AN HISTORICAL SURVEY, CONTEMPORARY ASSESSMENT AND RECOMMENDED DIRECTIONS FOR FEA PRE AND POST PROCESSING

KEYNOTE SPEECH, 2nd. CONFERENCE ON ADVANCES AND APPLICATIONS OF GID 2004

David Weinberg[*] and Tony Abbey[†]

[*] CEO, Noran Engineering Inc.
5555 Garden Grove Blvd.
Westminster. CA. USA
e-mail: david.weinberg@noraneng.com , web page: http://www.nenastran.com

[†] Technical Director, Noran Engineering Inc.
5555 Garden Grove Blvd.
Westminster. CA. USA
e-mail: tony.abbey@noraneng.com , web page: http://www.nenastran.com

Key words: Pre Processing, Post Processing, Meshing, Geometry, CAD

Abstract. *This paper presents a very brief historical survey of pre and post processing products available to the FE analyst over the last thirty years. Some of the trends that have evolved over this period are highlighted and discussed in the context of a modern day pre and post processor.*

An assessment is made of the general range of modern tools to deal with the enormous growth in FE model size and complexity, together with the spread of FE usage from the traditional dedicated analysts to the designer/analyst.

Sample areas of functionality are discussed in more detail and a 'wish list' of attributes that the authors consider essential for the future are presented.

The objective of this paper is to stimulate discussion about modern pre and post processor requirements across a very broad range of topics. The central theme is that the user must be able to achieve the most suitable mesh with associated loads, boundary conditions and analysis set up, in the most efficient and simple way. The user must then be able to assess the results of the analysis, whatever the scale and complexity, so that the essential engineering results can be investigated – with a tool that the user is hardly aware he is using.

1. Review of historical development:

Prior to 1980 there was very little manifestation of the interactive GUI we now associate with a modern pre-processor. Visual Display Unit (VDU) development was in its infancy, screen Graphics, local processing and other required technologies had barely been addressed. The type of menu and layout we are now so familiar with, based largely on Apple Mac type ideas, was unknown.

Most automated mesh generation was based on batch commands and processes and commands within the solvers or closely aligned tools. External geometry import was not feasible as CAD was also in its infancy. The CAD technology at that time was strictly 2D, being basically an electronic mapping of the traditional drawing projection techniques. No true 3D entities existed.

A large proportion of the mesh creation was done by hand, resulting in a very labor intensive process. Coupled with the cost of computing hardware, high software costs and a slow take up of FE technology, this resulted in FE Analysis being largely restricted to safety critical or technology driven industries such as Aerospace, Nuclear and Maritime. The attention to detail and idealization skills required in these areas meant that the FE analysis route was a slow, highly specialized and highly skilled arena.

Initially two classes of preprocessor emerged in early 1980's both having the clear objective of improving the productivity of mesh creation. The first method, typified by SDRC SUPERTAB, took a very simple approach and dealt only with pure FE based entities, nodes, elements, loads, boundary conditions etc. The mesh was created explicitly by a range of direct methods, sweeping methods and transformation methods. The second method was typified by the PDA PATRAN type approach which used two stages to create the mesh. The first stage was to build geometry within the preprocessor and the second stage was to create the mesh on that geometry. A strict association of geometry to mesh and to all other FE entities such as material properties, physical properties as well as loads and boundary conditions was made.

The advantage of the FE only approach was that it was very pragmatic. The end objective – a mesh – was clearly in sight at all times. Mesh fixing, breaking and direct editing were standard methods. The main disadvantage was the complexity of the model was limited (i.e., only so much could be built with mesh based extrude, sweep, etc.)
The advantage of the geometry based methods was that they gave a very powerful way of associating other entities via the geometry to the mesh. This allowed easy remeshing, complex load and boundary condition application etc. The main disadvantage was that creation and manipulation of the geometry could become time consuming and there were no easy ways to create mesh entities directly. To this extent the 'purist' geometry based approach became could become a distraction where the focus was on building geometry rather than creating a mesh.

The working environment in this period was the same for both SUPERTAB and PATRAN. It was a command line driven approach with graphical display (wireframe only) of the model. The typical user was still an FE specialist.

It is also important to note that the concept of a preprocessor 'database' was not established at all. The archived FE data was almost universally the analysis 'deck'. In fact it is still the archive format in many cases today! The reasons for this were various, including natural distrust of a binary database, storage requirements

The mid 1980's saw a rapid increase in graphics technology, both in the traditional 'dumb' terminal environment of the main frames and mini computers and the emerging PC industry. This meant that novelties such as color, animation and 3d shading could now be used increasingly in the preprocessor environment. Compared to today the cost was enormous and the technology slow and crude, but the foundations of today's GUI driven approach were emerging.

A vital development during this period was the progress in 3D CAD and the creation of the early data interchange formats including IGES. For the geometry based solutions such as PATRAN this gave a clear advantage as they conceptually had the means to take advantage of CAD data. In practice the early IGES support was notoriously bad, so a surprisingly small amount of data transfer was made in practice.

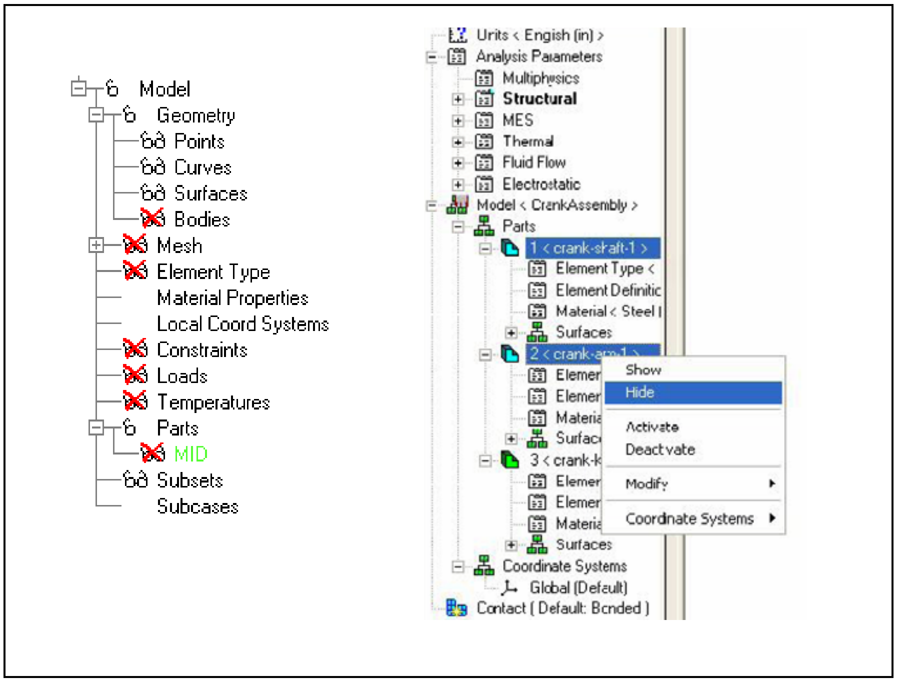
Continued development in all technologies through to the early 90's allowed the development of new and powerful GUI's, new database techniques and a much improved interface with CAD.

These developments also brought new challenges. The well accepted command line approach had to be replaced by a GUI which would not confuse the existing user and would be simple for the new user. The reliability of the database would have to be maintained as the GUI and architecture were modernized. In retrospect the command line approach probably eliminated in haste and there may well be a place for a modern alternative to complement the GUI approach.

But the biggest challenge was and still remains, how to deal with CAD geometry. SUPERTAB took the route of wrapping a complete CAD environment around itself and emerged as the MASTER SERIES suite. PATRAN took the stance that its geometry tools were simply a means of fixing geometry and would only be used for geometry creation in the case where no CAD data existed. Both products suffered considerable teething pains as they moved into the modern paradigm. It was clearly a difficult task for example to be a pioneer of the modern GUI form and get the 'look and feel' just right first time. Subsequent products have benefited greatly by learning from the mistakes of the leading pre and post processors of the early 90's.

Over the last ten years a very significant spread in the range of FE users has developed. Instead of being limited to the realm of highly skilled analysts, many other engineers are using FE technology as part of their everyday process. The need to address the needs of the full range of users is a very significant challenge for any preprocessor.

Many other preprocessors are in existence now besides the two we have classified and concentrated on as the pioneers. To a large extent many of these other products are moving forward with the latest generation of preprocessor technology with more sophisticated user interfaces such as object trees, drag and drop methods, multiple undo levels, etc. A large number of these new methods coming into use in preprocessing products are based on the paradigms seen in CAD products. This provides a dual motivation for preprocessor products to use these techniques. They are proven, and they



are standard methods for the new range of CAD based users. Figure 1 illustrates some of these ideas

Figure 1. Modern Object Tree Menus

There is also a significant increase in the number of imbedded products where the preprocessing is carried out completely within the parent CAD product. This paper takes the current scene as the starting point, what is good right now, and what are we looking for in the future. We also look at post processing in the same light.

THE ROLE OF CAD

This is one of the most important design factors in any preprocessor. The previous section has illustrated the steady migration of CAD and FE towards a unified approach. The basic problem remains however that CAD geometry is not set up exclusively for the FE analyst. It serves many purposes in the design process before being passed to manufacturing, quality control and all the other discipliners involved in a product manufacture. The requirements of FE are very different from other disciplines. We require geometry which is reduced to the simplest possible definition of the structurally significant part of the product. In fact in many cases we do not know what the 'structurally' significant part is.

The level of cooperation between the designer and the analyst is vital. In the worst case scenario company policies and tradition will result in a situation where the FE analyst has to take what he is given. The best case scenario, which should give encouragement to many designers entering FE, is if the analyst and CAD generator is the same person.

What are the difficulties with CAD? A short list includes the following:

- Over complex geometry which does not allow a simple idealization of the structure.
- Redundant features such as tooling holes, fillet radii, draft angles etc.
- Dirty geometry resulting from tolerance mismatch, CAD errors, etc.
- Unnecessary slivers and complex shapes which are difficult or impossible to mesh
- Incompatible geometry math definition between CAD and preprocessor

The ideal solution to all these problems in many respects is to avoid the use of a third party preprocessor altogether and have all the process through to mesh creation in the CAD product. This ideal is not met in practice as the CAD products simply do not yet have the quality and depth of functions required to create FE data for all but the simplest analysis. Imbedded products come closest to this for simple analyses. Alternatively a direct connection between the CAD program and the pre-processor that eliminates the need for export/import is attractive.

Never the less it is true to say that the best place to 'fix up' any CAD problems is in the CAD package. Hence the statement that the best case scenario at present is the analyst who is using the CAD package and the FE product in parallel.

In many cases due to lack of communication or availability of CAD resources a preprocessor is used to 'fix up' the geometry. This is an area where current preprocessors are still weak in general. One observation here is that too much attention is paid to the geometry in isolation. Bearing in mind the objective is a robust mesh, then more modern approaches need to consider the mesh requirements and to take a more pragmatic approach. It should not be a two stage method – fix the geometry, then mesh. It should be a one stage method – assess the geometry and required mesh and then mesh using whatever means (fair or foul from the 'purist' geometry perspective!). Ideally this process should be automated as much as possible so that the designer/analyst is not required to intervene in dealing with bad geometry.

Another perspective on the CAD-FE relationship is the growing need for rapid virtual prototyping, Many industries can no longer afford the parallel existence of CAD and FE models that do not 'talk' to each other. When a change is dictated by FE results and the design is modified this must be reflected immediately in the CAD model. This approach brings many challenges in areas of parameterization and version control. However it seems the lead is being taken by CAD products and not by the preprocessors. Independent preprocessor development is required to address these issues.

PREPROCESSOR OBJECTIVES

If we assume there is a role for an independent preprocessor outside the CAD product domain then the fundamental question to ask is; what is the objective of any preprocessor?

The objective must be to enable the FE analysis to be set up as accurately and efficiently as possible. This means we require rapid, accurate and robust meshing. As much as possible of the meshing process must be automated – but controllable. The mesh generated by any mesher must avoid shapes and orientations that will give poor results or fatal errors in the solver. Figure 2 shows some simple meshing cases where good mesh control has been achieved.

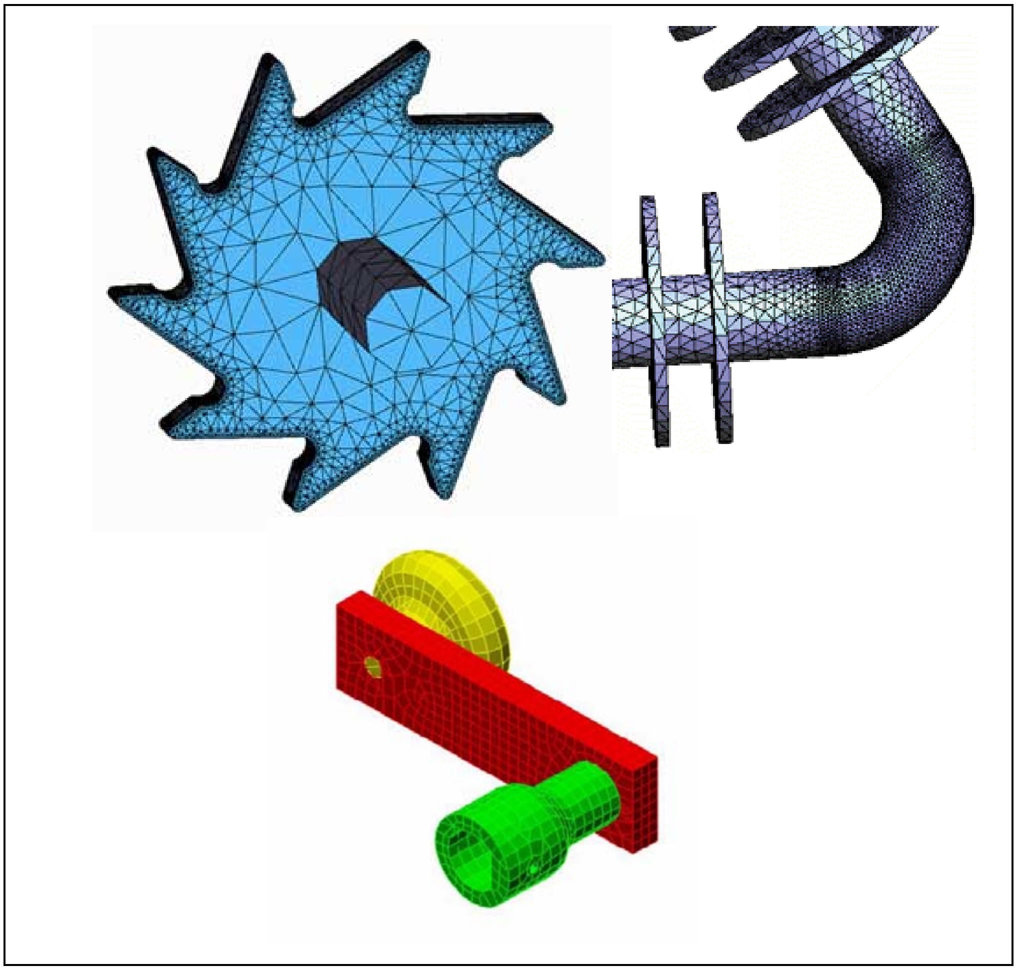


Figure 2. Clean Simple Mesh Examples

The current range of products on the market show surprising weaknesses considering the importance of the requirement. The main reasons are that most preprocessors are not solver specific and have a generic approach to FEA QA. The mathematics of many of the geometry engines are fundamentally incompatible. Many products are a collection of incompatible tools rather than a single unified toolbox. The weaknesses that hinder the fundamental objective include:

- Geometry requires labour intensive investigation and correction before a mesh is achievable.

- Mesh is not checked for analysis specific geometry and orientation requirements for distortion, aspect ratio, warp, Jacobian value etc.

- Mesh is not congruent across adjacent geometry without special attention.

- Different geometry types or creation methods give different and incompatible meshes.

- Very hit and miss approach in getting a 'good' mesh which is solver legal and smooth.

- Controls on the interior of a mesh are generally weak. We need far better controls to enable regions or features with coarse or fine mesh requirements to be defined.

- No true Hex-meshing. This is the 'holy grail'. Many approaches exist such as hex dominant, hex rich, mixing incompatible elements or just hiding the bad interfaces deep inside the solid mesh. The argument is that worst stresses are always at surface – but can we can get contamination of stiffness matrix by poor singularity.

PERFORMANCE

The performance requirements of a modern pre and post processor are extremely demanding. The key areas to address are:

- Ability to generate, store and manipulate and graphically present meshes running currently into 500,000 to 1,000,000 elements. This number will continue to grow as more computing power and more efficient solution processes are available to the analysis solver – and hence allow the user to build more complex models.

- The same abilities are needed to deal with result quantities from these very large scale analysis. Consider Frequency Response or Transient analyses where the user is demanding higher fidelity of results – resulting in vast quantities of data, and also demands interactive presentation of this data via screen picking etc.

- The inability of the pre/post processor to deal effectively with large amounts of data in the database is becoming the major bottleneck in some industries. The graphics overhead of calculating and drawing 'hidden line' or shading is enormous.

- The size of a model database is also becoming a significant issue in some cases. Very efficient storage and updating methods are required. Database 'bloat' due to numerous creation and deletion operations of mesh and results degrading the database storage packing also needs to avoided.

THE USER FRIENDLY GUI

As stated in the abstract, the ergonomics of the pre and post processing tool are vital. One of the easiest ways to observe the success or otherwise of a preprocessor is to teach a class of engineers who are migrating from another product. There will naturally be resistance to change – of varying degrees! The rate at which the new process is absorbed and that students gain confidence in its application, is directly related to the ergonomics, consistency and 'common sense' of the GUI design.

One of the basic principles of successful preprocessing is the concept of having the right view with the right entities visible and a straightforward menu ready. Unwanted distractions are kept to an absolute minimum so that the mental and physical tasks of performing the operation are made as simple as possible.

The idea of the ergonomics or the 'look and feel' of the GUI is examined with relation to several vital elements of the preprocessor.

WORK FLOW

A logical workflow, which reflects the usual FE model creation process, should exist and be very clear in the GUI. This would mean that menu items such as CAD Import, Geometry Operations, Meshing, Material Properties, Physical Properties, Loads, Boundary conditions and Analysis Set Up should all be present in a top level menu.

This is a straightforward approach and most products follow this idea. For a basic imbedded CAD product, not much else would be required and in fact the first two items would be parent CAD product functionality.

However, for a general purpose preprocessor many supplementary tasks are required. A list includes entity editing, deletion and copying across all Geometry and FE entities. Most models will require modification and correction as the model and subsequent analyses evolve. This is where the simplicity of the workflow approach can become lost. In some products these supplementary processes are split between global functions, which operate on all entity types, and local operations which are accessed via the entity menu. This immediately causes confusion for the user. Consider an example task below:

- Where does the engineer go to in the menu to perform a specific task such as modifying a load definition.
- Does the user go to a global Edit/Modify function or to a Load menu?
- Is there a way of editing all the nodal loads in a large model at the same time to adjust the magnitude of load.
- Can the user modify a subset of nodal loads?

This is really one of the key measures of success of a GUI, that there is a consistent pattern to these supplementary but vital operations that a user can rapidly adapt to.

The design of menus is vital. There is a wide range of approaches to this in contemporary products, ranging from very elegant forms that allow a user to focus on the essential tasks needed, to forms that have every option crammed into a single busy form. Ironically the latter may be more useful as the user gains experience if it presents him with fewer menu levels and a quicker more pragmatic way of getting the job done! Figure 3 shows a very simple and intuitive menu for controlling mesh density as an example.

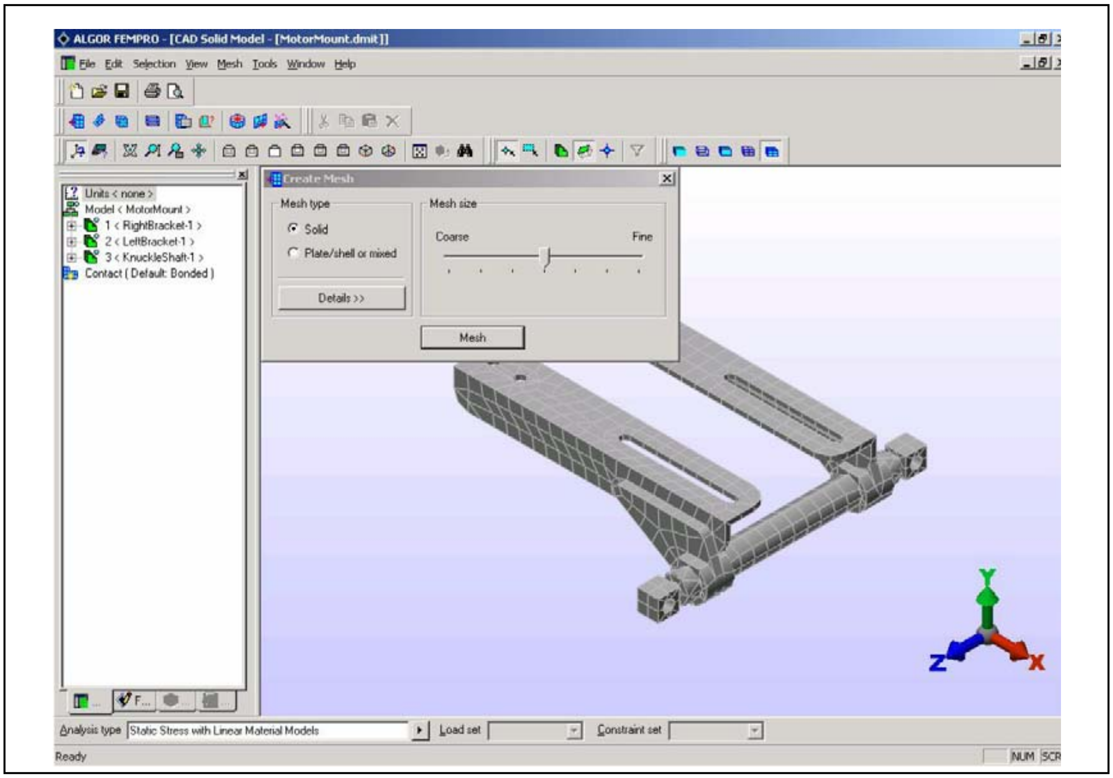


Figure 3. Simple slider bar for mesh density control

Typical menu issues are:

- Ambiguous or unclear objectives of the form. What job am I supposed to be doing here?

- Very busy and confusing forms.

- Very sparse forms that seem to be a waste of space or unnecessary.
- Physically long forms requiring a large amount of scrolling or where the overall content should be really be seen in one view to understand the scope.

- Inconsistent or counter intuitive use of ‘OK’ versus ‘Accept’ versus ‘Cancel’ versus ‘Apply’ where it is not clear what the action is doing.

- Excessive levels or depth of Forms.

- A Form containing cross reference data which must be closed before another Form requiring that data can be opened.

- Cryptic text on forms, or verbose text on forms.

- Forms which have icons for decoration rather than function.

Many current products have forms, menus etc.do not have a consistent approach to the basic questions:

- What task am I trying to achieve

- Where do I access the right menu for that task

- How do I achieve that task using this menu

ENTITY PICKING

This is a simple task at first glance. However, consider a very complex model which has hundreds of thousands of elements and multiple components. Assume the task is to select a single independent node to connect to multiple dependent nodes to form a rigid element. The challenges can include:

- Coincident or near coincident nodes need to be filtered properly to select the correct ‘target’ nodes.

- Multiple nodes are in line of site of ‘target’ nodes and can easily be picked in error.

- Nodes are required from multiple component regions.

- The target nodes do not form a simple pattern and they may or may not associated with geometry.

The key to these challenges are to provide the GUI with a flexible picking technique which is controlled from the current operation without needing the user to lose focus on the main activity – picking the nodes. Good solutions include the following, which could form part of pre-processor specification.

- A selection buffer or collection box which will permit multiple instances of picking.

  This means that the view can be changed or the picking technique can be changed whilst still accumulating the list of entities. A final ‘commit’ button is used when the picking task is complete and the buffer contents can be committed to the database.

- Multiple ways of region picking:

  Polygon
  Rectangular drag
  Circular and elliptical drag

- Front Face option

  This means the mesh will be viewed as a ‘hidden line’ object and interior nodes in line of sight will not be picked.

- Entity Association

  This means association via convenient Elements, Material or Physical Properties, Loads, Boundary Conditions or Geometry.

- Group Association

  This means association via convenient Groups

- Geometric position

  This means a locus of nodal positions defined by geometric global or local x,y,z coordinates of the forms ‘greater than’, ‘less than’, ‘between’ etc.

- Coincident Listing

This means when multiple nodes are present within a defined geometric tolerance, then a list of the instances is presented to the user to enable the correct selection to be made. The tolerance should be capable of adjustment on the fly.

- Existing selection indication

This means the entity will not be re- picked if it is already in the selected list. A visual indication such as failure to highlight will make this clear.

- List Highlighting

This means the current list will be highlighted to assist confirmation of correct selection.

- Entity labeling on the fly

This enables an ID to be confirmed without postponing the picking process.

- Multiple actions on picking

The default action is to add each sub selection to the buffer or collector box, but an option should be available to delete the current sub selection from the buffer or to empty the buffer.

- Post to a group on the fly

A new or existing group can be defined to send the current node selection to.

- Use a previous list

The previous list in the collector buffer may be appropriate for re-use.

- Drag and Drop of the Object Tree

This means the Object tree can be used to make the selection in parallel with the graphic screen methods described above.

- Clipboard Cut and Paste

Multiple windows clipboards should be available to transfer node lists from previous captures within the preprocessor or external windows applications such as excel, word or text files.

- Use a previous view or list

Repetitive tasks can be simplified by toggling between previous views and entity lists

If we consider element picking then we have some additional requirements:

- Adjacent Element picking

This means that elements occupying a common plane as defined by an angle criterion can be picked via a target element. This is essential for contact surface definition, pressure loading etc.

- Picking by Attribute

This means picking by element type, topology or shape.

GROUP DEFINITION

Anything other than a trivial model will require a means of grouping entities into logical sets. This is primarily for three purposes:

- Enabling the model to be broken down into groups based on appropriate physical components, entity type, entity attribute etc. This is basically to perform housekeeping on the model so that it can be checked and manipulated easily, including further Boolean operations.

- Using groups to create further entities by copying, sweeping etc. This is an extremely powerful technique which allows for rapid model creation in many situations. All the group associations such as loads, material properties should also be capable of optional replication.

- Using groups as a selection option in entity picking, load or boundary condition assignment, material or physical property assignment etc.. This again is a very powerful way of enabling many operations to more efficient.

The following list of desirable functionality will clearly show the extent to which groups can be used productively.

- An entity can be in any number of groups. Thus a beam element can be in a group called ‘BEAMS” which permits easy visualization of all beams. It can also exist in “FRAME32” which permits manipulation and visualization of the component in forms part of.

- Any single group can be ‘current’, meaning that whatever new entity is created will be placed in that group. The action to place new entities into a current group should be capable of being disabled.

- Any number of groups can be ‘posted’, meaning that they are visible. The option to view by ‘posted’ groups or all entities should be selectable.
- All usual Boolean operations should be available for groups.

- When using groups to create new entities by copying the options should be available to recreate or modify loads, boundary conditions, properties etc. as appropriate.

- Multiple levels of groups should be available to allow the concept of ‘super groups’

- Groups will be visible as objects in an object tree. Drag and drop operations will allow Boolean operations on groups and listing and manipulation of entities between groups.

- Automatic group creation will be possible based on entity type, attributes, associations, result values, geometric position etc.

- Building and disassembly techniques such as ‘peeling’ outer layers of assemblies of elements will be possible

- Color coding and shading by group.

- Automatic grouping based on feature, position, properties, materials

VERTICAL APPLICATIONS

Most preprocessors have a strong history of macros, APIs or programming languages that have allowed the user to extend the range of applications and to tailor to his own specific requirements. The most potentially powerful applications use Microsoft VBA as the API language, bringing standardization and maturity of use to bear. This is becoming an essential requirement for a preprocessor. Another recent move has been to include many commonly used vertical applications such as mesh refinement around holes, bolt group idealization and other useful tools in basic suite of tools. This is seen as a vital development and is clearly based on evaluation of what the tasks are that the typical user needs to achieve. It is sobering to think that it has taken this time to develop functionality to idealize bolt hole fixation or loading automatically! Figure 3 shows a typical tool now available which allows the user to create a rigid connection to the center of a bolt hole ready for connection or loading.

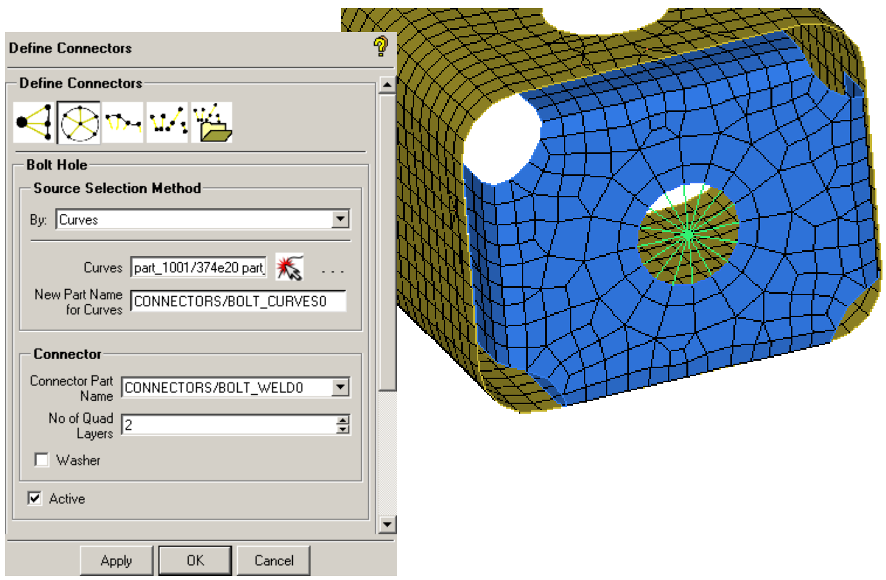


Figure 3. Automatic Bolt connectivity creation tool.

By their nature vertical applications will tend to migrate to industry or discipline based component products within the parent product. Instances of this are seen already. We look forward to a more radical approach in the future so that, if the need is to set up advanced dynamic analysis for a specific analysis code, then the preprocessor can focus on that requirement. If a user needs to change focus to deal with non-linear analysis then the preprocessor can adapt itself to provide an environment dedicated to tasks required in that field.

CONCLUSIONS

We have covered a range of pre and post processor requirements that are considered essential for a modern tool. The list is not exhaustive, but focuses on some important areas that from our experience need to be improved upon. We have not attempted to identify individual software products with particular strengths or weaknesses as to do so would be unfair in such a fast developing field. However most users will recognize the trends described to a lesser or greater extent in all contemporary products.

We present ideas for the future based on end user experience over many years with many tools and hope they may provide guidance and insight to developers and users.

ACKNOWLEDGEMENTS

The authors acknowledge the image copyright of the various software vendors who have provided pictorial examples of innovation or best practice.