DESIGN OF AN INTELLIGENT STRUCTURAL QUALIFICATION ENVIRONMENT USING MSC/PATRAN

N.J. Dullaway, A.J. Morris

Structures & Materials Group, College of Aeronautics, Cranfield University, England.

ABSTRACT

Increasingly, modern structures are becoming ever more complex, large and expensive particularly when large full scale or similar qualification tests are required. This is particularly true when the structure being designed is safety-critical. In addition, questions are now being asked about the ability of conventional test practices to adequately qualify and validate new structures. This is a situation which is causing concern in a number of industrial domains including aerospace, maritime and civil engineering.

The arrival of computer-based analysis, particularly finite element analysis, has provided the ability to reduce reliance on conventional, or "real", testing and instead go down the path of "virtual" testing. However, virtual testing raises the question of the reliability of analysis and the possibility that the use of poor procedures in the analysis process may produce results that at best are meaningless and at worst are extremely dangerous.

This paper describes SAFESA™ (SAFE Structural Analysis), a research project to develop a computer-aided engineering environment for automated structural qualification in a range of domains by means of virtual testing. This application is built on the MSC/PATRAN & MSC/NASTRAN platform and implemented using PCL as the development language.

INTRODUCTION

In the context of this paper *virtual testing* is defined as qualification by means of analysing a mathematical model of the subject structure; *Physical testing* is the more traditional method of qualification whereby an actual example of the item is subjected to various environmental conditions and its behaviour measured. Common examples of contemporary tools available for virtual testing purposes are Finite Element Analysis (FEA), Computational Fluid Dynamics (CFD) and Computational Electromagnetics (CEM). One of the key advantages of virtual testing is that it enables the design to be evaluated and validated before manufacture thus promoting the use of concurrent engineering methods as well as reducing the design-to-development costs.

Test Paradigms

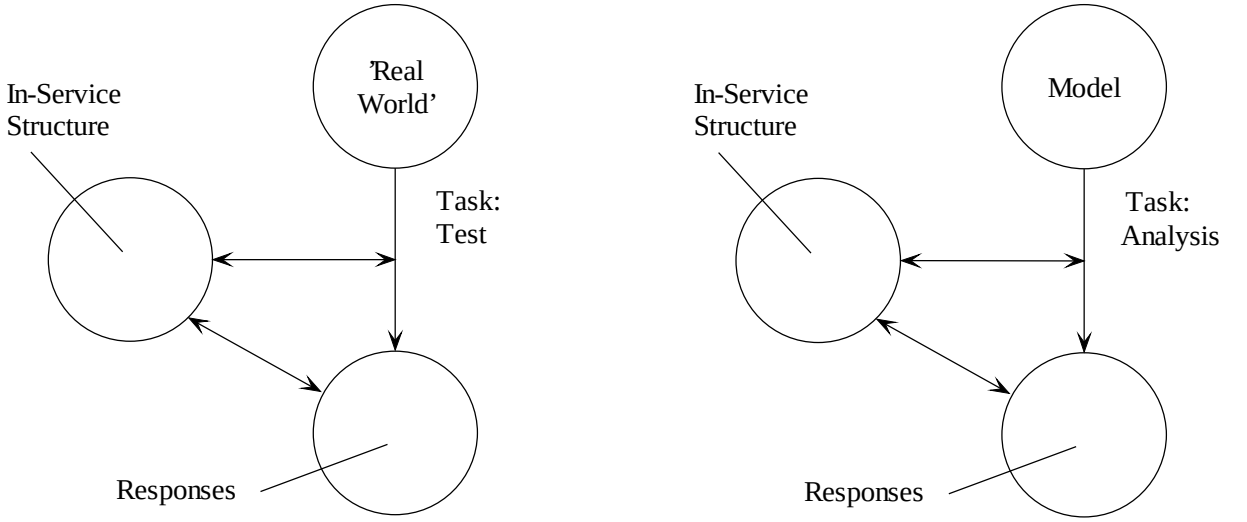

Fig.1

A common worry of those who encounter virtual testing for the first time is 'accuracy'. There often exists a belief that a laboratory test programme (including, in the aerospace industry, flight-testing) of a physically existent prototype or pre-production sample is inherently more reliable than analysis conducted on a virtual model that exists only within a computer memory. This belief is not always justifiable. *Fig.1* compares the two paradigms of test and analysis. The *In-Service Structure* is the entity that leaves the factory as a product for use by a customer. The *'Real World'* is the representation (usually a prototype) of the product that is to be tested. The *Model* is the computer-based representation of the in-service structure. A series of tasks are carried out on each representation and responses are generated.

A series of laboratory tests cannot be expected to cover all forms of behaviour that will be required of the in-service structure by the customer; only to ensure that safety-regulations are satisfied. In the case of physical testing it is often difficult to replicate the support conditions experienced by the real world structure and obtaining a realistic and comprehensive set of test loads always poses serious problems. Consequently there are a number of (almost always trivial) differences between the responses of in-service structure and representation.

While contemporary FEA software can generate responses to 32-bit detail, there is almost certain to be some level of idealisation between the in-service structure and the model. Consequently there are a number of differences between the responses of in-service structure and representation.

The conclusion is that a 'Real World' test sample is just as much a model as a computer-based virtual model. Neither can produce fully accurate responses, but both are capable of producing responses that are within the bounds of allowable error. These, and other factors, are often overlooked and the very fact that a physical structure is being tested is taken as a guarantee that the results obtained do model the real-world environment of the designed structure. In addition, if one then considers that a particular example of an object will fail in a different way from others within a given production run because of microscopic inconsistencies in its structure, such as cracks, then it is very difficult to say whether that one example of a production run is able to represent the batch 'better' than a model created from the design. Therefore, virtual testing should not be dismissed as a testing strategy on the grounds of accuracy alone.

In any form of testing the manner by which the data was obtained (the method) is often more vital to a reliable test than the data itself. Once it is accepted that the concept of virtual testing is not inherently less reliable than physical testing, the problem of ensuring that the method itself is reliable becomes paramount.

SAFESA

INTRODUCTION

In recent years there has been a trend towards ever-larger and more complex structures and engineering applications - so large in fact that physical testing techniques are no longer adequate. At the same time computer power has increased to the extent that these same large structures can be modelled without difficulty. Add to this the trend for the large structures to be increasingly in the safety-critical domain and there then exists a need for a formalised and reliable virtual testing procedure.

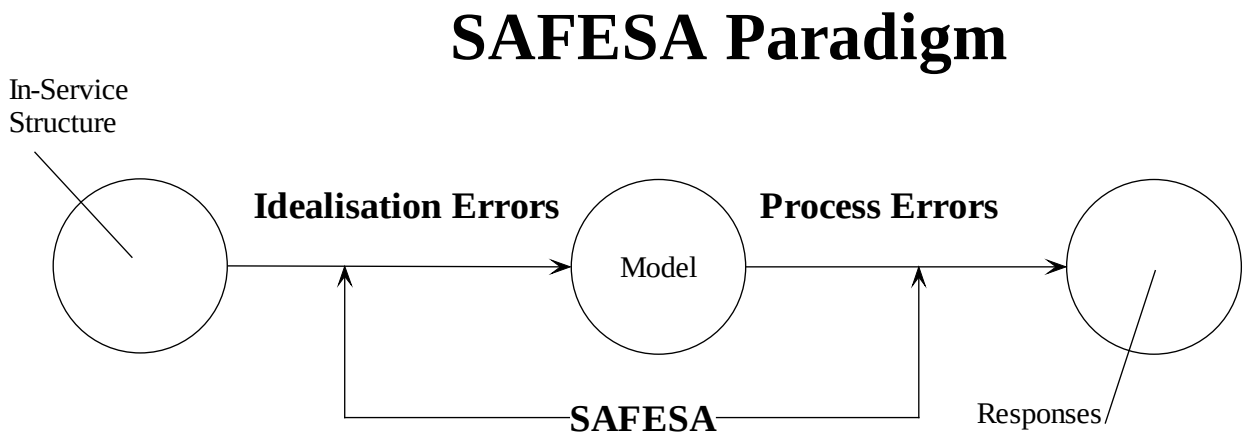

Fig.2

SAFESA is one of a number of projects sponsored by the UK Government's Department of Trade & Industry (DTI) as part of its Safety-Critical Systems Initiative. The aim of the project is to enable structural qualification to be carried out reliably and accurately using the FEA method

in safety-critical situations by a 'Best Practice'[1]. The philosophy of SAFESA is that of error-management; errors in the virtual testing process are identified, classified and treated. *Fig.2* shows the SAFESA paradigm in simple terms. Firstly, the in-service structure is defined in terms of the loading environment, the response environment, the certification or qualification requirements, etc. Secondly, an idealised model is generated from the in-service structure which can be used to generate a finite element model, whilst acknowledging that this idealisation is a possible source of error. Thirdly, the finite element model is used to produce a set of responses that can be used to qualify the in-service structure. More errors are generated at this stage. The SAFESA process is used to analyse the errors at all stages of virtual testing such that the in-service structure can be qualified with confidence.

SAFESA was developed with FEA in mind although the method is portable to other testing procedures. The drivers for the project included:

1. The trend to reduce physical testing by virtual testing.
2. The reduction of costs via reduced design cycle time and the promotion of concurrency.
3. The ability to provide full transparent auditing.
4. The improved legal position provided by the audit trail.

ERROR TREATMENT

FEA is a process which attempts to make certain generalisations or assumptions of the real world when constructing a model, sources of which have the effect of introducing errors into the analysis process. This does not invalidate any results obtained from a finite element analysis provided a proper error control system is used. The general procedure for error control is as follows:

1. Identification and classification of the error.
2. Quantification of the error.
3. Treatment.

There has been much work done on the process of error classification [2, 3] and a four-level taxonometric system has arisen. The four classes of error are:

1. *Modelling*, or *Idealisation*, errors, caused by a lack of knowledge of the real structure and its environment.
2. *Procedural* errors, due to discretisation meshing and post-processing.
3. *Formulation* errors, created during the conversion of a model to an actual finite element problem ready for solution.
4. *Solution* errors, produced during the solution of the Finite Element problem.

These classes can be further broken down. Each constitutes an error source, for which various error treatment techniques are available. The goal of error treatment is to progressively reduce the error estimate to less than a predefined threshold value, as the idealisation process is redefined.

The error treatment techniques are:

1. Rules based on experience,
2. Scoping calculations,
3. Comparison with existing test results,
4. Hierarchical modelling (model improvement),
5. Sensitivity analyses.

The current development phase - the construction of a SAFESA-based expert system - aims to automate both the identification of errors and possible treatment strategies.

Full details of the SAFESA process, a detailed breakdown of each stage, example problems and a more comprehensive discussion of the philosophy have been published in three reference works: the "*SAFESA Technical Manual*" [4], the "*SAFESA Quick Reference Guide*" [5] and the "*SAFESA Management Guidelines*" [6].

SAFESA EXPERT ADVISORY SYSTEM

The initial phase of SAFESA relied on the engineer, who was to perform an analysis by following the SAFESA methodology, to identify sources of error and to flag them for later treatment. Once SAFESA has been defined and published as a 'Best Practice' the aim of the project is now to implement SAFESA as a computer-based Expert Advisory System (EAS) that will advise the users of FEA software on the correct approach to take so that the final analysis is valid, with well-defined error-bounds. Such an analysis might be accepted as a good representation of the in-service structure.

In order to use a SAFESA EAS profitably in a design environment it must be implemented as a computer system that can integrate well with other engineering software. The requirements of SAFESA as a software package are that it:

- 'understand' the model or representation of the in-service structure using a feature/primitive paradigm,
- 'understand' the certification and qualification requirements for the domain of the relevant problem,
- 'understand' the requirements that the in-service structure must fulfil,
- have a rule-base allowing it to judge the actions of the analysis engineer and give advice as to what course of action should be undertaken,
- is able to communicate its opinions to the analysis engineer and present options for courses of action,
- is able to drive the analysis and post-processing software,
- provides an audit trail that details the current stage of the analysis, the options available to the analysis engineer, the recommendations given and the choices made,
- is flexible.

The software will have two uses during its experimental phase:

1. To demonstrate the benefits of a supervisory software system that can monitor the analysis process and alert the user if errors of judgement or procedural mistakes are being made.
2. By eliciting knowledge from one or more experienced engineers who themselves have no experience of SAFESA, but who can make a competent and independent analysis themselves, a useful cross-check can be made on the SAFESA paradigm, particularly some of the finer decisions.

It might be argued that an important part of many computer systems is the user. Two classes of user are here envisaged. Firstly, the experienced engineer who will use the system for reference purposes and to provide a clear audit trail. Secondly, the novice engineer who will learn about good practice in analysis by heeding the advice given; the advisory system can also be a tuition system in this respect.

IMPLEMENTING SAFESA WITHIN MSC/PATRAN

CHOICE OF MSC/PATRAN

The implementation of SAFESA employs knowledge elicitation techniques to create a knowledge-base of testing, analysis and error-treatment procedures. Analysis engineers perform a structural analysis and, through analysis of their actions, the software derives rules on how to perform future analysis in a similar manner and according to the appropriate criteria. The first stage of knowledge elicitation will involve an engineer performing an analysis while making reference to the SAFESA manual. This will transfer the SAFESA “knowledge” into a usable software form. At the same time, the software will generate a case-base of problems and solutions for future reference. Once the SAFESA “knowledge” has been elicited satisfactorily this knowledge-base will become the main reference system for advice on future analysis.

MSC/PATRAN has been chosen as the platform for SAFESA ESA software development because:

- it has well-integrated geometric design system,
- it supports most major analysis software, including MSC/NASTRAN and MSC/DYTRAN,
- it is extensible through a flexible programming language, MSC/PATRAN Command Language (PCL),
- it supports a database that is user-accessible and can be used to store client-defined data.

The provision of user-defined client-side database entities allows great flexibility in choice of structures for rule- and knowledge-bases. Most elicited data is stored in the form of *n*-dimensional arrays of real, integer, string or logical data. The arrays can be connected in networks that can mimic the structure of neural networks or connect via a system of pointers. System calls, function names and user-defined commands can be stored as freeform string data and then executed using *sys_eval()* or *ui_exec_function()*. This allows the user to add complex structures, from MSC/PATRAN built-in functions to entire PCL functions, to the database using a very simple referencing format. Such a process greatly aids the development of error-treatment processes that are the heart of SAFESA.

It was realised quite early on that a software implementation of SAFESA would require the use of artificial intelligence techniques. Finding a language that can support AI algorithms was easy; finding one that can support and drive analysis software was not so easy. It was recognised that PCL has certain helpful features, including the ability to:

- interpret freeform string data as system commands,
- store data (including knowledge-bases) in the MSC/PATRAN database,
- during run-time, automatically generate code, compile it and load it into memory,
- operate MSC/PATRAN while bypassing the user interface,
- support data structures that allow a feature/primitive syntax.

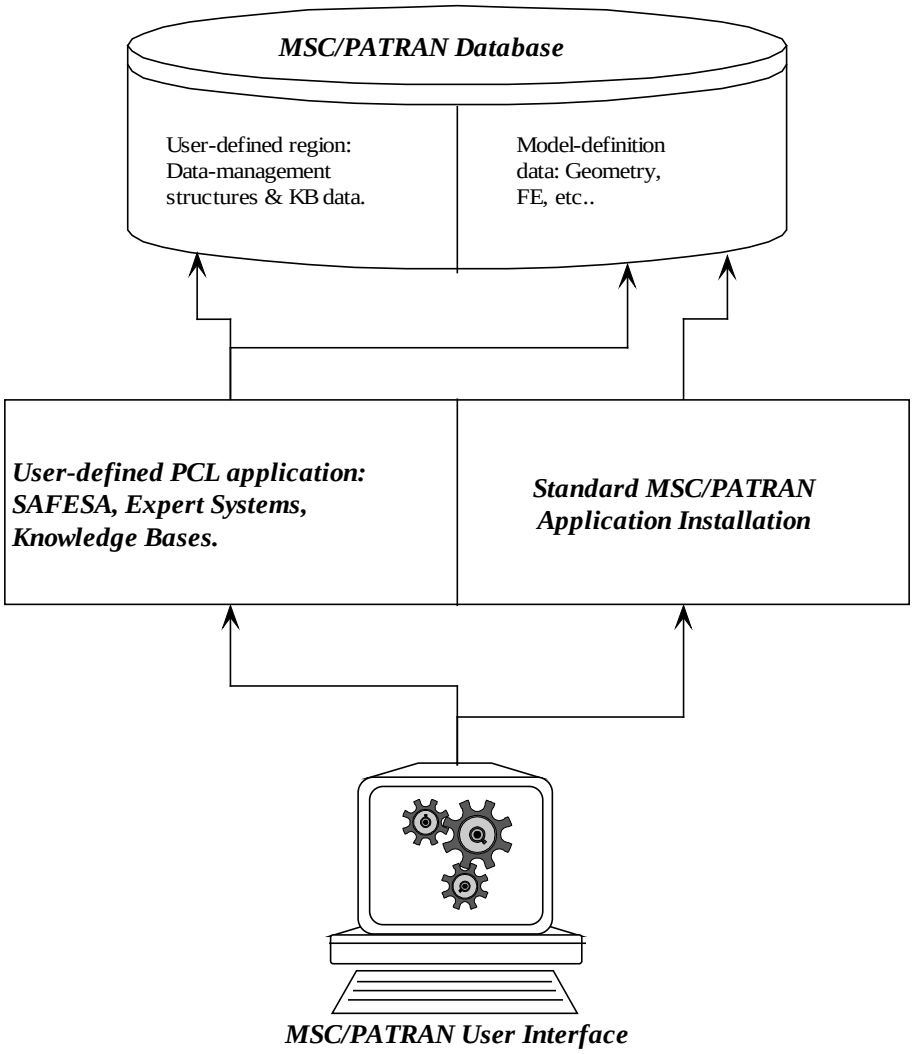

Fig.3

Whilst PCL is not the language of choice for developers of AI software (the two main criticisms are that it is slow and that it does not support object-oriented programming to the level that, say, C++ does), the above features allow many structures common to top-down AI to be used within

the software. Because of this, and noting that PCL, MSC/PATRAN and MSC/NASTRAN would integrate seamlessly without the need for encouragement by the developer, PCL was chosen as the development language for this project.

The SAFESA EAS interfaces with the user in the form of a series of extensions to MSC/PATRAN. Dialogue takes place between the user and the EAS via PCL forms hosted by the MSC/PATRAN user interface. Special SAFESA functionality is offered either from menus or, again, via a series of forms. *Fig.3* shows the basic layout of the system. There are systems that:

- elicit, store and propose the SAFESA rule-set,
- track the labels and MSC/PATRAN internal structure of geometry, finite elements, materials, element properties and other entities in a way which is more meaningful for the purposes of SAFESA,
- alter the geometry, nodes, materials, element properties and choice of elements in the model by negotiating directly with the MSC/PATRAN event manager, bypassing the user interface.

EXAMPLE APPLICATION

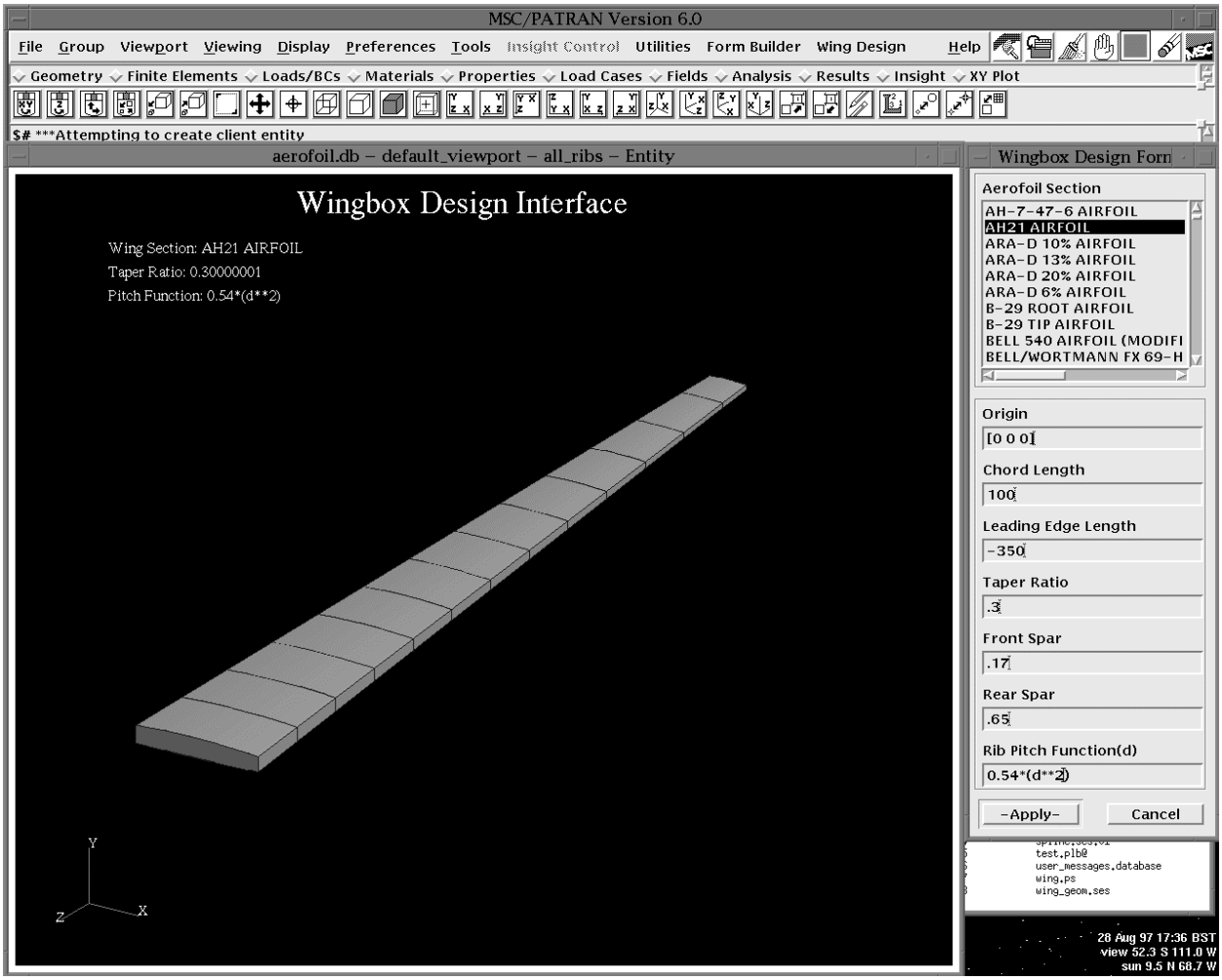

Fig.4

During development the SAFESA EAS will be tested on problems in the ‘large, transonic transport-aircraft wings’ domain. This domain was chosen because it was of a size small enough to allow a good understanding of the global function, yet complex enough to allow a number of procedural options. A support application (*Fig.4*) has been developed to generate simple wingbox models using the following criteria:

- Aerofoil section,
- Chord length,
- Leading-edge length,
- Taper ratio,
- Spar locations,
- Rib pitch function.

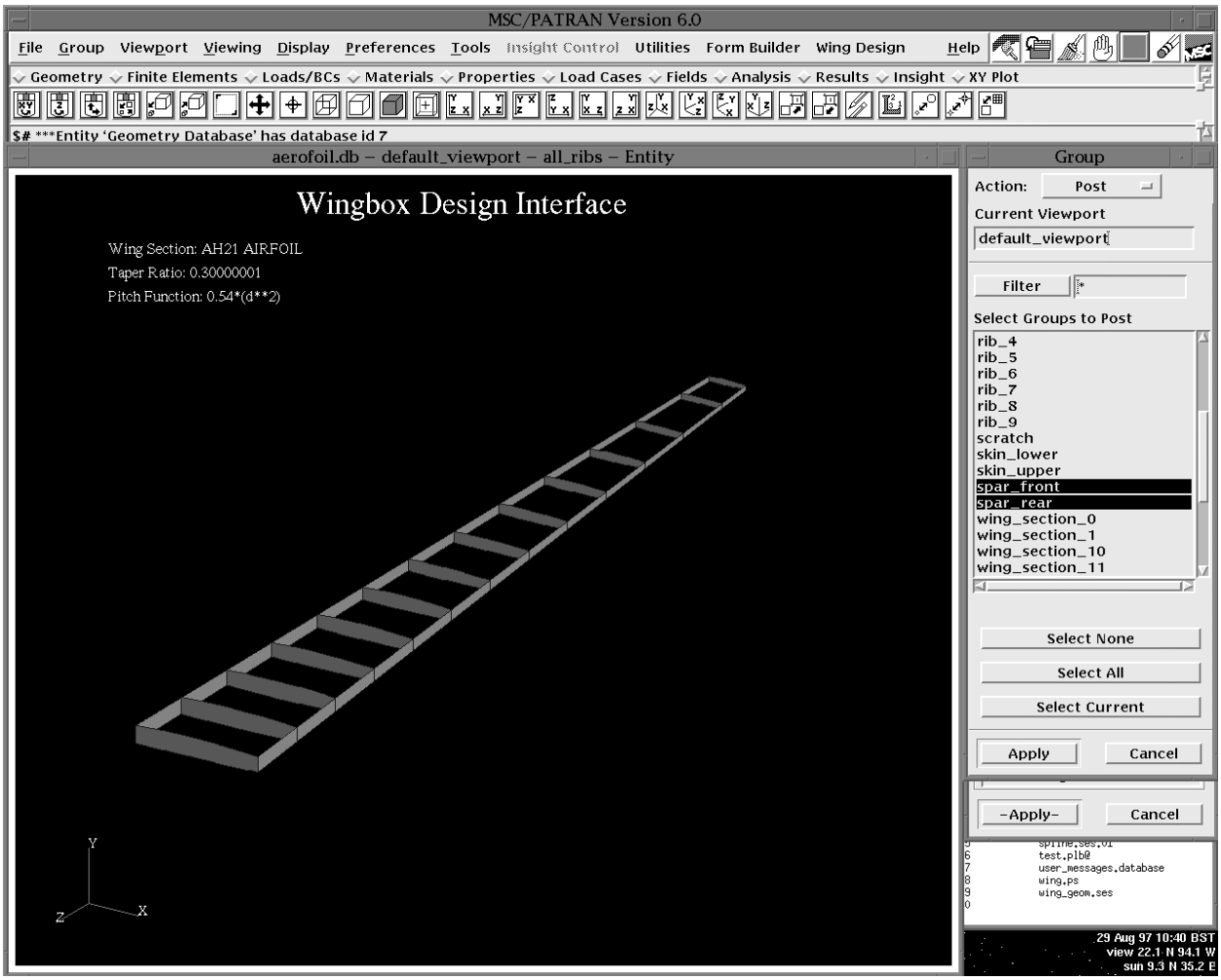

Fig.5

Once the idealisation is generated, the EAS determines the features of the structure. In the case of the simple wingbox the available features are skin sections, ribs, spars and stringers. *Fig.5* shows a list of features converted to groups to allow manipulation by the user via the ‘Group’ form.

The SAFESA process then performs a number of operations:

1. The features are checked to ensure that they all have material, loading and boundary qualities. Having determined the features, appropriate nodes are located in preparation for placement of finite elements. At this stage most of the activity is automated with very little input from the user.
2. A first pass of the model is performed, with finite elements being allocated to features. The choice of element is made by a SAFESA knowledge base which has prior (or built-in) experience of problems of this type. For example, a rib feature will require a different type of element than a skin panel feature. Additionally, if a medium level of accuracy is required, the feature may be modelled with just a single specialist element; if, however, greater detail is required then the single element can be replaced by a mesh of solid 3D elements. The element is selected from a sub-set of the total set of MSC/NASTRAN elements according to the following criteria:
    - Feature type (automatically determined),
    - Material type (from user),
    - Desired level of accuracy (from user),
    - Applied loads (from user),
    - Boundary conditions (automatically determined),
    - Existence of errors in the model (automatically determined).
3. The model is examined and known (to the knowledge-base) potential sources of error, such as particular joints or boundary conditions, are flagged up for investigation.
4. A first-pass analysis is performed and the response obtained.
5. The obtained response is compared with the allowable response.
6. If necessary, idealisation of the model is performed again, taking into account the error flags and endeavouring to treat them. Error-treatment strategies were discussed above. It is expected that there will be occasions when a schedule of laboratory tests are recommended to test a particular component or material property. Virtual testing recognises that such calibration with the real world is always necessary and seeks only to reduce *unnecessary* (and expensive) testing.

This iterative process of error-flagging, analysis and re-idealisation is performed until there is a satisfactory comparison of obtained and allowable response. The full process will often require several analysis sessions and some re-idealisation of the model. A flow-chart of the entire process is shown in *Fig.6*.

In theory, the software could be applied to any safety-critical structural domain, from aircraft to oil-rigs. In addition, some thought has been given to how the SAFESA paradigm can be applied to other areas of analysis, such as CFD and CEM.

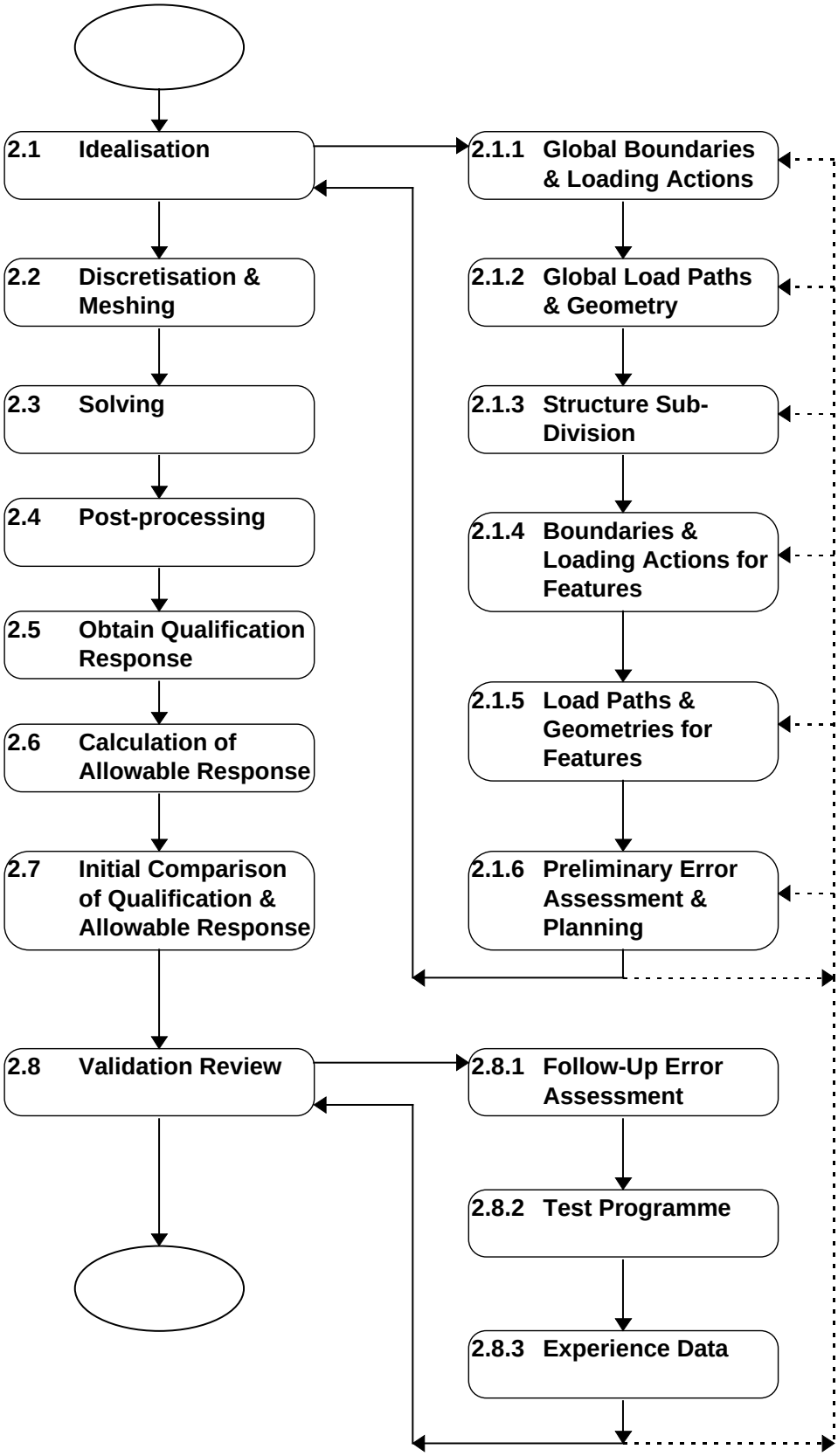
2.1 Idealisation
2.2 Discretisation & Meshing
2.3 Solving
2.4 Post-processing
2.5 Obtain Qualification Response
2.6 Calculation of Allowable Response
2.7 Initial Comparison of Qualification & Allowable Response
2.8 Validation Review
2.1.1 Global Boundaries & Loading Actions
2.1.2 Global Load Paths & Geometry
2.1.3 Structure Sub-Division
2.1.4 Boundaries & Loading Actions for Features
2.1.5 Load Paths & Geometries for Features
2.1.6 Preliminary Error Assessment & Planning
2.8.1 Follow-Up Error Assessment
2.8.2 Test Programme
2.8.3 Experience Data

Fig.6

FUTURE DEVELOPMENTS

The work carried out so far has been within the domain of the ‘large, transonic transport-aircraft wings’. The approach of this project can be extended in the following manner:

1. All problems within the aviation domain.
2. All structural problems (e.g., maritime, civil, mechanical).
3. Problems using other forms of computational physics (e.g., CFD, CEM). This paper is mostly concerned with FEA, although future work on this project will look at applying the techniques of SAFESA (see below) to CFD, CEM and other fields of virtual testing.

CONCLUSIONS

The SAFESA paradigm is a rigorous process for performing FEA as part of a virtual testing philosophy. It has a growing user base in the UK and is regularly applied to problems in civil engineering.

The implementation of SAFESA as a computer-based expert system will aid the development of engineering projects by reducing the cost of test programmes, defining the limits of reliability of the structure, producing an audit trail and training novice engineers in the process of coherent and reliable FEA.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the contributions made by the members of the SAFESA team; C. Kaucky-Laurence, A. Rahman, D.MacKay, N.Fox, N.Knowles, J.Maguire. The work was supported by the UK Department of Trade and Industry and the Engineering and Physical Sciences Council. The member organisations of the SAFESA consortium are Assessment Services Ltd., Cranfield University, Lloyds Register Ltd., Nuclear Electric PLC and W.S. Atkins Science & Technology Ltd.

REFERENCES

(1) Maguire, J.R., "A 'Best Practice' approach to finite element analysis", in "Proceedings of the 5th international conference on reliability of finite element methods for engineering applications", NAFEMS, 1995.

(2) Morris, A.J., Vignjevic, R., "Consistent finite element structural analysis and error control", Comput. Methods Appl. Mech. Engrg. 140, 87-108, 1997.

(3) Morris, A.J., "The qualification of safety critical structures by finite element analytical methods", College of Aeronautics, Cranfield University, UK

(4) The SAFESA Consortium, "SAFESA Technical Manual", NAFEMS, Ref: R0041, 1995.

(5) The SAFESA Consortium, "SAFESA Quick Reference Guide", NAFEMS, Ref: R0039, 1995.

(6) The SAFESA Consortium, "SAFESA Management Guidelines", NAFEMS, Ref: R0040, 1995.