Computer-Aided Design and Applications
© 2009 CAD Solutions, LLC
http://www.cadanda.com

Integration of Design and Analysis Models

M. Sypkens Smit and W. F. Bronsvoort

Delft University of Technology, {m.sypkenssmit,w.f.bronsvoort}@tudelft.nl

ABSTRACT

Computer-Aided Design (CAD) and Computer-Aided Engineering (CAE) are disciplines with separate histories and consequently different conventions, data representations and workflows. In particular going from the design geometry to analysis geometry, let alone the other way around, can be a laborious task because of the separation between the two fields.
We briefly characterise the design and analysis model spaces, and discuss how the gap is commonly bridged. The exchange between the two spaces is still a significant bottleneck and worth attention, in particular with the practice of *simulation-based design* being employed ever more. We discuss five previously proposed solutions or 'visions' of how to better integrate the two disciplines, and contrast these with our own vision of the *analysis view* as part of the *multiple-view feature modelling* paradigm.

Keywords: design models, analysis models, CAD/CAE integration.
DOI: 10.3722/cadaps.2009.795-808

1. INTRODUCTION

CAD (Computer-Aided Design) and CAE (Computer-Aided Engineering) started out as two separate fields of both practice and research [11]. The first was to replace the practice of drawing designs by hand. The computer could offer higher precision in drawing, assist the designer with simple calculations for dimensioning, make it easier to experiment with variations of a design, and undo mistakes. It offered a more efficient and versatile drawing tool. The earliest instances of CAE occurred well before the start of CAD. It involved basically the numerical approximation of physical problems that were difficult or impossible to solve by hand. Since computers were relatively slow and numerical algorithms not as refined as they are today, the geometry of the problems that could be practically investigated was simple. Comparatively little time was spent describing the geometry of the domain.
The two fields matured separately over the decades and are now quintessential to product development. As both grew more complex, their underlying data structures evolved naturally. At the same time, however, the demand to perform analysis with CAD-developed geometries grew. Most products in development today are at some stage subjected to analysis. At that point, a geometric input suitable for the analysis is usually derived from the CAD-geometry. This is a laborious task that regularly has to be (partially) repeated. Though the design and analysis models in essence describe the same object, the models are so different that going back and forth between them is a clear bottleneck within the product development cycle. Trying to remove this bottleneck is, in effect, an effort to improve the integration of CAD and CAE models.
In this paper, we compare some of the approaches that have been proposed earlier to achieve this integration, and contrast this with our own vision of the *analysis view* incorporated into the paradigm of multiple-view feature modelling. We start with a description of design and analysis models in

Section 2, followed by a characterisation of how they are commonly used side-by-side in the product development cycle in Section 3. In Section 4 we compare five previously proposed approaches to improve the integration of CAD and CAE. We contrast these with the multiple-view approach, and in particular the analysis view, in Sections 5 and 6. Finally, the paper is concluded in Section 7.

2. DESIGN AND ANALYSIS MODELS

To better understand the issues concerning the integration of design and analysis models, in this section we characterise the two models and their common representations individually. The differences exist due to the different focus and perspective that the fields have of the represented objects.

2.1 Design Models

A design model is the primary data structure that is used in building a precise description of a product, most importantly its shape [13]. Most likely the designer will already have an idea of the shape when he starts to input the design model, but it is within the process of working with the design model that he makes the shape and all the dimensions exact. We further characterise design models by discussing the following two topics: 1) underlying geometry and topology, and 2) high-level elements for construction and interaction.

Design models need to be able to reflect arbitrary shapes and allow precise control over the shape, e.g. having a curve or surface pass through particular points, or the specification of derivatives of curves at certain points. The most common way to achieve this is by having curves and surfaces in the underlying representation that can be manipulated on an individual basis. To this end the curves and surfaces are represented by means of simple splines or NURBS. The curves and surfaces are all stored in a network of vertices, edges and faces that records how the geometric elements relate topologically to each other. Commonly the only geometric elements stored in the design model are those of the boundary representation (Brep), but some models store more elements in order to enable a more advanced modelling approach. For example, a cellular model [3] has been used to store semantics in addition to the geometry. If only a Brep is stored, then the topology is often required to be *manifold*; if a cellular model is used, then the topology can be *non-manifold*. The topology describes where geometric elements meet each other, which generally involves a certain *tolerance*, e.g. the bounding curve segments of two adjacent surfaces might not be strictly identical. This is because exact geometrical computations are not feasible with computers. Since the boundary of an object can be divided into a set of surfaces in many ways, the underlying topology of the design model says as much about the choices that were made in modelling the object, as about the object itself: identical objects can have different underlying topologies.

The low-level geometry and topology are not the only elements to constitute the design model, but they are present in almost all design applications. The designer, however, does not normally work with the geometry on this level, but has higher level entities at his disposal, such as features. Features represent parameterised shapes that are ultimately described by the low-level geometric elements. The design model is thus manipulated through the addition and reparameterisation of features, and as such they are the primary constituents of the model, whereas the lower-level geometry that comprises the Brep is derived from the feature model by boundary evaluation. Also, the designer can relate the parameters, such as size and relative position, of different features to each other by means of *constraints*. These are also part of the design model. Lastly, there can be a broad set of properties that do no directly relate to shape, such as material or color, associated with (parts of) the model. These are usually implemented by means of *attributes* which are linked either to the features or directly to the low-level geometric elements. They are also part of the design model, as they have to be properly maintained when modifying the model.

A design model thus consists of low-level curves and surfaces stored by means of NURBS or similar types of representation, a topological structure that describes how all geometric elements are related to each other, a set of parameterised features and constraints that implicitly describe the geometry of the model, and a set of attributes that is attached to certain features or low-level geometric entities.

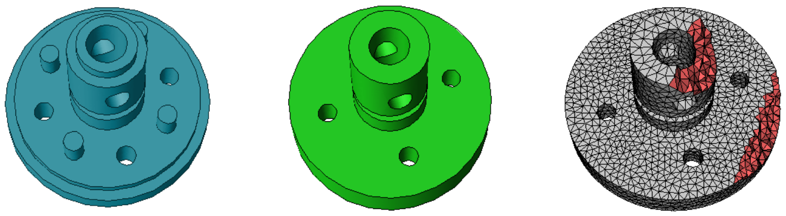

Fig. 1: Left: design model, middle: analysis model, right: tetrahedral mesh.

2.2 Analysis Models

An analysis model, in our treatment, is a geometry description on which an analysis is based. This is generally *not* the analysis mesh, which is generated from the analysis model. The analysis mesh, often a finite element (FE) mesh, is a decomposition of the analysis model into simple geometric elements such as triangles and quadrilaterals in 2D, or tetrahedra and hexahedra in 3D, that are instrumental in the numerical analysis. Fig. shows an example of a design model, together with a corresponding analysis model and a corresponding mesh. The distinction, however, between the mesh and the analysis model is not always clear. In some cases, the mesh is effectively the analysis model. In addition to providing the geometry, the analysis model can also store information to steer the mesh generation, and all other inputs for the analysis, such as initial values and boundary values [5].

The geometry of the analysis model is almost always an abstraction of the design product geometry, better suited for analysis, though the difference between the two can be small. Not all detail of the design is needed for the simulation of the physical behaviour of the real product. Too much detail might even reduce the global accuracy of the result, in particular when the result needs to be obtained in a limited amount of computing time. With too many geometric details, computational resources are devoted to sections that are irrelevant for the central question that the analysis is supposed to answer, with an increased likelihood that the fidelity of the result is worse. Analysis models thus tend to have a notably simpler geometry. It is even common for the geometry of the analysis model to be of a lower dimension than the design model, or, in case of symmetries, to only represent a subset of the model. For example, thin plates can be modelled by *shell elements* that have 2D geometry (embedded in 3D), and incorporate the thickness of the plate as a numerical parameter that affects the behaviour of the elements. This reduces the complexity of the calculation compared to full-dimensional analysis, without necessarily spoiling the fidelity.

The representation of the analysis model varies. In some cases it is a Brep composed of splines, similar to the representation of the common design model, but rarely can the geometry of the design model be used directly as the analysis model. Even if there is no need to reduce detail, then the meshing software might still be unable to deal with the geometry of the design model. The common alternative is a finely faceted (usually triangulated) model that has been generated from the design model. This faceted data structure is straightforward, so that it can easily be transferred and manipulated, and has no issues with tolerance, as only the topology is explicit. Algorithms that operate on this kind of representation are relatively easy to implement. A facetted Brep, however, is by itself not a useable analysis model yet. The geometry most likely needs to be adapted to make it more suitable for meshing, and be prepared for the assignment of boundary values and conditions. More about this in the next section.

The analysis model needs a topology that divides the surface into more or less smooth patches that are bounded by smooth edges and vertices. This is required by the meshing software for the creation of a good mesh. This topology also contains the faces, edges and vertices on which boundary values or boundary conditions are specified. Also part of the analysis model is information for steering the generation of the analysis mesh, in particular the mesh size. This is most often stored in a regular 3D grid or a 3D mesh, possibly the mesh of a previous analysis, but no real standards seem to exist as these data structures often have close ties with the particular meshing method.

3. CURRENT COMBINED WORKFLOW OF DESIGN AND ANALYSIS

Because the diversity in types of design and types of analysis is so large, there are several kinds of specialised environments that all operate (at least) slightly different [2]. There is thus no single,

complete characterisation of the *combined workflow* of design and analysis. We can only describe those aspects that are common and of importance to the integration issue. Furthermore, the combined workflow in current practice concerns mostly the derivation of analysis models from design models, which is essentially a one-way model conversion executed with a wide range of tools and techniques.
We are not aware of any survey research on the combined workflow of design and analysis in relation to the size and complexity of the models, the type or level of abstraction of the analysis models, and the type of simulations, as used in practice. We can only go by what scholars are indirectly suggesting about current practice by means of their research efforts, and our personal experiences. Starting with a design model, there are several ways to get to an analysis model:

- build the analysis model from scratch
- adapt the design model into an analysis model, sticking to the representation of the design model
- create the analysis model starting from a facetted Brep of the design model.

The first option is attractive if the geometry of the analysis model to be constructed is simple, at least compared to the design model. In this case recreating those few elements that are similar in both models is manageable. Creating a model from scratch by hand does, however, carry the risk of introducing errors. The engineer can make a mistake with the dimensions. Since there is no link between the two models, whenever the original model changes, the analysis model has to be updated by hand. Either the analysis model is updated each time a change is made, which is tedious, or after a range of modifications, which is difficult and error-prone.
The second option for creating an analysis model is to adapt (a copy of) the design model in its existing representation. This is attractive if the changes between the geometry of the analysis model and the design model are relatively small, e.g. if just a few design features need to be removed. When this is done by hand on a separate copy, there is no link between to the models. To maintain the relation, a specialised environment for performing the abstraction would have to be used, which seems uncommon. Also, there is more to be dealt with in preparing the analysis model than just feature removal. The model has to be made suitable for mesh creation, which means that a topology has to be specified and, more importantly, that the model should not exhibit any geometric characteristics that are detrimental for the generation of a quality mesh, such as very sharp angles or very thin faces. This is partly related to the tolerance issue.
The third option seems popular, considering the number of publications on the manipulation of faceted models for analysis idealisation purposes. A faceted model is generated from the design model, which is then adapted in steps to become a suitable analysis model. The model is made watertight, the surface is remeshed, feature size analysis is performed, and in a (partly) automated fashion the model is simplified. Though there is no real link between the design and analysis model with this approach, the advantage is that many steps can be automated. It is often noted though, in particular with respect to dimensional reduction, that the automated steps can give you a quick start, but frequently this will not give a result exactly as desired. Some manual tweaking seems unavoidable.
The distinction between these three approaches to obtaining an analysis model is not very sharp. There are approaches in use that are a mix of these, where, for example, first the design model is somewhat simplified, and then exported as a faceted model for further automated processing.
It is obvious that the workflows just characterised, have deficiencies with respect to efficiency and maintaining consistency.

4. APPROACHES TO CAD-CAE INTEGRATION

There have been several propositions, or *visions*, on how to make the combined workflow of CAD and CAE more efficient, or, in other words, to achieve actual CAD-CAE integration. We have selected five of them to serve as a basis for comparison and discussion of approaches to CAD-CAE integration. The approaches differ in level of technical detail, emphasis and general scope. Together they give a good impression of the spectrum of CAD-CAE integration approaches.

4.1 An Early Vision

The paper *Steps towards CAD-FEA integration* [1] from 1993 is one of the early publications to signal a need for more efficiency between the CAD and CAE process, and gives an overview of *"a future system which would allow and encourage more automated CAD-FEA transformation using tools that operate directly on the solid model."* The authors remind us that CAE has not always been performed upfront to

assist the design phase and to confirm that products will work as expected, but rather was used in the early days to investigate product failures a posteriori. Though there was effectively no integration at all at the time of writing, the authors already acknowledge the need to couple the design and analysis model: *"For a truly integrated CAD-FEA system, attributes must be applied to the design model so that they can be taken into account during the different stages of the idealisation process. A two-way link between attributes and geometry is required to be able to query the geometry with regard to its attributes, and attributes with regard to their geometry."* They also remark that: *"[...] it should be possible to relate analysis results back to the design geometry to allow for design modification/optimisation. Because of differences in the geometric domain [...], a simple reverse geometric transformation may not be possible. Formal links may therefore be required between design and analysis models [...] by the exchange of parameters."* These ideas are still topical today. In general, their view on CAD-CAE integration is similar in many ways to the current views expressed by scholars.
Their vision, however, is mostly conceptual. The need for various tasks, and for corresponding specialised tools, is recognised. They distinguish functional components such as an attribute editor, detail editor, dimensional reduction aid, and tools to preprocess the geometry for meshing. Fig. illustrates how the components fit together. They mention requirements for data structures and representations, but details are, not surprisingly, absent. In fact, they primarily perceived a lack of tools to properly derive an analysis model from a design model, and properly prepare it for analysis. Though they recognise the need to link both models, they may have underestimated how difficult this would prove to be.

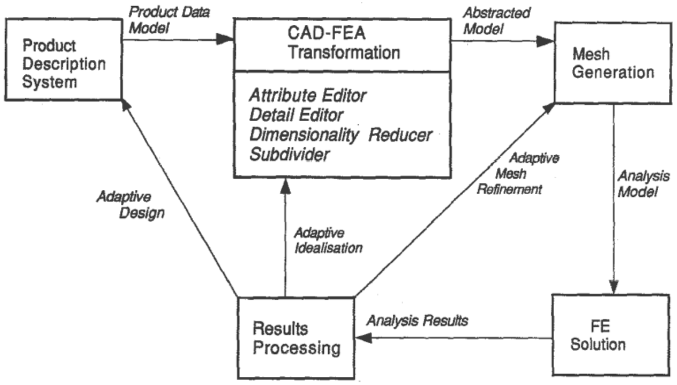

Fig. 2: Overview of the system proposed in [1].

4.2 Multiple Representations

In the paper *Integrating engineering design and analysis using a multi-representation approach* [15] from 1998, a *multi-representation architecture* (MRA) is presented. The evaluation and validation of the ideas was done in the specialised domain of solder joint fatigue routine analysis, but the proposed architecture is generic and can also be applied in other domains. In many domains, *routine analysis* is regularly applied, i.e. an analysis process that is performed (almost) by default for each design, has a well-defined series of input parameters and modelling elements, and requires little human creativity. Such an analysis usually focuses on a single aspect of the physical behaviour. Specialisation obviously makes it easier to achieve integration and implement a system with richer semantics. This enables the MRA to address the "*information-intensive nature of CAD-CAE integration*", which is the apt characterisation made by the authors.
The MRA consists of four components: 1) solution method model (SMM), 2) analysis building block (ABB), 3) product model (PM), and 4) product model-based analysis model (PBAM). There are inter-representation mappings between the SMM and the ABB, and between the ABB and the PM. The PBAM helps to bridge the gap between the ABBs and the PM. See Fig. for a schema of the architecture.
The PM effectively represents the design model. The design-oriented attributes are related to idealisation attributes to support the information needs of potentially many analysis models. The

analysis model, as we characterised it in Section 2.2, is here created by a particular PBAM, which mainly represents a particular physical model, through the combination of ABBs and creating fitting idealisations by means of the design-analysis associativity and the rules of the PBAM. The ABBs are elementary analysis primitives, such as a spring. They serve as product-independent elements of analysis models that can be instantiated and combined. Analogous to the design feature concept, we might consider them a kind of *analysis features*. The analysis model is thus an assembly of several ABBs as created by a PBAM. It can be converted into a solution method model, which can be a finite element mesh together with all the other inputs that an external, specialised analysis program might need. Identifiers are maintained along the path of mapping one representation to the next, such that the analysis results can be related back all the way to the design model, e.g. to identify in the design model for which components the maximal stress was exceeded.
These results demonstrate that within a specialised domain and with the modelling space sufficiently constrained, integration of design and analysis can be achieved to a reasonable degree.

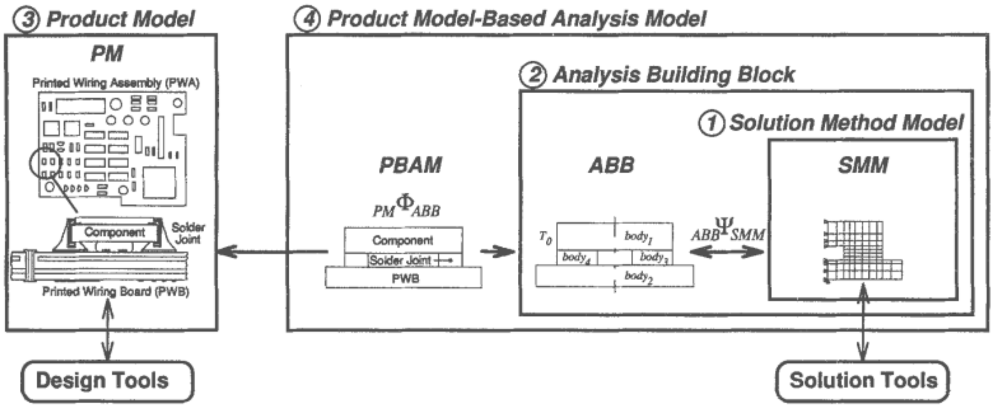

Fig. 3: The multi-representation architecture.

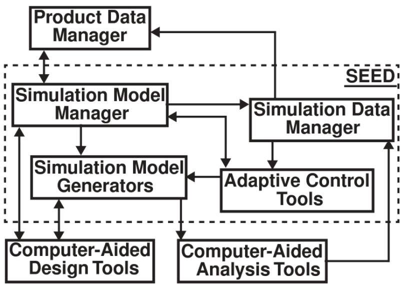

Fig. 4: The Simulation Environment for Engineering Design (SEED) within the product design cycle.

4.3 Simulation Environment for Engineering Design

In *Toward simulation-based design* [16] from 2004 the *Simulation Environment for Engineering Design* (SEED) is described, and how this integrates with CAD and product data management tools. This approach is quite generic, as it is not tied to particular types of analysis, modelling techniques or tools. There are four functional components of SEED: 1) simulation model manager, 2) simulation data manager, 3) adaptive control tools, and 4) simulation model generators. Fig. shows how the components of SEED interact with each other and with the other tools outside SEED.
The simulation model generators are responsible for the construction of the analysis model. They do this based on the input of the adaptive control tools, the simulation model manager, and the CAD system. The simulation model manager contains a high level functional view of the design to support analysis of specific subsets of components. In conjunction with the CAD system and the adaptive control tools, which decide on the need for idealisation of geometry, a complete geometric

representation for analysis can be generated. The simulation model manager also contains the topology associated with the geometric model. How the system deals with the creation and/or maintenance of this topology, in particular when subsets of components are analysed, seems to be an unresolved issue. The simulation model generators are also responsible for creating the mesh of the analysis model, based on cues from the adaptive control tools and the topology. The simulation data manager is responsible for the effective utilisation of simulation information within the design process. It does this by providing access to the analysis results, and communicating this to the user as a response that is effective and fitting in the context of the analysis problem.
The authors state that *"[...] the historic mistake made by both CAD and CAE developers has been to take the simplest view of this integration as a direct data transfer process."* The objective of SEED is to bring the design model and the knowledge involved in the analysis process closer together. With SEED, the traditional way of supplying the input to the CAE tools is effectively replaced by a layer of closely cooperating tools between the CAD system and the CAE tools that generate the desired inputs for the latter. The interacting components contribute and deal with knowledge important to the analysis on the one hand, and knowledge of the design model on the other hand. In the interaction of these components, the design and analysis models are brought closer together. Though this generic approach results in a high level of automation, the models are not actually linked. For instance, after a change to the design model, the process of preparing the analysis model has to be executed again.

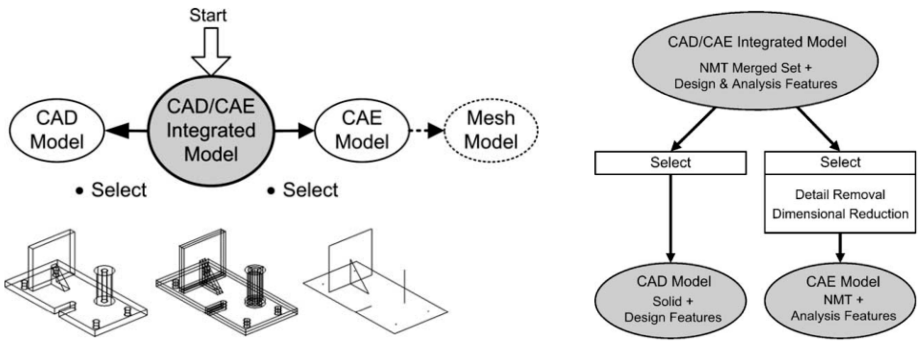

Fig. 5: CAD/CAE integration by means of a single master model.

4.4 A Single Master Model
The author of *A CAD-CAE integration approach using feature-based multi-resolution and multi-abstraction modelling techniques* [14] describes a feature-based system wherein both CAD and CAE models reside within a single master model. This is referred to as *full integration of CAD and CAE models*; see Fig..
The master model is stored as a non-manifold topology (NMT). A design starts with the Boolean combination of form features. For each feature, an idealisation feature is automatically added to the model. The geometry of all these features is stored together in the master model. When an analysis needs to be performed, the analysis model can be automatically extracted from the master model by the process of idealisation. This consists, firstly, of choosing a level of detail (LOD), by default eliminating features on the basis of the size of their volume. Secondly, for the resulting model the level of abstraction (LOA) is selected. The criteria for determining the LOA depend on the type of analysis. Due to the history-based modelling approach, to make the proper assembly of a subset of dimensionally reduced geometries possible, the features need to be rearranged, which is accomplished by the identification of the *effective zones* of the features.
It is stated that in the master model the geometries of all features of all abstraction levels are stored in the *merged set*, a structure that seems similar to the cellular model described in [3]. It is, however, unclear to what extent all geometric entitities of the features at different levels of abstraction reside in the same data structure. Obviously, when working with multiple geometric representations for a single feature, care has to be taken that efficiency remains acceptable, even for more complex models.

The underlying assumption of this approach is that an analysis model can always be formed by the individual idealisations of a subset of the design features. This will not always hold, since, for example, the geometry of some analysis models is based on global shape characteristics such as symmetry. The author acknowledges that the range and flexibility in dealing with the analysis model could be improved. Though the proposal is conceptually powerful, due to its very tight integration, it might not be generic enough to scale to complex and realistic cases.

4.5 A Mixed Shape Representation

The authors of *Interfacing product views through a mixed shape representation (Part 1 and 2)* [7,8] want to ease the interface between *product views* in general, by introducing a representation that can bring different data representations, basically CAD geometry and facetted geometry, closer together. They call this *a mixed shape representation*, and one of its applications can, obviously, be to bring the design and the analysis model closer together. The mixed shape representation mainly consists of a Brep NURBS topology and a facetted model that is organised by means of *Polyedges* and *Partitions*, and a *High Level Topology* (HLT) that links the CAD geometry and the polyedges and partitions. See Fig. for an illustration. The facetted model is a more natural data structure for the preparation of an analysis model/mesh, and by means of the HLT this representation also has information from the design model at its disposal, making certain operations for the preparation of an analysis model/mesh easier to automate.

The model or *product view* (PV) that is actively being modified is denoted the *master* representation, whereas the other is the *slave*. The manipulation of the mixed shape representation works by the application of operators, which modify this representation from the point of view of either the CAD or the facetted representation in a structured way, and can have semantics associated with them. The link between the two representations makes it easier to use reasoning based on geometric algorithms, e.g. feature size detection, and relate the facetted geometry to the NURBS model, and vice versa. Operations in one view can thus be aided by the representation of the other view. Though the link by means of the HLT is dynamically maintained under the operations that the authors define, in particular operations that prepare the geometry for analysis, it seems that the structure is not maintained throughout the modelling process, as not all operations that can be applied to a master representation can correspondingly be applied to the slave representation. The authors also acknowledge that the linking of dimensionally reduced models has not yet been dealt with.

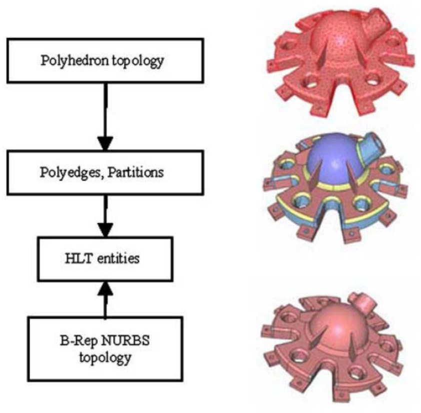

Fig. 6: Mixed shape representation.

4.6 Conclusions

The five approaches that we have reviewed all contribute to the spectrum of solutions for integrating CAD and CAE models, and through valuable observations give us insight into what can and what needs to be done.

One of the principal tasks in bringing CAD and CAE models closer together, is to bridge the gap between their different geometries. This gap consists of different levels of abstraction and detail,

distinct underlying representations, and requirements that the analysis model has to satisfy to make a suitable model to be meshed. In many operations that are commonly performed in the process of turning a design model into an analysis model, knowledge from both models is helpful. Performing a straightforward conversion from the one representation to the other, means losing the benefit that information from the other representation has to offer. At least throughout the process of deriving an analysis model from a design model, there should be a link that relates the two models to each other.
Creating, and in particular maintaining such link, has not been solved in a general fashion, and the lack of knowledge of the other model stands in the way of automation. In general, the incorporation of knowledge is essential for effective automation, as the modelling system needs to replace or assist human actions that require some degree of 'understanding' of the analyses, model types and common problems. Basically, anything that decreases the number of tasks and input choices for the engineer can be considered knowledge, e.g. the software knowing the meshing requirements for a specific analysis context, such as the need for a boundary layer mesh. Other examples of knowledge incorporation would be features that help to steer the abstraction process of their geometry, and coupling a facetted representation to a traditional Brep geometry, such as in [7,8], which opens up new ways to reason with the analysis model. Obviously, it is easier to incorporate knowledge for specialised domains or tasks, as it is easier to identify common tasks and to enumerate special cases.
Though automation is improving, there is no true linking of models in the sense that they are dynamically maintained under modification of the design. This means that, in particular with many, quick iterations of the design cycle, some steps will have to be repeated again and again. Also, making modifications to the analysis model, as might happen, for example, during design optimisation, and propagating those changes back to the design model, seems infeasible to automate with the proposed approaches.
In essence, we need to integrate two separate specialisations, design and analysis, that share a product model. Such integration has already been realised with reasonable success for several tasks other than analysis within the product development cycle. Complete integration of all technologies and processes involved in product development, is called *Product Lifecycle Management* (PLM) [17]. An important requirement for PLM is that a designer working in any product development phase can work with information that is relevant for that phase, without being diverted by information that is relevant for other phases. However, the information for all development phases should be integrated, so that no inconsistencies arise. We believe that a promising way to support this is *multiple-view feature modelling*. After having discussed, in this section, other approaches to integrating design and analysis models, or views on a product, we now look at how various views have been integrated in this modelling approach.

5. MULTIPLE-VIEW FEATURE MODELLING

Current commercial feature modelling systems do not adequately support multiple views on a product. Some systems have specific models for different development phases, but these models are not very well integrated. Other systems do integrate information for, for example, assembly design and part detail design, but these systems typically do not have specific models for these development phases.
Multiple-view feature modelling can do better here, by providing a separate view on a product for each development phase, and integrating all views. Each view contains a feature model of the product specific for the corresponding phase. Since the feature models of all views represent the same product, they have to be kept *consistent*.
Quite a lot of research has been done on multiple-view feature modelling during the last years [12,6,9]. These approaches all work on a single part of a product. A new view on a part can be derived from an existing view by feature conversion, the process of converting one feature model into another feature model. Several views on a part can be simultaneously maintained. Modifications made in one view, are automatically propagated to the other views on the part, also by feature conversion, although not all approaches propagate modifications in both directions. The approaches can be used to, for example, support design for manufacturing: while a part is being built with design features, these features can be immediately converted into features in a manufacturing planning view, which can be used to concurrently check the manufacturability of the product.
To even better support the concept of PLM, an approach to multiple-view feature modelling has been developed in our group, and implemented in the SPIFF prototype feature modelling system, that

supports more product development phases, in particular conceptual design, assembly design, part detail design, and part manufacturing planning [4]; this is illustrated in Fig..

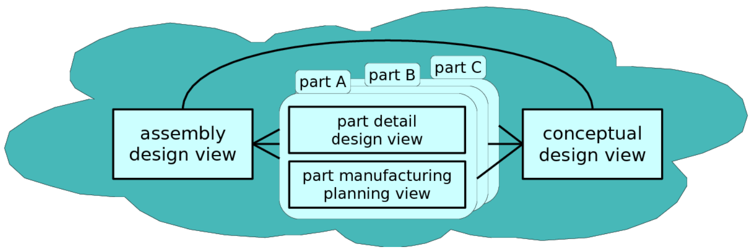

Fig. 7: Multiple-view feature modeling as currently implemented.

In the *conceptual design view*, the product configuration can be modelled with conceptual components, which are to be implemented by one or more parts, and interfaces between these conceptual components, which are to be implemented by a connection. Components are built from a base shape, concepts, such as depressions and protrusions, and reference elements. Interfaces between components are characterised by means of degrees of freedom between the components. The complete geometry of the components does not have to be specified: for example, for some concept only certain properties, such as its maximum volume, might be specified.
In the *assembly design view*, the connections between the components in the product can be modelled with assembly features, in particular *connection features* [10]. A connection feature needs to be created for each interface in the conceptual design view, and linked to that interface. The interface and the connection feature should reduce the same freedom. In order to accommodate the connection feature, features may have to be created on the components in the assembly design view, e.g. a pin feature and a hole feature for a pin-hole connection feature.
In the *part detail design view*, the detail shape of the parts can be modelled, typically with features such as through holes and protrusions. It allows the designer to refine the parts that are represented by the components in the conceptual design view, and which may have been refined in the assembly design view to accommodate connection features. Features may be created for concepts in the conceptual design view, and linked to those concepts. A feature should satisfy the requirements specified for the corresponding concept, e.g. that the volume should be less than 80 cm^3.
In the *part manufacturing planning view*, the way each part is to be manufactured is determined. Manufacturing planning features are similar to detail design features, but note that the set of features for the manufacturing planning view on a particular part can be quite different from the set of features that has been used to design the part, simply because a product can look different from the points of view of design and manufacturing planning. This view allows the designer to analyse the parts for manufacturability and to create a manufacturing plan for them. The feature model in a manufacturing planning view is automatically linked to the feature model in the corresponding part detail design view. See Fig. for the part detail design view and the part manufacturing planning view on a part.

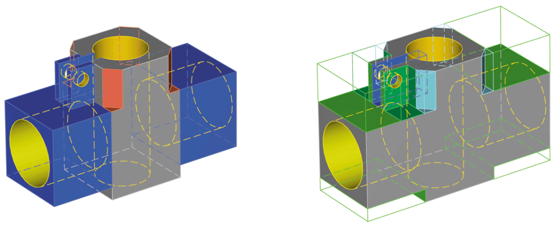

Fig. 8: Part detail design view (left) and part manufacturing planning view (right) on a part.

Consistency maintenance integrates all views on a product, by ensuring that their feature models remain consistent. It checks the consistency of pairs of feature models, based on consistency

definitions specified for these pairs. If an inconsistency is found, it recovers the consistency of the models. This involves, among other things, constraint checking and feature conversion in both directions [4].
In the multiple-view feature modelling approach described above, there are no analysis views yet, although analysis is an integral part of PLM. We therefore here propose to add analysis views to the multiple-view feature modelling approach, which leads to the integration of analysis with design, and other product development tasks, in a generic way.

6. ANALYSIS VIEWS

An analysis view in the multiple-view feature modelling approach should be a view of the product that is suitable for an engineer to perform analysis with. The view does not stand by itself, but is an integral part of the product model, and is maintained as such. The engineer, however, should be able to control the analysis as flexibly as in a completely separate analysis environment.
Putting the analysis model on equal footing with the design model is different from the approach of integration that considers analysis as a derived activity using a model with similar geometry. Making the analysis model an integral part, automatically entails the realisation of enduring consistency. This should ultimately be the most efficient, as every operation that is applied to a particular view and has an effect on other views as well, such as modifying shape dimensions, can be propagated automatically to the other views where the operation has effect; any information that essentially refers to the same model aspect, should not be entered or modified more than once. Consistency has to be maintained just as well when employing several separate environments, only the burden then lies with humans, and mistakes are therefore much more likely.
A product can have multiple analysis contexts, and within each context several views. A particular analysis view presents a physical model of the product that is best suited for that analysis context, e.g. joint fatigue, injection mold flow, or stress analysis. The view allows interaction with the analysis model, assignment of boundary values and boundary conditions, and control over miscellaneous parameters such as parameters within the physical model, parameters for mesh generation, and parameters providing control over the solution methods.
Having specialised analysis views for different analysis contexts based on the underlying physical models, allows for specialisation. This helps to bring the knowledge from specific analysis domains into the product model space, and allows for reasoning with that knowledge to support the creation and maintenance of analysis views and associated meshes in a context-sensitive manner. In the work discussed in Section 4.2, it is nicely illustrated how knowledge in a specific context can help to streamline the automation.
Since analyses are performed at various stages of the design, also when the detailed geometry is not yet complete, it should be possible to associate a range of analysis views, at different levels of (dimensional) abstraction, to a product model. Each view in turn can have multiple simulation runs associated with it, with possibly different meshes.
In Fig. a schematic overview is given of how multiple analysis views fit in our multiple-view feature modelling approach. There are several analysis contexts that represent different physical problems. Within each context, multiple analysis views can be created and maintained. Each view essentially represents a single analysis model and can have its own particular geometry. An analysis view has a mesh associated to it, and possibly the results of earlier analyses, including the mesh that was used in the analysis, and all other relevant parameters. An analysis view can relate to a single part or an assembly of parts.
A central characteristic of the multiple-view feature modeling paradigm is multi-directional propagation of modifications. With multiple views available, modifications of the model should be made *within the view* that prompts for these modifications, as that view offers the most natural interface. Not every modification in a particular view has effects on the other views, but if it does, then the modelling system should either know how the effect should be propagated to the other views, or let the engineer assist the propagation at an appropriate time. This should also hold for the analysis view. When the engineer, based on the analysis result, concludes that the geometry of the model should be modified, he should be able to do this through the analysis model. In particular in the case of analysis for geometry optimisation, where the result is a shape that is optimal in certain respects, it should be possible to propagate the result to the design view and other views.

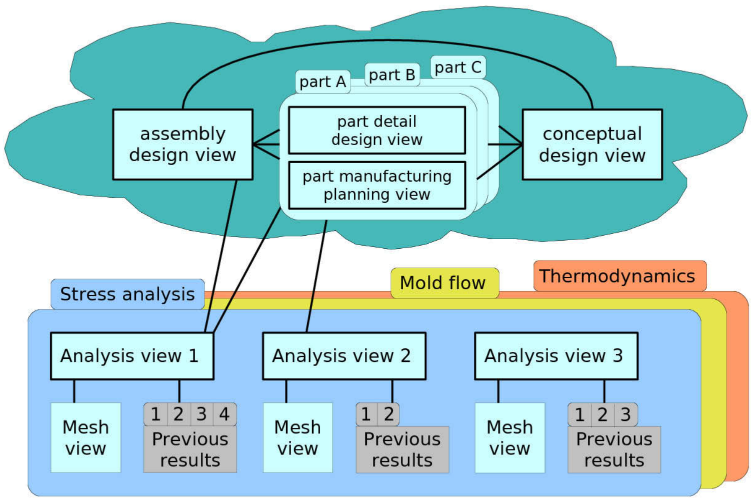

Fig. 9: Multiple-view feature model with the addition of analysis views.

Integrating CAD and CAE models by means of the multiple-view feature modelling paradigm, will maximally exploit the similarities and relations between the models. Since the geometry of the design and analysis model can have notable differences, the biggest challenge is to keep the models synchronized in an automatic way.

From the proposals discussed in Section 4 to improve the integration of design and analysis models, we can basically distill two approaches to bridge the gap between the two models: 1) convert the design model to an analysis model on the basis of individual features [14,15], and 2) establish a link based on more low-level geometric elements, and support the creation of the analysis model by operations along this path [7,8]. The first approach can work in specialised cases, but is not general enough to handle every type of geometry abstraction. In particular combining the geometry of multiple features into a single unit of abstraction is not supported, nor are analysis models that exploit symmetry in the model, and thus can be based on the partial geometry of some features. The second approach is more generic, but is harder to automate completely. More manual control and insight from of the engineer are required, as he often has to select and finetune the results of multiple, geometric algorithms. The ineffective automation that results from this approach can be partially addressed by relating the geometry of the design model and that of the analysis model in preparation, as then reasoning based on both representations can be used in the process of deriving the analysis model.

None of the approaches discussed in Section 4 are, however, well suited for actively maintaining the analysis model in combination with the design model. For instance, when the analysis model has already been prepared, it is generally not possible to make changes to the design model, and then have the analysis model updated accordingly, without the need to derive and prepare it yet again. Nor is it possible to make changes to the geometry of the analysis view and have these automatically propagated to the design view.

To achieve propagation of changes, it seems a prerequisite that the analysis view is a feature model, with parameterised features. The individual features in the design view should support the abstraction process where possible by containing options for abstraction, and possibly rules that help to automatically choose between multiple options or handle special cases of interaction with other features; to support complex abstractions, the relation between the features of the design view and the analysis view will not be one-to-one in all cases. Applying boundary values and conditions to the model should happen by means of analysis features, which demarcate areas, line segments or points within the analysis model. In addition to the parameterised representation, a more low-level geometry, such as a facetted representation of the model, could be associated to the analysis model and ease geometric reasoning, e.g. to determine local feature size throughout the model.

The two models consisting of features can be linked by means of operations, similar to the concept of operations for the transformation of models as used in [7,8], together with constraints. The use of constraints is vital for achieving the level of automation that we are aiming for. It is therefore a natural requirement that models, both design and analysis, are described in terms of features, and that it is specified how these relate to each other in terms of constraints. The constraints implicitly encode how the models can be varied, and thus enable propagation of changes with relative ease. Important research is how the relation of the constraints between the design and the analysis view should be managed, considering also that not all constraints that exist between features within a particular view, are valid in the other view due to the lack of a one-to-one relation between features in the two views. For instance, a constraint might refer to a feature that exists in only one of the views.

Input and maintenance of the models should, of course, not take more effort than what is common in deriving the analysis model. Views should certainly be kept synchronised, but to prevent too frequent checking, in particular amidst modifications, synchronisation may have to be delayed to suitable moments.

7. CONCLUSIONS

Attempts to improve the integration of CAD and CAE models already span multiple decades. The models differ not only in representation, but also in shape. Most efforts focus on automating the process to derive the analysis model from the design model. Drawbacks of this unidirectional approach are that the steps have to be repreated each time the design model has changed, and that changes to the geometry of the analysis model cannot be propagated back to the design model. These drawbacks can be addressed by extending the multiple-view feature modelling concept to the realm of analysis.

Integrating design and analysis models through the addition of an analysis view to a multiple-view feature modelling system, implies a more equal treatment of the two models: both models are maintained simultaneously and kept consistent. The requirement for consistency can be implemented by linking the views in such a way that either can be modified and that changes are propagated to the other. Research is ongoing on how such a link can be established and maintained.

Incorporation of knowledge of the analysis process and the requirements for the analysis model is essential for improving integration of CAE and the rest of the product development cycle. Depending, for instance, on the analysis context and the level of abstraction, different requirements hold. Product development systems should ‘understand’ when operations should be executed automatically, or else from which limited set the engineer should be offered a choice. Integrating design and analysis models by means of a multiple-view feature modelling approach, in which both models are maintained concurrently as views on a single product, provides a good way to incorporate knowledge on the transition between the two models.

8. REFERENCES

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

[2] Beall, M. W.; Walsh, J.; Shephard, M. S.: A comparison of techniques for geometry access related to mesh generation, Engineering with Computers, 20(3), 2004, 210-221.

[3] Bidarra, R.; de Kraker, K. J.; Bronsvoort, W. F.: Representation and management of feature information in a cellular model, Computer-Aided Design, 30(4), 1998, 301-313.

[4] Bronsvoort, W. F.; Noort, A.: Multiple-view feature modelling for integral product development, Computer-Aided Design, 36(10), 2004, 929-946.

[5] Chand, K. K.; Freitag Diachin, L.; Li, X.; Ollivier-Gooch, C.; Seegyoung Seol, E.; Shephard, M. S.; Tautges, T.; Trease, H.: Toward interoperable mesh, geometry and field components for PDE simulation development, Engineering with Computers, 24(2), 2008,165-182.

[6] De Martino, T.; Falcidieno, B.; Hassinger, S.: Design and engineering process integration through a multiple view intermediate modeller in a distributed object-oriented system environment, Computer-Aided Design, 30(6), 1998, 437-452.

[7] Drieux, G.; Léon, J.-C.; Guillaume, F.; Chevassus, N.; Fine, L.; Poulat, A.: Interfacing product views through a mixed shape representation. Part 2: Model processing description, International Journal on Interactive Design and Manufacturing, 1(2), 2007, 67-83

[8] Hamri, O.; Léon, J.-C.; Giannini, F.; Falcidieno, B.; Poulat, A.; Fine, L.: Interfacing product views through a mixed shape representation. Part 1: Data structures and operators, International Journal on Interactive Design and Manufacturing, 2(2), 2008, 69-85.

[9] Hoffmann, C. M.; Joan-Arinyo, R.: CAD and the product master model, Computer-Aided Design, 30(11), 1998, 905-918.

[10] van Holland, W.; Bronsvoort, W.F.: Assembly features in modeling and planning, Robotics and Computer Integrated Manufacturing, 16(4), 2000, 277-294.

[11] Hughes, T. J. R.; Cottrell, J. A.; Bazilevs, Y.: Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, Computer Methods in Applied Mechanics and Engineering, 194(39-41), 2005, 4135-4195.

[12] de Kraker, K. J.; Dohmen, M.; Bronsvoort, W. F.: Maintaining multiple views in feature modeling, Proc. of Solid Modeling '97, 14-16 May, Atlanta, USA, Hoffmann, C. M. and Bronsvoort, W. F. (Eds.), ACM Press, New York, 1997, 123-130.

[13] Lee, K.: Principles of CAD/CAM/CAE systems, Addison-Wesley, USA, 1999.

[14] Lee, S. H.: A CAD-CAE integration approach using feature-based multi-resolution and multi-abstraction modelling techniques, Computer-Aided Design, 37(9), 2005, 941-955.

[15] Peak, R. S.; Fulton, R. E.; Nishigaki, I.; Okamoto, N.: Integrating engineering design and analysis using a multi-representation approach, Engineering with Computers, 14(2), 1998, 93-114.

[16] Shephard, M. S.; Beall, M. W.; O'Bara, R. M.; Webster, B. E.: Toward simulation-based design, Finite Elements in Analysis and Design, 40(12), 2004, 1575-1598.

[17] Stark, J.: Product Lifecycle Management – 21st Century Paradigm for Product Realisation, Springer-Verlag, London, 2005.