Embedding Reliable Numerical Analysis Capabilities into an Enterprise Wide Information System

Ottmar Klaas* and Mark S. Shephard
Scientific Computation Research Center
Rensselaer Polytechnic Institute
Troy, New York 12180-3590
USA

November 15, 2000

Abstract

A simulation environment to support engineering design embedded in an enterprise wide information system is presented. The environment consists of a set of structures and managers housing the problem definition, tools for controlling the simulation model construction and execution, and the interaction of simulation processes with the product data management system. The issues associated with the introduction of automated and adaptive geometry-based simulation processes into the engineering design process are emphasized.

Keywords. Geometry-based simulation; Engineering Design; Product Data Management; Computational Mechanics

*Correspondence and offprint requests to: O. Klaas, Scientific Computation Research Center, CII 7311, Rensselaer Polytechnic Institute, Troy, NY 12180-3590, USA. email: oklaas@scorec.rpi.edu

1 Introduction

Companies fully realize that the generation, control and integration of all levels of engineering information is key to the design and manufacture of superior products. The technical challenges they face in addressing the complex array of issues associated with this area are formidable. Fortunately there are a number of tools available to them to address this area. A key problem is that many of these tools have been developed independently with little knowledge of the technologies underlying the other tools with which they interact. The goal of this paper is to begin to address the technical issues associated with bringing advanced simulation methods into the corporate information management systems that have historically needed to focus their attention on dealing with distributed engineering design and manufacturing processes.

Historically, advanced numerical simulation processes have not been well integrated into the information systems of a company. Efforts to bring simulation into the engineering design/manufacturing process have focused on individual capabilities which operate only on data descriptions natural to the specific operation to be performed rather than tying processes together through product data management and workflow systems. The effective use of numerical simulation in product design requires a shift from individual point solutions to the effective handling of the appropriate data. To meet this goal an automated simulation environment must interact seamlessly with product data management and workflow systems to execute the simulations in a reliable fashion. Appropriate information control structures, and model tracking and modification processes, are needed to execute the full range of processes associated with a numerical simulation and to communicate the results back to the design process.

Section 2 overviews a geometry-based simulation environment designed to provide users with the tools needed to automate the execution of reliable engineering analyses within an integrated design and manufacturing information system. The proposed geometry-based simulation environment includes a set of existing information management, CAD modeling and CAE analysis tools. These tools are coupled to a set of tools which not only link the existing tools together but provide the additional technologies needed to automate the simulation processes and adaptively control their execution to ensure, to the level possible with existing technologies and knowledge, the

accuracy of the results obtained. Section 3 defines the technologies needed to support such automated adaptive procedures and indicates their current status of development. Section 4 indicates the methods to link these new technologies to the existing CAD and CAE technologies.

2 Geometry-Based Simulation Environment

In engineering design, simulation technologies must provide reliable estimates of performance parameters and sensitivity information with respect to parameterized product designs, thus providing key information for the application of design procedures. The input information provided for a simulation includes the definition of design as it currently exists in the product data management system, and the set of performance parameter values and/or sensitivities to be estimated and level of accuracy required. Based on this, the simulation is to determine the values of the performance parameters, to the level of accuracy requested, linked into the product data management system. Arabshahi et al. [1] present the vision of a design-analysis environment that contains some of the major components needed for the described simulation process. They include a Product Data Management System, a CAD-FEA transformation component that prepares the geometric model for the meshing module and supports the application of the analysis attributes to the geometric model, a FE-solution component, and a results processing component to feed back the results to the CAD-FEA transformation component. However, they do not mention the underlying structures and technologies necessary to support the functional units.

The current paper focuses on the structures needed to support the complete integration of simulation into the design/manufacturing processes and the technologies needed for their automated and adaptive execution. In addition, specific consideration is given to providing these new components in such a way that they will effectively integrate with the existing data management, CAD and CAE technologies that are currently used in industry. Figure 1 provides a view of the components of this proposed simulation environment (components within the dashed box) and the links to the other key system components.

The key existing components indicated in Figure 1 include:

- Product Data Management System
  This system is responsible for storing the problem definition data. It

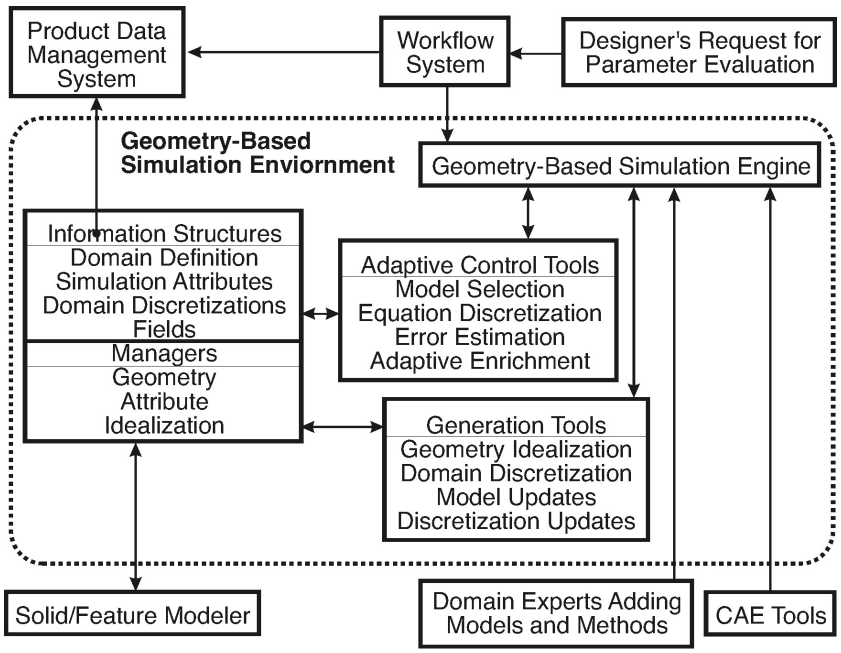

Figure 1: Simulation environment to support engineering design

provides a single resource for the data eliminating redundancy, and guaranteeing accessibility and security. Revision control is provided so that older versions of data can be retrieved, and alternatives can be considered by introducing branches in the revision tree. The Information Managers interact with the Product Data Management System to retrieve up to date data and store the necessary changes. The support for the concept of concurrent engineering crucially depends on these tools.

- Workflow Management System
  This system is responsible for coordinating the flow of data between the different components of the design/manufacturing environment. It monitors the problem definition data for changes, and automatically initiates operations such as a simulation if needed. Due to its link with the data flow of a simulation it also serves as the interface for designers requesting parameter predictions.

- Solid/Feature Modeler
  The representation of the domain of the object being designed is a key

component of the system. From the standpoint of effective simulation-based design processes this model would be a feature-based model designed to support design decision processes. As indicated in [2] such approaches also support the effective specification of the other information needed to support simulation processes. However, with the exception of specific ad-hoc procedures [2], such feature-based models with simulation knowledge are not available. Therefore, we will assume the domain description is built upon basic solid modeling technologies. For the purpose of supporting automated simulation processes during a design process the appropriate form of solid model representation is a non-manifold boundary representation capable of supporting the specification of general combinations of solids, surfaces and wires. Current commercial systems support such non-manifold representations and the ability to interact with them through functional interfaces [3],[4],[5]. The procedures discussed here will focus on the interactions directly with the non-manifold solid model representation.

- CAE Tools
  Engineering analysis procedures that support engineering design processes range from simple design rule calculations to generalized analysis procedures capable of approximately solving field equations over general domains to any desired degree of accuracy assuming sufficient computational effort. The paper considers only the second group of procedures due to their ability, given proper usage, to provide solutions to the desired level of accuracy. Furthermore, emphasis is on those methods that employ generalized domain discretization techniques so that domains of arbitrary complexity can be considered. The commercial version of such CAE tools include finite element, finite volume and finite difference based analysis codes. Although these codes are capable of providing the desired levels of accuracy, current codes only provide this accuracy if the user is knowledgeable enough to make the correct modeling decisions and create adequate domain discretizations.

- Experts Models and Methods
  Although there are a number of technologies available to automate the execution of engineering analyses and to, under software control, adaptively improve the analysis discretizations, there a number of modeling and analysis execution decisions that require expert knowledge to be

made. Since the expertise required for these decisions is often outside the expertise of the design engineer that wishes to employ those analysis results, there must be a mechanism for capturing this information. Although those components are not discussed in any detail in the current paper, the simulation environment must include technologies to capture such information and to ensure it is properly considered and accounted for when more automated procedures are not available.

The structures and tools needed to link the existing technologies together and to provide technological capabilities needed for automation of reliable simulation procedures include (see items in dashed box in Figure 1):

- Information Structures and Managers:
  A rich set of structures is needed to house the problem definition, simulation models and simulation results. In addition to being able to support the representation of the information, these structures and managers must function within an adaptive environment in which all structures are constantly evolving to meet the needs of the simulation procedures to provide the requested parameter predictions. The managers interact directly with the product data management system to store and retrieve the problem description data as well as modifications to that data which become apparent during the simulation.

- The Geometry-Based Simulation Engine:
  This represents the framework responsible for executing the simulation process. It must be able to work with the geometry-based structures underlying the product definition and the adaptive evolving discretizations to execute the simulations. In some cases this will include the linkage to external CAE analysis engines, while in others this engine will perform the entire simulation. A simulation can be triggered automatically through the workflow system if the input data for that simulation has been changed, or a request for the evaluation of certain parameters is given.

- Adaptive Control Tools:
  These tools are responsible for determining the appropriate mathematical models, selecting discretization technologies, evaluating the accuracy of the predictions obtained, and determining the improvements of the models and discretizations needed to obtain the desired accuracy.

Initial mathematical model selection must be based on the application of rules on the information contained in the current problem definition. The discretization technology selection is typically based on what CAE tools are available for solving the selected mathematical model and which is typically most effective. A posteriori error estimation and indication procedures are responsible for the identification and estimation (to the extent possible) of the modeling and discretization errors.

- Tools for the Automatic Generation of Simulation Models:
  These are the tools responsible for performing the geometry-based operations to construct the geometric domain to be used for a simulation from the product definition, performing domain discretization, and updating the simulation models and discretizations as dictated by the simulation process. Included in these procedures are rule bases to determine initial modeling systems, geometric modification tools, and automatic discretization (mesh) generation and enrichment tools.

The following section describes the structures and tools that were summarized here in more detail.

3 Tools and Structures to Support Automated Adaptive Simulations

3.1 Information Structures and Managers

Two structures central to the problem definition are the geometric domain of the objects to be considered, and the attribute structure used to specify the remaining physical parameters needed to define a simulation problem. The attribute structure also supports information used to convert the simulation problem definition into a mathematical model form appropriate for simulation. The third structure houses the domain discretizations employed by the simulation procedures. These domain discretizations can include finite element meshes [6], finite difference grids, partition of unity discretizations [7], etc. The fourth structure, referred to as a field, links the simulation results, defined on the appropriate discrete models, to the problem definition and the product definition.

3.1.1 Geometric Domain

In most cases the solid modelers underlying the CAD systems used in industry can fully support the definition of a current instance of a product domain. Although there have been advances in the ability to transfer geometric model information between systems [8],[9], the demands of the geometry-based operations needed in engineering simulations can not be fully supported through these data exchange methods. For example, it has been shown how the geometric modeling system tolerance information, which is not represented in the exchange specification, is needed to support automatic mesh generation functions [10]. On the other hand, it has also been shown that robust automatic mesh generators can be developed [10],[11] that directly rely on the geometry engine of the solid modeling system through the abstraction of topology [12]. This approach to supporting simulations using a non-manifold boundary representation is capable of representing the full range of domains needed by engineering simulations. We present a more detailed description of the linkage to the geometry engine in section 4.3.

3.1.2 Discretization

Structures are needed to effectively represent the various forms of domain discretizations and maintain their association with the domain definition as needed to associate attribute data to the discrete models and to relate the simulation results back to the domain definition. Such a structure has been developed for finite element meshes using an effective boundary hierarchy of regions, faces, edges and vertices [6]. Partition of Unity methods rely on a different type of discretization to be effective [7]. A key aspect of any discrete representation used is maintaining the relationship of the entities in the discrete representation to those in the domain representations [6].

3.1.3 Attribute System

In addition to geometry, the definition of a simulation problem requires other information that describes material properties, loads and boundary conditions [13], [14]. This information, to be referred to as attributes, is tensor valued, has general variations and dependencies, and must be associated with the definition of the domain. Although current CAD and CAE systems support some specification of attributes, they do not provide the full set of

capabilities needed to effectively support simulation during the design process. Starting from requirements [13], [14] an initial attribute specification capability has been developed [15].

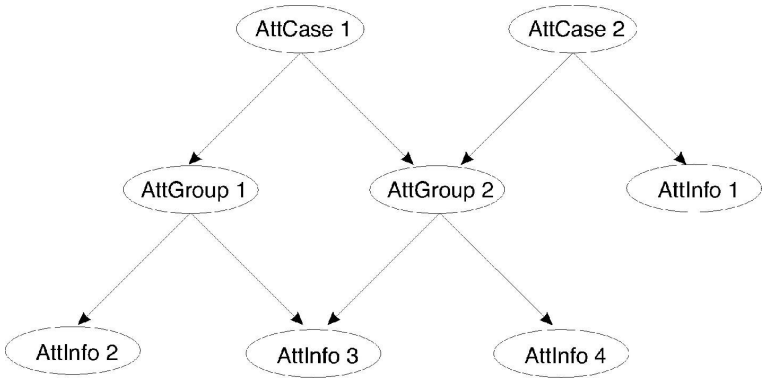

Figure 2: Attribute Graph

In general attribute information can be classified into three main groups:

- physical parameters for the analysis problem specification,
- numerical discretization control information, and
- mathematical and numerical methods used in analyzing the problem.

The physical parameters are directly associated with geometric entities. This information is associated to the numerical analysis discretization using the links from those entities to the geometric model. Using a geometry-based attribute specification is needed to support the use of adaptively defined discretization. The numerical discretization control information provides a means to control the discretization parameters, such as the size of the discretization entities. The possibility to apply these type of attributes to the model as well as to individual model entities provides easy global and local discretization control. Mathematical and numerical methods such as the type of nonlinear solver used and the type of time integrator, belong to the last class of information needed to specify a numerical analysis.

The attribute information is stored in a hierarchical manner in a directed acyclic graph [15]. Leaf nodes (Attribute Information Nodes) in the graph hold the actual attribute information, like tensors, strings, integer or floating point values. Group nodes are used to collect information for specific

purposes. Special group nodes, called Case Nodes refer to a set of attributes that have a meaning as a whole with respect to the specification of a simulation. For example, all the attributes describing the numerical solution procedure for a specific problem would be called a case. Figure 2 shows the basic structure of the attribute graph.

The association of analysis attribute information with the geometric model is done by specifying the topological entity of the model onto which the attribute is being applied. Model associations are introduced that store the relation between topological entities and attributes. That ensures that different cases can reuse an Attribute Information Node and assign its information to a different model entity. The implementation of the attribute structures as well as the modules it interacts with (expression system, model interface) has been developed in C++.

To be able to retrieve the attributes associated with certain model entities the association is represented as a list of attributes stored as a member variable in each model entity class. The attributes can be retrieved by name, which is an arbitrary string defined by the user of the attribute system. Each attribute stores its node type (Case Node, Group Node or Information Node) and its representation type (void, integer, double, string, tensor, field). Case and group nodes contain member functions that allow retrieval of the child attributes. Information nodes provide implicit type conversion operators to convert an attribute into its underlying representation.

The attribute system makes use of the expression system to provide support for spatial and time dependent variations of attributes. The expression system acts as a parser and lexicographical analyzer to convert strings into expressions that can be numerically evaluated. The expression system understands general algebraic expressions, several common mathematical functions (sine, cosine, logarithm etc.), well known mathematical constants, and simple control structures (if, else). Reserved key words are introduced to consider the time and spatial orientation as parameters for an attribute.

3.1.4 Field Structure

To support the proper association of the simulation results with the domain definition more than just result values at a set of discrete points is needed. The field structure [16] stores all the information needed to calculate the represented tensor over the geometric domain.

Classical numerical analysis codes provide results at specific points in the

solution domain. Those points are usually nodes or integration points of the discretization where results are readily available. As numerical analysis codes have mostly been used as a single point solution capability in the past this approach was sufficient. Evaluation of the obtained results was done merely to provide a visual feedback of the analysis. No further processing was performed. However, this approach fails in an automated simulation environment where coupled physics simulations are required. Analysis results produced might be needed as input into another analysis code, for error estimation purposes, or to be mapped onto another discretization. Without knowing the specifics of the analysis code that produced the original results it is impossible to properly perform that task given just a list of discrete values. To avoid the problems, a construct known as a field is introduced [16].

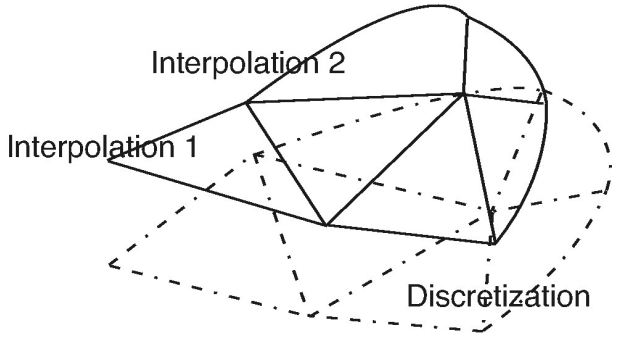

Figure 3: Representation of a field

A field describes the variation of a tensorial quantity over one or more entities in a geometric model. That variation is defined in terms of interpolations. Each interpolation defines the variation of a tensor over the domain of an entity of the discretization. A field is a collection of individual interpolations, all of which are interpolating the same quantity (Figure 3).

Interpolants can be written as a linear combination of shapefunctions multiplied with coefficients:

$$\mathbf{A}(\xi) = \mathbf{a}_o N_o(\xi) + \mathbf{a}_1 N_1(\xi) + \ldots + \mathbf{a}_n N_n(\xi) \tag{1}$$

The $\mathbf{a}_i$ (and thus $\mathbf{A}$) can be any order tensor. ξ is the location in the local coordinate space where the interpolation is to be evaluated, and N_i are shape functions. The coefficients $\mathbf{a}_i$ are stored on the entities of the discretization.

Coefficients that are shared between interpolations enforce C^o continuity of the interpolation. Discontinuous interpolations can be written by storing the coefficients on entities of the discretization that are not shared, e.g. in a 2-dimensional setting the coefficients could be stored on mesh faces instead of the mesh edges.

The main functionality of a field is to provide evaluations of the solution field and various spatial, temporal or other derivatives as desired. The request for evaluation at a certain point is directed to the interpolation in whose domain the point is located. The global coordinates of the point are then transformed into local coordinates reflecting the local coordinate system of the interpolation based on the mapping that was defined for the interpolation. The interpolation can than be evaluated using (1), and transformed back into global coordinates. The process of evaluation is completely self contained in the field structure, i.e. no access to the numerical analysis code that calculated the result is needed. Any further processing of the results can be performed without loss of accuracy.

3.1.5 Managers

The managers are responsible for controlling the information flow to and from the structures presented in the previous sections. Within an adaptive simulation environment these managers must do more than simply provide operators to transfer data into and out of the structure. They are responsible for performing the operations needed to ensure the information is in the form needed by the simulation process, and for interacting with the product data management system to retrieve and store the current product data in order to keep the repository up to date.

The geometry manager is responsible for coordinating the operations to create the simulation domain needed. The process of defining the simulation model domain often requires modification of the domain definition to account for the analysis idealizations. Therefore, the manager must interact with the solid modeler to reflect these modifications. Since in the conceptual design process the design may have idealizations that are improved based on knowledge gained in the simulation, and when simulation driven design optimization may alter the design, it must also be possible to communicate these modifications to the product definition housed in the product data management system.

The idealization manager is responsible for coordinating the activities

of all simulation idealization processes. It must maintain knowledge of the idealizations performed, what dictated them, and how they affected the information in the domain and attribute structures.

3.2 Geometry-Based Simulation Engine

The simulation engine needs to be implemented as an extendable framework for the following reasons:

- The technologies needed for the full range of simulations that must be performed is not known at this time. The flexibility of a well designed framework will allow it to easily incorporate future simulation needs.
- There are a number of current CAE tools that can be applied as a major component of a simulation. Those tools can be incorporated into the framework and take over part of the simulation with the framework providing the needed interface functionality.
- The range of applications to which simulation is applied will be constantly growing. Therefore, an effective simulation engine must provide the ability for domain experts to easily introduce new models without the need to understand the details of the entire system.

Recently, Sahu et al. [17] presented such an extendable object-oriented framework for computational mechanics. It is designed to close the loop between CAD and analysis packages, facilitating the automatic solution of complex evolutionary problems where both the mathematical and numerical models may evolve. Beall et al. [16] also present an object-oriented framework for computational mechanics. Their primary focus lies on developing a simulation package that enables reliable numerical simulations in a geometry-based environment.

Conceptually the simulation framework is built on the view of an analysis as a transformation between three levels of description [16]. Figure 4 shows the basic transformations taken to go from a physical object to a discretized analysis model. Starting from a real world problem posed in terms of physical objects and the environment they are situated in, a mathematical model has to be selected that approximates the real world problem. It is being defined in terms of a domain definition (geometric model), the properties of the entities and external influences on them (attributes), and the

mathematical equations describing the unknown solution in terms of known parameters. The next transformation is to form a numerical problem from the mathematical problem given. In the present case this is the finite element mesh. The mesh information is further transformed to a set of simultaneous equations which are then solved. The result information is then transformed back to the higher level models using the field constructs.

The framework currently under development [16] makes use of modern software development techniques [18],[19],[20] to make sure that it is extendable. So far it has been used for Finite Element Analysis, as well as Partition of Unity methods [7].

3.3 Adaptive Control Tools

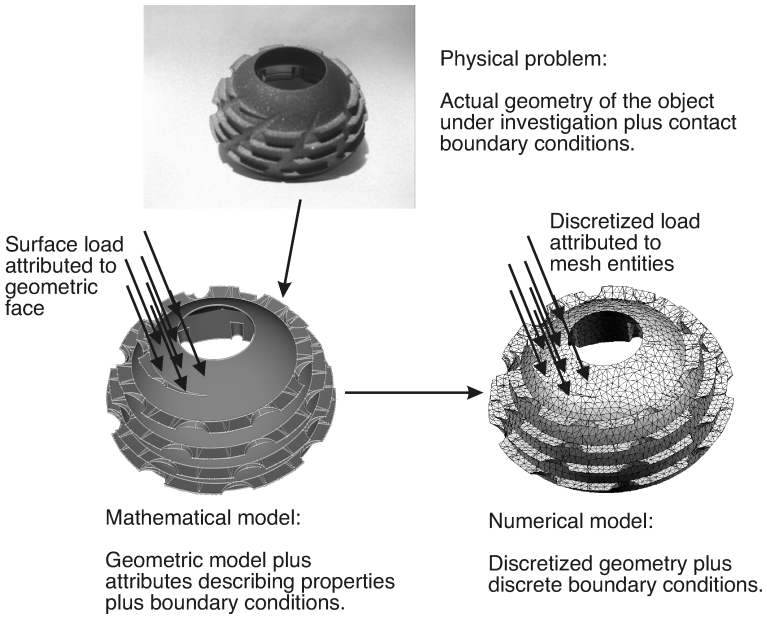

Figure 4: Idealizations of a physical problem

The goal of the analysis is to obtain reliable estimates of the response of the physical system. The mathematical problem description introduces some level of idealization, which needs to be controlled to yield the desired accuracy. The next step in the idealization introduces the numerical problem, which is another set of idealizations that also need to be controlled.

Based on this view current CAE tools are simply the solution engine to solve one single problem when it has reached the second idealization stage in an overall adaptive process controlled by the framework. Tyson [21] pointed out that the use of the numerical problem description as the starting point for integrating design functions is one of the primary deficiencies of simulation software packages.

In an automative simulation environment the first operation that must be performed is to determine which techniques can be used to perform the simulation needed to estimate the requested parameters. Considering a case of estimating a single parameter, the minimum this process includes is determining which sets of mathematical models and discretization techniques can be applied to determine the parameter to the requested accuracy. Given the set of possible mathematical modeling approaches and discretization methods, the product definition must be examined to determine if the level of definition is complete to the level needed to apply one or more of the possible mathematical modeling and discretization methods. An overall framework that can support these operations is needed. Technologies to support these operations will range from inference engines to support sets of heuristically-based rules [22], to mathematically based procedures that interrogate information in the product definition to measure parameters needed for the decision on the appropriate models. In the majority of simulations to be performed there are no a-priori means to determine the level of accuracy to be obtained from the simulation. Over the past two decades researchers have been developing techniques to take the results for a given simulation and, using knowledge of the approximations made in the process, estimating the errors for the simulation [23], [24]. The form of this information is such that it provides useful guidance as the best means of improving the approximations made to gain the desired level of accuracy. The application of such knowledge in an adaptive feedback loop [25], [26] is the ideal means to obtain the accuracy desired. Adaptive feedback procedures for predicting simulation errors are only known for discretization processes for some mathematical models. Therefore, it is necessary to consider and support all means that are, and may become, available to control all simulation errors. These will range from heuristic rules, to specific levels of validation against experiments, to new mathematical modeling techniques.

3.4 Automatic Generation of Simulation Models

To be fully automatic, the system must automatically generate the initial discretization, associate the appropriate attributes with the discretization and perform the simulation. In the case when the discretization is of the existing domain definition, the key capability needed is the appropriate automatic domain discretization procedure. Although implementations of such procedures are available [27], most do not support the ability to create the discretization directly from a non-manifold solid model while a small number do support a geometry-based approach [28]. In addition, such procedures are available for only limited discretizations techniques. In the cases where the domain to be discretized for the analysis is different from that in the product definition, procedures are needed to account for these domain differences. In many cases these procedures must perform dimensional reductions of portions of the domain [29] and geometric simplification to remove unneeded geometric details. In other cases, the information available in the product definition must be used to drive geometric modeling operations to define the forms of geometric representations needed by the automatic discretization procedures.

4 Linkage with Data Management, CAD and CAE Tools

4.1 Product Data Management Tools

Product Data Management tools, which are also known as Engineering Data Management systems (EDM) [30] were developed to manage all the data related to a product and the processes used to design, manufacture and support the product over the entire product lifecycle. Ideas, concept designs, engineering designs, test results and reports are examples of the data that can be put under control of a PDM system. A commercial grade database, like Informix [31], IBM's DB2 [32], Oracle [33], or Sybase [34] forms the fundamental unit responsible for storing the raw data. The PDM system supports the presentation of the same data from different views according to the needs and customs of the people working on that data. The improved management of the data helps to reduce the time and cost to introduce new products, and improves the quality of products. PDM systems aim

to improve the overall product development cycle rather than improving an individual task in one functional area, like a CAD system.

The coupling of the geometry based simulation tools with the PDM system adds data security and revision control to the simulation environment. Besides the storage of the geometric domain in the form of a CAD model, which is assumed to be under control of the PDM system alrady, the problem definition in the form of attributes needs to be stored in the PDM system. The PDM system keeps proper track of changes to the problem definition providing means to back track changes, and explore design options using branches in the revision tree. In some instances it may also be advantageous to store analysis results in the PDM system, e.g. if the computational effort to calculate the results is rather high.

4.2 Workflow Management System

With the data being controlled by a PDM system, workflow management systems (WfMS) [35], [36] are used to coordinate and automate the execution of processes. The flow of data between the different tools is controlled through the use of software, according to a set of rules that have been designed using tools provided by the WfMS. Overall, processing time is being reduced, and the workflow can be monitored to identify bottlenecks. Workflow management systems are in high demand in large corporations. However, many technologies (Database Management System, Workflow server, data bridging technologies) have to be integrated to successfully use such a system.

For the purpose of automizing the design/simulation interaction the workflow system needs to be responsible for the following tasks:

- Keying the need for a simulation initiated by
  - a direct request of the designer to obtain estimates of requested physical parameters
  - the need to update parameter estimates based on a design modification
- Coordinating simulation processes in a distributed environment with multiple organizations

The WfMS interacts with the PDM system to be able to perform the described tasks. A design modification committed to the PDM system triggers

the WfMS to initiate those simulation processes whose outcome depend on the modified data. The geometry-based simulation engine is activated to retrieve the appropriate information from the PDM system. To do so it utilizes the information managers that inquire the product definition housed in the PDM system. The WfMS will then control the analysis work flow by initiating the appropriate tools from the geometry-based simulation environment as needed to complete the analysis.

4.3 Model Abstraction to link with CAD

In a fully automatic simulation environment a key point is the ability to interact with the geometric model as it is stored in a commercial geometric modeling system. To provide extensibility and maintainability the modeler functionality needed for a geometry-based simulation environment has been extracted and implemented in a model interface. The model interface provides an abstraction of a non-manifold geometric model using a boundary representation. It implements the Radial Edge structure [12] to provide the topological adjacencies between vertex, edge, face and region elements in the geometric model.

By using the model abstractions all components of the simulation framework are independent of the specifics of any particular geometric modeler [10], [14], [16],[28]. Figure 5 exemplifies the interaction of the analysis tool, the discretization tools and the attribute system with the modeler system through the model abstraction. In addition, the model interface can provide extensions of the model representation that are beneficial for a geometry-based analysis, but which might not exist in the chosen geometric modeler. The modeler can also support related multiple representations of the same geometry. This functionality can prove useful for, e.g., small feature removal, conversion of assemblies into proper non-manifold models etc.

The model abstraction is central for the reliable operation of key components of the simulation framework. It is being used in the discretization tools to control the interactions between the discretization generation operations and geometric modeler, and to support the linkage between topological entities of the discretized model to topological entities of the geometric model [6]. Further, the boundary representation is the crucial feature used by the Attribute system to associate attributes to the geometric model [13], [15].

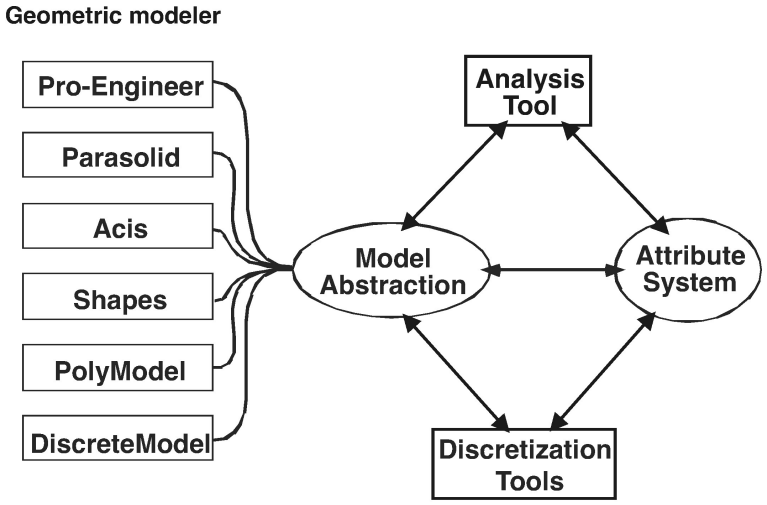

Figure 5: Model Abstraction

4.4 Linkage with CAE

The integration of commercial CAE tools into an automated simulation framework is desirable to take advantage of available simulation procedures. The linkage occurs in two places, the first being the preparation of the input deck for the commercial program based on the geometry-based problem description, and the second being the transfer of the results back into the simulation framework. Since the geometry-based problem description is richer than the information specified in the input deck for a commercial CAE tool, the first task boils down to a reduction in information plus appropriate formatting. The tools needed are readily available in the geometry-based solution engine.

To properly use CAE tools in a fully automated adaptive environment it is also necessary to construct field structures from the results returned from a CAE tool. This requires access to the results at discrete points and the definition of the interpolants to which they are related, which is not necessarily readily available information.

5 Closing remarks

This paper has described the design of a simulation environment to support engineering design embedded in an enterprise wide information system. The different components of that environment are under development at the Scientific Computation Research Center (SCOREC). The level of functionality achieved as of today varies greatly with each component. There exists already a commercialized version of one of the generation tools developed at SCOREC (MEGA [28]), and significant contributions have been made to the geometry-based simulation engine [16], the information structures and managers ([16], [37], [15]), and the adaptive control tools. However, further research and development is required to improve their applicability to full scale industrial sized problems, and to allow for a fully automized simulation environment in an industrial setting.

References

[1] Arabshahi, S.; Barton, D.C.; Shaw, N.K. (1993) Steps towards CAD-FEA Integration, Engineering with Computers, 9, 17 – 26.

[2] Beall, M.W.; Webster, B.E.; Sandboge, R.; Gielda, T.P.; Shephard, M.S. (2000) Simulation-Based Design for Automotive Thermal Management. SCOREC Report #11-2000, Scientific Computation Research Center, Rensselaer Polytechnic Institute, Troy, NY, (submitted for publication), 2000.

[3] EDS Corp. (1994) Parasolid V6 Functional Description. Electronic Data Systems Corporation, 13736 Riverport Drive, Maryland Heights, MO 63043.

[4] Spatial Technologies Inc. (1990) ACIS Geometric Modeler Interface Guide. Spatial Technology, Inc., 2425 55th Street, Building A, Boulder CO, 80301-5704.

[5] PTC (1997) Pro/TOOLKIT Reference Manual. Parametric Technologies Corp., 128 Technology Drive, Waltham, MA, 02154.

[6] Beall, M.W.; Shephard, M.S. (1997) A general topology-based mesh data structure, International Journal for Numerical Methods in Engineering, 40(9), 1573 – 1596.

[7] Klaas, O.; Shephard, M.S. (2000) Automatic Generation of Octree based Three Dimensional Discretizations for Partition of Unity Methods, Computational Mechanics, 25(2/3), 296 – 304.

[8] ISO STEP, "Industrial Automation Systems - Product Data Representations and Exchange - Part 1: Overview and Fundamental Principals", ISO Report CD/10303-1.

[9] ISO STEP, "Industrial Automation Systems - Product Data Representations and Exchange - Part 42: Integrated Resources: Geometric and Topological Representations", ISO Report CD/10303-42.

[10] Shephard, M.S.; Georges, M.K. (1992) Reliability of Automatic 3-D Mesh Generation, Computational Methods in Applied Mechanics and Engineering, 101, 443 – 462

[11] Shephard, M.S.; Georges, M.K. (1991) Automatic Three-dimensional Mesh Generation by the Finite Octree Technique, International Journal for Numerical Methods in Engineering, 32(4), 709 – 749.

[12] Weiler, K. (1988) The Radial Edge Structure: A Topological Representation for Non-Manifold Geometric Modeling, Geometric Modeling for CAD Applications (IFIP WG5.2 Working Conference Rensselaerville, N.Y., May 12-14), Eds.: Wozny, M.J,; McLaughlin, H.; Encarnacao, J., 3 – 36.

[13] Shephard, M.S. (1988) The specification of physical attribute information for engineering analysis, Engineering with Computers, 4, 145 – 155.

[14] Shephard M.S.; Finnigan, P.M. (1989) Toward Automatic Model Generation, State-of-the-Art Surveys on Computational Mechanics. A.K. Noor and J.T. Oden, eds., ASME, 335 - 366.

[15] O'Bara R.M.; Beall, M.W.; Shephard, M.S. (1997) Specifying Analysis Information within a Geometry-Based Framework, Procedures of the Fourth US National Congress on Computational Mechanics, USACM, 137.

[16] Beall, M.W.; Shephard, M.S. (1999) An Object-Oriented Framework for Reliable Numerical Simulations, Engineering with Computers, 15(1), 61 – 72.

[17] Sahu, R.; Panthaki, M.J.; Gerstle, W.H. (1999) An Object-Oriented Framework for Multidisciplinary, Multi-Physics, Computational Mechanics, Engineering with Computers, 15(1), 105 – 125.

[18] Stroustrup, B. (1997) The C++ Programming Language, 3rd edition, Addison-Wesley.

[19] Matthew, H.A. (1998) Generic Programming and the STL, Addison-Wesley.

[20] Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J. (1994) Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley.

[21] Tyson, T.R. (1991) Effective Automation for Structural Design, Journal of Computing in Civil Engineering, 5(2), 132 – 140.

[22] Holzhauer, D.; Grosse, I. (1999) Finite element analysis using component decomposition and knowledge-based control, Engineering with Computers, 15(4), 315 – 325.

[23] Zienkiewicz, O.C.; Zhu, J.Z.(1987) A Simple Error Estimator and Adaptive Procedure for Practical Engineering Analysis, International Journal for Numerical Methods in Engineering, 24, 337 – 357.

[24] Ainsworth, M.; Oden, J.T. (1997) A posteriori error estimation in finite element analysis, Computational Methods in Applied Mechanics and Engineering, 142, 1 – 88.

[25] Oden, J.T.; Demkowicz, L. (1992) Computer Methods in Applied Mechanics and Engineering, Special Issue on the reliability of finite element computations, North Holland, 101.

[26] Shephard, M.S.; Baehmann, P.L.; Georges, M.K.; Korngold, E.V. (1990) Framework for the Reliable Generation and Control of Analysis Idealizations, Computer Methods in Applied Mechanics and Engineering, 82, 257 – 280.

[27] Shephard M.S. (1996) Update to: approaches to the automatic generation and control of finite element meshes, Applied Mechanics Reviews, 49(10/2), 5 – 14.

[28] Shephard, M.S. (2000) Meshing environment for geometry-based analysis, International Journal for Numerical Methods in Engineering, 47(1–3), 169 – 190.

[29] Rezayat, M. (1996) Midsurface abstraction from 3D solid models: general theory and applications, Computer-Aided Design, 28(11), 905 – 915.

[30] McIntosh, K. G. (1995) Engineering Data Management: a guide to successful implementation. McGraw-Hill.

[31] Informix Software Inc., 4100 Bohannon Drive, Menlo Park, CA 94025, USA. Web address: http:/www.informix.com

[32] IBM North America, 1133 Westchester Avenue, White Plains NY 10604, USA. Web address: http:/www.ibm.com

[33] Oracle Corporation, 500 Oracle Parkway, Redwood Shores, Ca 94065, USA. Web address: http://www.oracle.com

[34] Sybase Inc., 6475 Christie Ave., Emeryville, CA 94608, USA. Web address: http://www.sybase.com

[35] Workflow Management Coalition (1995). The Workflow Reference Model. Document Number TC00-1003. Accessible via: http://www.aiim.org/wfmc

[36] Mohan, C.; Alonso, G.; Günthör, R.; Kamath, M. (1995) Exotica: A Research Perspective on Workflow Management Systems. Data Engineering, 18(1), 19 – 26.

[37] O'Bara, R.M.; Beall, M.W.; Shephard, M.S. (in preparation) Attribute Management System for Engineering Analysis.

Figures