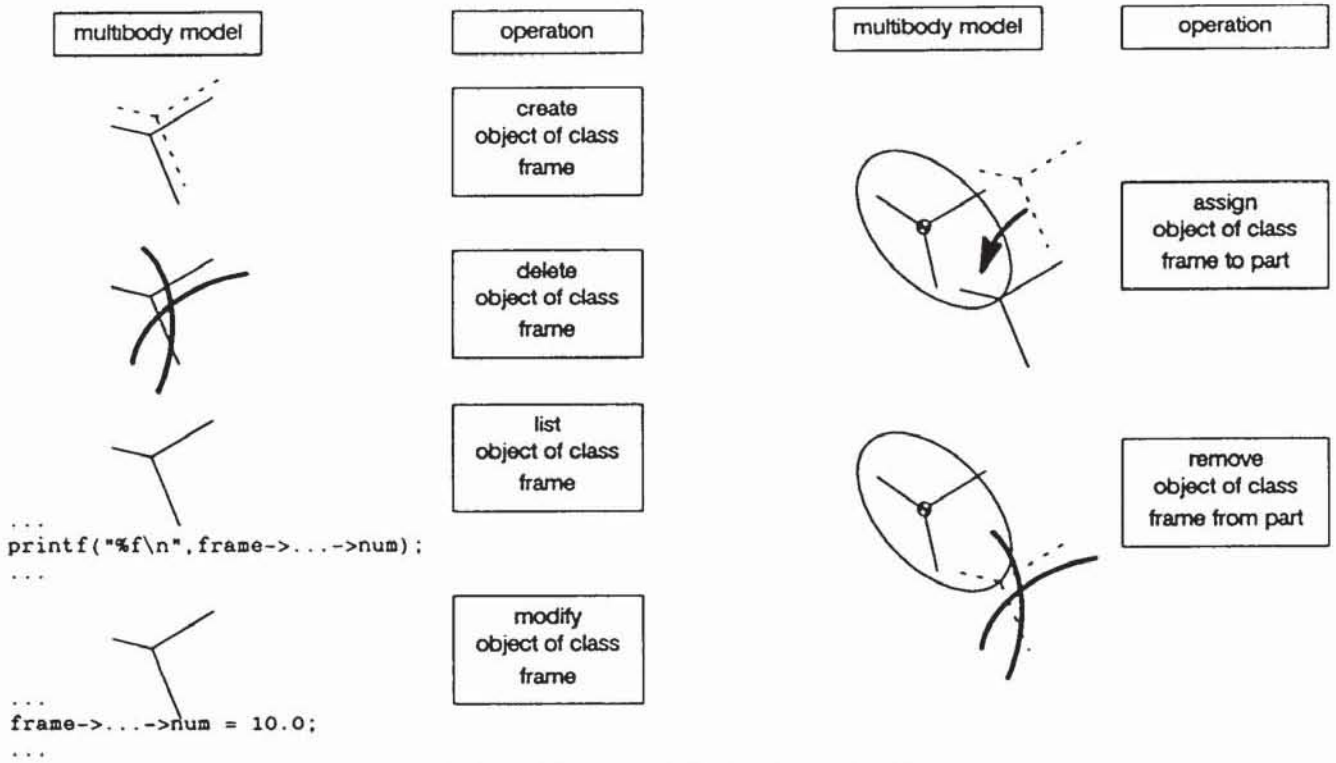

FIG. 14 OBJECT-ORIENTED OPERATIONS FOR CLASS *FRAME*

objects of class *frame*, the initial joint orientation is provided by the component *rangle*.

Further important classes like *force* and a superior class *mbs* containing all objects of a multibody system are also defined. In the class *global*, global properties of the multibody system are defined like direction and magnitude of the gravity acceleration. In the class *param* the symbolic variables of all multibody system objects are comprised (Otter et al., 1993).

Due to object-oriented software construction techniques, the composition of abstract data types in classes demands a description of the operations valid on the objects. These operations are designed reflecting a practical multibody modeling process. For all classes the basic operations *create*, *delete*, *modify*, and *list* are defined, more complex operations take the relationships between objects of a multibody model into account, see Fig. 14.

During the assembling in a multibody modeling process, objects of class *frame* have to be assigned to or removed from an object of class *part*, Fig. 14. The equivalent operation is required to assign objects of class *part* or *interact* to the root object of class *mbs*. Further operations representing the assembling process are the assignment and removal of objects of class *joint* and *force* to the associated object of class *interact*. As an advantage of the object-oriented approach, all these operations are structured in a hierarchical manner.

Special operations are designed to calculate properties of an object. The operation *calculate location* determines the position vector ${}_{i}\mathbf{r}_{ik}$ and orientation matrix $\mathbf{S}_{ik}$ of an object of class *frame* with respect to its reference frame. Depending on this operation, the result of this operation for an object of class *interact* is the location of the related aframe and bframe with respect to the global inertial frame. For an object of class *joint*, the joint specific relative position vector ${}_{k}\mathbf{r}_{kl}$ and orientation tensor $\mathbf{S}_{kl}$ is calculated.

The benefits of the object-oriented approach become evident from the design technique of the operations. Changes in class components, the addition of new joint or force classes affect only local modifications in the overall class and operation definitions. All further basic and extended operations for the multibody modeling process remain valid.

5.2 Multibody system classes and graphic description

While the classes of the preceding subsection describe the mechanical elements of a multibody system, further classes are required for the graphical representation. The mechanical and mathematical properties of an object of class *frame* is completely determined by its components,

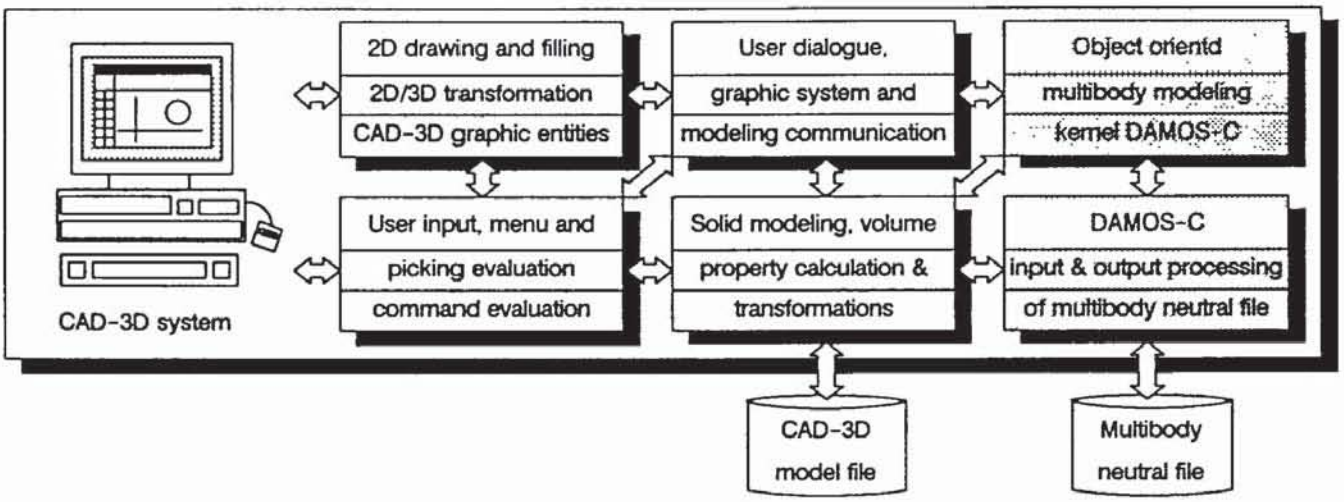

FIG. 15 INTEGRATION OF MULTIBODY MODELING KERNEL

the information about the actual frame axis length, its color or visibility depends on the actual multibody size and modeling state. A geometry data model for multibody elements well suitable for machine, robot and vehicle dynamics requires a unique spatial representation of the multibody element, its function and physical quantity. Moreover, an arbitrary adaptation of the graphic representation to the shape and size of multibody elements is necessary, as well as an spatial representation of special multibody elements like joints to model the relative degree of freedom.

From Fig. 5 and Fig. 8 it becomes obvious that spatial parametrized shapes satisfy a graphic representation for objects of class *frame*, *joint*, and *force*. The definition of the classes *g3frame*, *g3joint* and *g3force* and operations for the geometry data model is equivalent to the multibody data model and includes classes comprising color, projection and viewpoint data.

The implementation of the object-oriented classes and operations is performed by means of *data types* and *routines*, which result in a *system-independent modeling kernel library* for multibody systems. This high level library DAMOS-C (Data MOdel for multibody Systems implemented in C) supplies interfaces to input and output as well as for the graphic representation. Moreover, routines are designed to parametrize a multibody system in an early state of modeling. This open interface allows the integration in a commercially available CAD 3D and a new developed graphics system (Daberkow 1993).

The integration scheme shown in Fig. 15 reveals the interfaces to the different CAD 3D software moduls. An extension of the CAD command language supplies additional commands which are necessary for the execution of multibody modeling operations. To assure the graphic display of the modeling elements, the parametrized shapes are modeled via the 3D wireframe entities of the CAD graphic subsystem. A *multibody system neutral file* serves to store the multibody objects.

6 APPLICATIONS OF THE OBJECT-ORIENTED APPROACH

In this section advantages of an integrated modeling and simulation approach are presented. The modeling of a crank-slider mechanism, as a special planar case of a spatial multibody system, explains the data model application for complex systems.

6.1 CAD 3D modeling steps of a crank slider mechanism

Most mechanisms are well suited for a CAD 3D modeling, since the single parts and shapes can be modeled easily by a solid model design. The solid model construction is performed by volume oriented techniques in PARASOLID. All bodies of the crank slider mechanism of a single cylinder four stroke engine are shown in Fig. 16. Each body is supplied with adequate density attributes.

The first multibody modeling step is the initialization. Here, an appropriate solid is chosen as the inertial body of the multibody system, see Fig 16. In the next step the other solids are chosen to have the properties of a multibody *part*. Each object of class part retrieves its mass and inertia components from the mass property calculation modul *masspr* of PARASOLID. To visualize the multibody part property, the equivalent solids are supplied by reference frames, located in the center of gravity.

The following step consists of the creation of joint and force definition frames and objects of class *joint*. By default, the orientation of these frames is parallel to the spe-

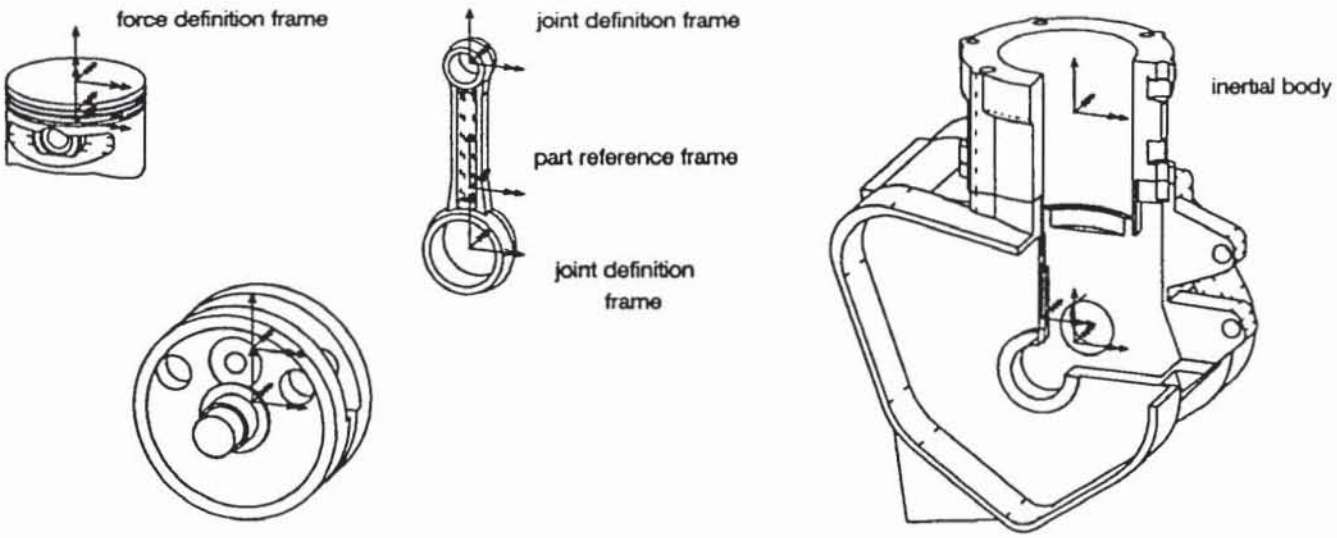

FIG. 16 BODIES OF CRANK SLIDER MECHANISM WITH FRAMES

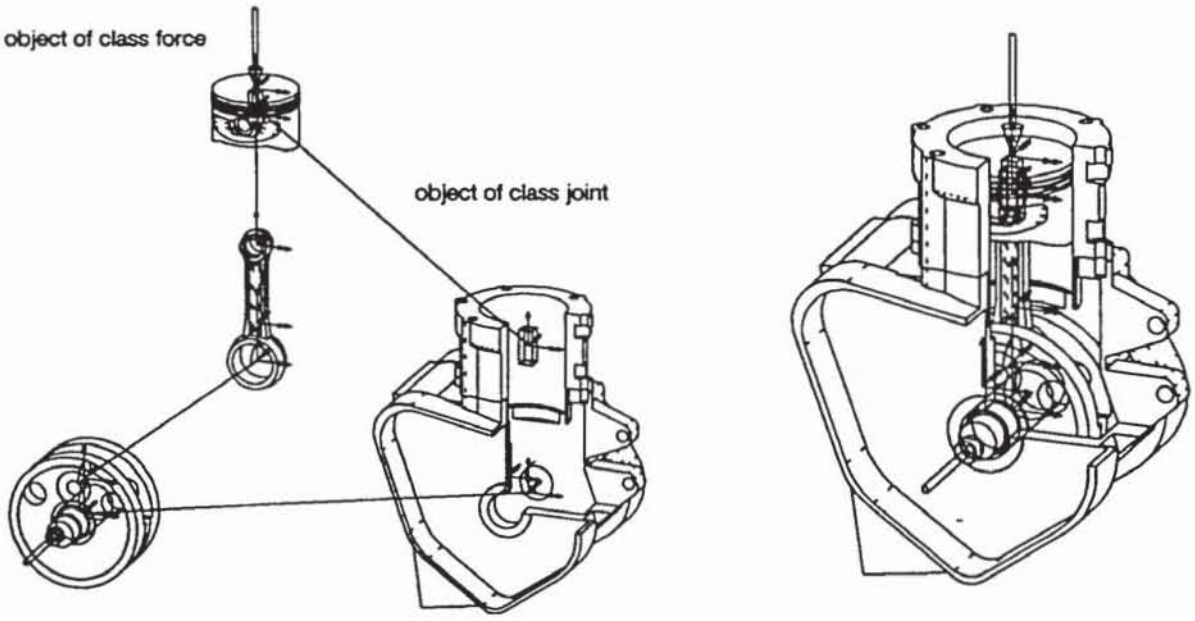

FIG. 17 DISASSEMBLED AND ASSEMBLED MECHANISM WITH JOINT AND FORCE OBJECTS

cified reference frame of the body under consideration. The position of the frames is defined by the CAD 3D picking commands performed by the user. Figure 16 shows these modeling steps and the graphic representation of the objects. Joint definition frames are located along the unit normals of those faces, which form bearing surfaces or bearing bores of a solid.

A planar system modeled for spatial analysis requires a proper constraint selection. Redundant constraints remain if the mechanism is supplied only with joints of class *revolute* and *translational*, making the system overdetermined. Moreover, inevitable manufacturing inaccuracies may prevent an assembly of the crank-slider parts and therefore call for nonredundant constraints, too. Consequently, for an analysis with the Cartesian or Lagrangian coordinate approach the modified joints shown in Fig. 17 are chosen, which results in a model with independent kinematic constraints.

The objects of class *revolute* are visualized by the parametrized shapes and wireframe entities, the connection between the objects of class *part* by the object of class *interact*

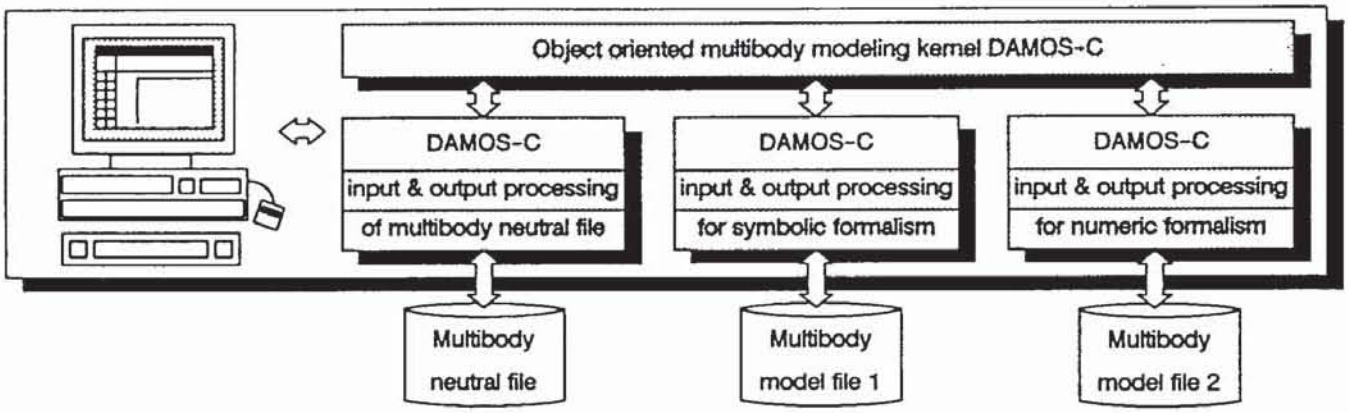

FIG. 19 MULTIBODY MODELING KERNEL INTEGRATION AND PROCESSING OF INPUT FILES

is visualized by a 3D line entity between the aframe and bframe object.

Further useful operations provided by the multibody modeling kernel library are e.g. an assembling of arbitrary objects of class *part*. Figure 17 shows the assembling of the individual objects over the equivalent objects of class *joint*. By modifying the *rangle* component of arbitrary objects of class *joint* , an initial multibody configuration is adjusted interactively, providing therefore an initial estimate for closed loop systems. Finally, an object of class force general is added to the piston part.

6.2 Analysis, simulation and visualisation

In the present implementation, the multibody model conversion from the extended CAD database to a multibody computer code is realized by the data base system RSYST, see Otter et al (1993). Based upon the multibody modeling kernel library output routines, the components of each object are converted into RSYST from the multibody system neutral file. Integrated RSYST multibody moduls like a symbolic Newton-Euler formalism NEWEUL generate the symbolic equations of motion and automatically produce a problem-specific simulation program. As a result of the simulation, a time plot of the resultant crankshaft constraint force of the mechanism under an applied piston gas force and an animated sequence is shown in Fig. 18.

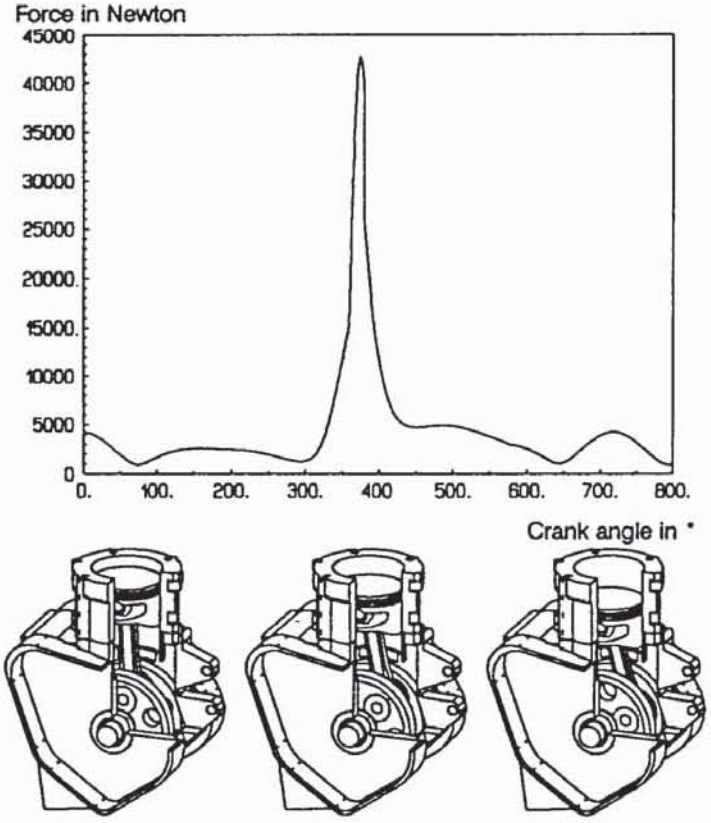

FIG. 18 ANIMATION SEQUENCE AND RESULTANT CRANKSHAFT BEARING FORCE

6.3 Automated generation and conversion of multibody software input data

Besides the integration of the multibody modeling kernel within commercially available or new developed graphics systems, another important application is the processing and conversion of different multibody software input descriptions. Most multibody computer codes obtain their input from data files in the specific multibody program language. As the object-oriented data model supplies a unified description for symbolical and numerical multibody programs relying on different coordinates, it provides a facility to generate different input data files. The central role of the object-oriented multibody kernel library DAMOS-C is obvious from Fig. 19.

DAMOS-C class and routine definitions

```
...
#include "../mbs_include/mbs_classes.h"
#include "../mbs_include/mbs_routines.h"
...
   MBS_SYSTEM   mbs;
   MBS_FRAME *frame;
   MBS_BODY   *body;
   MBS_PART   *part;
   INTEGER    ip,ifr;
   ...
   mbs = mbs_sy4gmp();
   numobj = mbs_sy6nup(mbs);

   for (ip=1;ip<=numobj;ip++)
       {
        part  = mbs_sy6gpi(mbs,ip);
/*      specific multibody part processing
        ...
        numfr = mbs_pa6nuf(part);
        for (ifr=1;ifr<=numfr;ifr++)
            {
             frame = mbs_pa6gfi(part,ifr)
/*           specific frame processing
             ...;
            }
        body = mbs_pa4gbp(part);
/*      specific body processing
        ...
       }
...
```

DAMOS-C object declarations

pointer to root object

number of parts retrieval

pointer to part object

number of frames retrieval

identification of frame object

identification of body object

Multibody Software 1 input file

```
...
 new part1 rigid
   frame
    new framjnt1 framgen
      rframe  = ' '
      sorigin = ' ',' ',' '
      dorigin = 0.0,-10.0,0.0
      axleseq = 1,2,3
      sangles = ' ',' ',' '
      dangles =0.0,0.0,0.0
    go
...
   go
```

Multibody Software 2 input file

```
...
PART/01, MASS=0.200000  Rigid--
...
MARKER/0101, QP=0.0, -10.0, 0.0
...
```

Multibody Software 3 input file

```
...
C>
C>    Data of Mass Distribution
C>  ****************************
C>
C>  Data of Mass Distribution for
KOSYNA: S1  Name of Coordinate System
C>  Mass
  MASS = M1
C>
C>  Inertia Tensor
KOSYNA: S1  System for Inertia Tensor
C>
  I1(1,1) = 10.000000
  ...
```

FIG. 20 SCOPE OF DAMOS-C STATEMENTS TO PROCESS MULTIBODY INPUT FILES

Based on the DAMOS-C library, input and output moduls are developed to parse and generate the multibody system neutral file, respectively. The effectiveness of the library is demonstrated by a sequence of routines necessary to generate an input data file for a symbolical as well a numerical multibody formalism. Figure 20 shows the DAMOS-C technique to identify, access and retrieve the components of all objects of class *part* and *frame* of a multibody system. From the pointer `mbs` to the root multibody system object, the number of objects of class *part* is retrieved. For each object of class *part* with pointer `part`, the equivalent objects of class *body* as well as the objects of class *frame* are identified. The components of the objects are formatted and processed with respect to the specific multibody computer code.

Access to each object is given by the pointer from which basic components like a unique name or a unique integer tag for a graphic object reference are retrieved. Further, equivalent statements in DAMOS-C exist to access *joint* and *force* objects.

7 CONCLUSION

In this paper an integrated, object-oriented approach for the computer aided modeling and analysis of multibody systems is introduced. On the basis of multibody methods and principles, a unified general data model including the graphic description is developed. From the mechanical properties of multibody elements, using object-oriented techniques, classes are defined. Additional operations are determined in an object-oriented manner supporting the multibody modeling process. For a preceding CAD 3D modeling stage, a unified spatial graphic representation for multibody elements is designed. Object-oriented

classes and operations are then implemented in a system independent multibody modeling kernel library. The integration of this kernel library into a CAD 3D system demonstrates the advantages of this approach. Fundamental and high level functions for the modeling of multibody systems fit the criteria of an extended modular automated design tool. Further benefits are demonstrated by the possibility to solve data exchange problems between different multibody computer codes.

REFERENCES

ARIES Conceptstation Software Simulation Mechanism Reference, Aries Technology Inc., Lowell, MA, 1990.

Braid, I.C., 1974, *Designing with Volumes*, Cantab Press, Cambridge, England.

Brown, M.D., 1985, *Understanding PHIGS, The Hierarchical Computer Graphics Standard*, Software Division of Megatek Corporation, San Diego, CA.

Daberkow, A., 1993, "Zur CAD-gestützten Modellierung von Mehrkörpersystemen," *Fortschritt-Berichte der VDI-Zeitschriften*, Reihe 20, Nr. 80, VDI, Düsseldorf.

Haug, E.J., 1989, *Computer Aided Kinematics and Dynamics of Mechanical Systems*, Allyn and Bacon, Boston, MA.

Hollar, M.G., Rosenthal, D.E., 1991, *Concurrent Design and Analysis of Mechanisms*, Rasna Corporation, San Jose, CA.

Kreuzer, E., 1979, "Symbolische Berechnung der Bewegungsgleichungen von Mehrkörpersystemen," *Fortschritt-Berichte der VDI Zeitschriften*, Reihe 11, Nr. 32, VDI, Düsseldorf.

Kreuzer, E., Schiehlen, W.O., 1985, "Computerized Generation of Symbolic Equations of Motion for Spacecraft," *Journal of Guidance, Control, and Dynamics*, 8, No 2, pp. 284-287.

Leister, G., Bestle, D., 1992, "Symbolic-numerical Solution of Multibody Systems with Closed Loops," *Vehicle System Dynamics*, Vol. 21, pp. 129-142.

Meyer, B., 1988, *Object-oriented Software Construction*, Prentice Hall, New York.

Mortenson, M.E., 1985, *Geometric Modeling*, John Wiley, New York.

Orleanda, N., 1973, "Node-Analogous Sparsity-Oriented Methods for Simulation of Mechanical Systems," Ph.D. dissertation, University of Michigan.

Otter, M., Hocke, M., Daberkow, A., Leister, G., 1993, "An Object Oriented Data Model for Multibody Systems," *Advanced Multibody System Dynamics - Simulation and Software Tools*, W.O. Schiehlen, ed., Kluwer, Dordrecht.

Pahl, G., 1990, *Konstruieren mit 3D-CAD Systemen: Grundlagen, Arbeitstechnik, Anwendungen*, Springer, Berlin.

PARASOLID Solid Modeling System Kernel Interface Reference Manual, Shape Data Ltd., Cambridge, England, 1990.

PATRAN Mechanical System Simulation P/Mechanism User's Guide, PDA Engineering, Costa Mesa, CA, 1990.

Roberson, R.E., Schwertassek, R., 1988, *Dynamics of Multibody Systems*, Springer, Berlin.

Rosenthal, D.E., Sherman, M.A., 1986, "High performance multibody simulation via symbolic equation manipulation and Kane's method," *Journal of Astronautical Sciences*, 34, pp. 223-239.

Schiehlen, W.O., 1986, *Technische Dynamik*, Teubner, Stuttgart.

Schiehlen, W.O., ed., 1990, *Multibody System Handbook*, Springer, Berlin.

Schiehlen, W.O., Daberkow, A., 1988, "Modeling, Simulation and Animation of Nonlinear Multibody Systems," *Proceedings of the Third Conference of Theoretical and Applied Mechanics*, Academy of Scientific Research and Technology, Cairo, pp. 27-48.

Thatch, B.R., Mykleburst, A., 1988, "A PHIGS-Based Graphics Input Interface for Spatial-Mechanism Design," *IEEE Computer Graphics & Applications* 2, Vol. 8, pp 22-38.

Weber, H.R., ed., 1988, *CAD-Datenaustausch und -Datenverwaltung*, Springer, Berlin.

Wehage, R.A., Haug, E.J., 1982, "Generalized Coordinate Partitioning for Dimension Reduction in Analysis of Constrained Dynamic Systems," *Journal of Mechanical Design*, Vol. 104, pp. 247-255.