Australian Government
Department of Defence
Defence Science and
Technology Organisation

Air Vehicles Division Computational Structural Analysis Facilities Policy and Guidelines for Users

S. Sanderson

Air Vehicles Division
Platforms Sciences Laboratory

DSTO-GD-0435

ABSTRACT

This document presents the operating policies and guidelines for the use of the Computational Structural Analysis Facilities (CSAF) within Air Vehicles Division (AVD). It has been created to assist users of the CSAF to perform professional, high quality finite element analysis (FEA). FE analysts from many tasks within AVD are using the facilities to conduct FEA with respect to the assessment of structural integrity and technical airworthiness of ADF aircraft. This document covers key issues such as data and resource management, as well as providing a comprehensive overview of the hardware and software facilities that are available for use. It also outlines recommended generic FEA modelling procedures and provides an overview of key FEA manuals and references. The specific FEA application software that is covered includes MSC.Nastran, MSC.Patran, ABAQUS, PAFEC and PIGS.

Published by

DSTO Platforms Sciences Laboratory
506 Lorimer St
Fishermans Bend, Victoria 3207 Australia

Telephone: (03) 9626 7000
Fax: (03) 9626 7999

AR-013-392
May 2005

APPROVED FOR PUBLIC RELEASE

Air Vehicles Division Computational Structural Analysis Facilities Policy and Guidelines for Users

Executive Summary

Air Vehicles Division (AVD) has built up a significant capability to perform finite element analysis (FEA). FEA is used extensively throughout the Division, particularly with respect to the assessment of structural integrity and technical airworthiness of ADF aircraft. The Computational Structural Analysis Facilities (CSAF) represent the considerable resources invested by the AVD on this FEA capability.

The main functions of the CSAF are to provide and manage the infrastructure required for advanced FEA within AVD. As part of the management of the facilities, it is the responsibility of the CSAF team to provide up-to-date reliable hardware and software to satisfy the increasing demands for and of FEA. The CSAF team also provides support to assist its users to perform professional, high quality FEA.

This document presents operating policies and guidelines for the use of the CSAF. It also covers key issues such as data and resource management, as well as providing a comprehensive overview of the hardware and software facilities that are available for use. It also outlines recommended generic FEA modelling procedures and provides an overview of key FEA manuals and references. The specific FEA application software that is covered includes MSC.Nastran, MSC.Patran, ABAQUS, PAFEC and PIGS.

This document will help to provide AVD with the benefits of more efficient utilisation of the CSAF resources by all users. It will also help to enable a higher quality of FEA to be undertaken by users, as well as enabling task managers to be more informed about the CSAF and therefore better able to utilise this important set of resources.

Authors

S. Sanderson
Air Vehicles Division

Mr. Sanderson commenced work at the Aeronautical Research Laboratory in 1981. He has developed flight data reduction and analysis software for Mirage, F-111 and F/A-18 projects. Several of these programs have been implemented by NAE for part of their data reduction in the IFOSTP project. Since 1992, Mr. Sanderson has undertaken finite element analysis of composite and bonded structures for the F-111 and F/A-18 aircraft and provided training in finite element modelling and analysis. In 1998 Mr. Sanderson was appointed as the Systems Administrator of the Air Vehicles Division Computational Structural Analysis Facilities within Platforms Sciences Laboratory.

Contents

1. Introduction

Air Vehicles Division (AVD) has built up a significant capability to perform finite element analysis (FEA). FEA is used extensively throughout the Division, particularly with respect to the assessment of structural integrity and technical airworthiness of Australian Defence Force aircraft. The Computational Structural Analysis Facilities (CSAF) represent the considerable resources invested by AVD on this FEA capability.

The main functions of the CSAF are to provide and manage the infrastructure required for advanced FEA within AVD. As part of the management of the facilities, it is the responsibility of the CSAF team to provide up-to-date reliable hardware and software to satisfy the increasing demands for and of FEA. The CSAF team also provides support to assist users of the CSAF to perform professional, high quality FEA.

This document presents operating policies and guidelines for the use of the CSAF. It is also an instruction manual and comprehensive reference guide, intended mainly for new users of the CSAF. However, more experienced users should also find this document a useful quick reference to information that they are less familiar with. It is also recommended that supervisors of the users of the CSAF be familiar with the usage policies.

The aim of this document is to also provide an overview of the computing hardware and software that comprises the AVD CSAF without going into in-depth technical details. The key areas covered in this document are: (i) data and resource management, (ii) an overview of the hardware and software available for FEA, (iii) outlines of generic FEA modelling procedures, and (iv) an overview of key FEA manuals and references. The specific FEA application software that is supported includes MSC.Nastran, MSC.Patran, ABAQUS, PAFEC and PIGS, and is run on a few high-performance Unix servers and workstations. From time to time other FEA codes may also be trialled.

By making a selection of advanced FEA codes available, together with maintaining a modern high-performance computational facility, the CSAF provides AVD with the capability to conduct large-scale FEA for through-life support of Australian Defence Force aircraft (e.g. F-111, F/A-18, P-3C) and helicopters (e.g. Blackhawk). This document will provide AVD with the benefits of more efficient utilisation of the CSAF resources by all users, as well as enabling better quality FEA to be undertaken. It will also enable task managers to be more informed about the CSAF and therefore place them in a much better position to utilise the full potential of this important resource.

2. Overview of CSA Facilities and Usage

The AVD CSAF forms the backbone of many AVD tasks undertaking FEA, and it is therefore imperative that AVD has and maintains reliable, modern and capable facilities for CSA. These facilities presently include a Hewlett-Packard (HP) 9000 Series 700 Model J7000 4-processor FEA Server system, a Silicon Graphics Inc (SGI) Origin 2000 16-processor Numerically Intensive Computer (NIC) system, an HP 9000 Series 800 Model K260 2-processor system, an HP C240 graphics workstation, peripherals, commercial finite element analysis software packages (e.g. MSC.Nastran, MSC.Patran, ABAQUS and PAFEC), scientific plotting software (Tecplot), Fortran and C/C++ compilers, text editors, debuggers, backup and system management software, and other assorted workstations and X-stations that facilitate user access. A generalised representation of the major hardware features and the network topology of the CSAF is shown in Figure 1. The NIC (administrated by SCIS and located in the Fishermans Bend Computer Centre) performs the long computationally intensive FEA processing (using software purchased by AVD). The HP FEA Server performs jobs that pose a medium computational load, as well as almost all pre- and post-processing of FEA models and results. The HP FEA Server is capable of handling very big and complex FE models that require large memory and disk capacities. User access to these computers is through workstations, X-terminals, or personal computers using X-terminal emulation and telnet software. Most of the CSA servers, workstations, terminals, manuals and references are housed in the CSAF Workstation Room, which is located in Bldg 3.

The core function of the AVD CSAF is to provide and manage the main advanced FEA infrastructure within AVD. The CSAF computers are linked through the DSTO/Defence restricted network. See section 3.1 for more details regarding the conditions of access and use of the facilities. Most of the CSAF software licenses (see Table 1) are node-locked to run only on their specific server. The MSC.Patran licenses served by jayted are an exception, and they are available over the network for use on other machines.

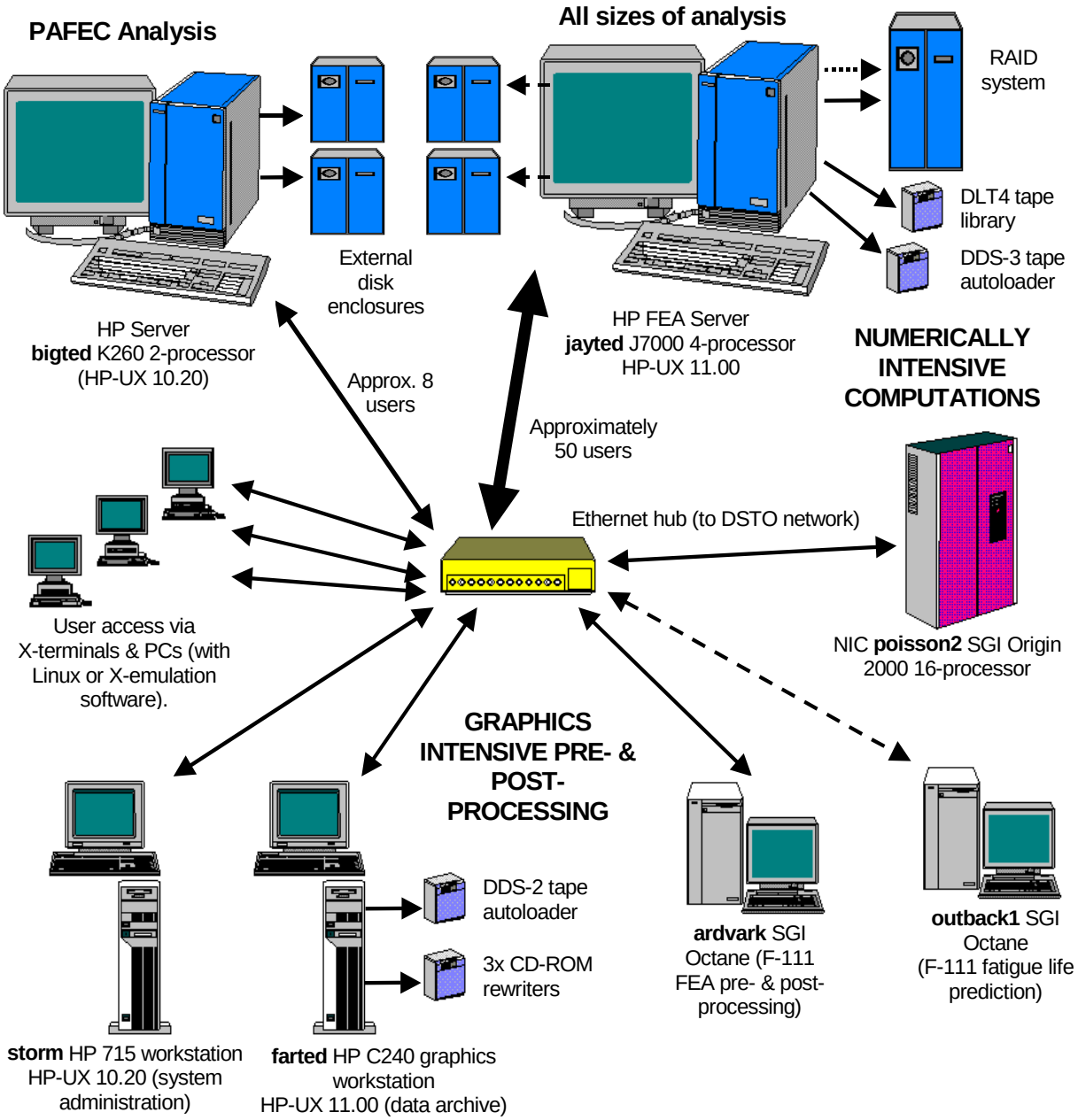

Figure 1 The AVD CSAF on the DSTO network

Table 1. Software licenses available on each server and workstation - as at Feb 2004

FEA software	Licences on server/workstation				
	jayted	poisson2	bigted	farted	ardvark
MSC.Patran	10*			*	*
ABAQUS	2	1			
MSC.Nastran	2	2			
PAFEC	2		2		
PIGS	1		2		
TMP Slim/Vision					1

Other software	Licences on server/workstation				
	jayted	poisson2	bigted	farted	ardvark
Tecplot	2				
Fortran 77	1	1	1	1	1
Fortran 90	1		1	1	
Ansi C/C++	1				

* Note that jayted is the server for the MSC.Patran licenses.

2.1 CSAF team responsibilities

The CSAF team is responsible for providing, maintaining and managing the CSA Facilities. This includes keeping the hardware and software up to date to cope with the ever-increasing demands of FEA, as well as providing adequate backup of data. The team places a high priority on minimising the extent of any interruptions to services that could disrupt the achievement of milestones in other AVD tasks.

The CSAF systems have been configured (as far as possible) to minimise machine downtimes in the event of hardware failures. Data backup schemes are in place to ensure that little or no user data is lost if any of the computer systems have problems. In particular, the potential loss of data from large-scale disk failures is protected against by the combination of a policy of performing daily incremental backups and employing hardware RAID devices for disk storage. Disaster recovery procedures have also been established to try to avoid any unacceptably long computer system downtimes while the hardware is repaired/replaced and file storage systems are being rebuilt.

3. Policy of Use

3.1 Conditions of access to the CSAF

It is a condition of access to the site Network (i.e. the DSTO-Restricted Network) that all users must have both read and understood the DSTO POLICY document DSTO-SEC-003 on Information System Security Practice & Procedures (IS-SPP) For the DSTO RESTRICTED Network. Those users unfamiliar with the contents of the DSTO IS-SPP for the DSTO-R Network must ensure that they read and understand this document as a matter of high priority. The URL link to this document is given below:

http://web-vic.dsto.defence.gov.au/workareas/SCIS/networks/dsto-sec-003.pdf

*It is important to note that only **unclassified** data is to be stored on the CSAF.*

3.2 CSAF policy on data management

It is the general policy of the CSAF that the majority of work, particularly the larger analyses, be carried out on temporary work areas, such as /tmpwrk/<user name> on jayted (where <username> represents the user's login name). It is up to each user to copy any essential data to their home area, such as /home/<user name> on any of the HP machines. It is also the user's responsibility to periodically organise the data from specific projects and request that it be archived onto CD-ROMs. As a result of the large amount of data that is usually transient in nature, users must be aware that **no** data will be backed up from any temporary work areas. Only data on the user's home area will be regularly backed up by the CSAF.

3.3 Responsibilities of users

3.3.1 General data and resource management

It is the individual responsibility of each and every user to:

a) Maintain records of the analyses that they perform.
b) Ensure that any important data is made available to be backed up and archived in an appropriate manner.
c) Monitor their usage of resources. This includes both disk space and software licences. The user is responsible for the clean up of any unwanted files and, where possible, reducing the volume of output requested, particularly for larger analyses. On jayted, this also includes running only one MSC.Patran session at a time and closing it as soon as possible; running a maximum of one ABAQUS, one MSC.Nastran and one PAFEC analysis at a time; and restricting analyses to run

times of less than two hours in duration. On poisson2, ABAQUS and MSC.Nastran analyses (with a limit of one MSC.Nastran job at a time) should be submitted through the NQE queue manager. Also, on bigted, only one PAFEC analysis at a time per user is permitted.

d) Use the CSAF systems and software in a responsible manner, with consideration for other users.

3.3.2 Backup and archiving of data

The user is required to arrange for the archiving of important data:

a) Throughout the life of a project.
b) To ensure that all relevant data from a completed project is archived (onto CD-ROM), with copies distributed to Task Managers and CSAF archives.
c) It is also highly desirable that the data be removed from the CSAF system after the project has been completed and the data has been archived.

3.4 Supervision of new/novice users

Task Managers must ensure that any new users are supervised by more experienced FE analysts and that all users are fully aware of the policies governing the use of the CSAF. This includes having read this document and being briefed on any changes in the current policy and user guidelines that are above and beyond those provided in this document.

To assist all users, but in particular new users, the CSAF maintains a catalogue of recent AVD reports on work involving finite element analysis. The catalogue also contains lists of many useful references. The catalogue is presently available on request from the author (S. Sanderson) and is to be eventually made available at http://jayted/CSAF/reports/catalogue.html. The aim of the catalogue is to provide references to past and present work that was conducted using the CSAF, so that any user may obtain assistance from the reports and advice. It may also be possible for users to obtain examples from others who may have recently done similar work using the same or different FEA codes.

4. CSAF Hardware

The following sections provide an overview of the computing hardware that is utilised by the CSAF. The first section (4.1) covers hardware that is directly managed by the CSAF Team. The last two sections (4.2, 4.3) cover hardware managed and operated by contractors or other sections of DSTO, for which the CSAF provides guidance and assistance in the management and day-to-day operations.

4.1 Hardware managed by AVD

This section details the computer hardware of the CSAF and also lists the software available on each server and workstation platform. The specific FEA software that is available on each platform is described in detail in sections 6 and 7, and examples of typical operations to perform FEA on the CSAF are given in section 8.

4.1.1 Main AVD FEA server

The main CSAF FEA server is an HP 9000 Series 700 C-class Model J7000 workstation called jayted, which is shown in Figure 2. It is located in the CSAF Workstation Room in Bldg 3. Its hardware specifications are:

- 4x 440 MHz PA-Risc 8500 CPUs
- 8 GB RAM
- ≈200 GB disk space
- HP-UX 11.0 Operating System

The main functions of jayted are to perform the bulk of the FEA computations, which consists of all sizes of PAFEC jobs and small- to medium-sized jobs using MSC.Nastran and ABAQUS, and the bulk of FEA pre/post processing using MSC.Patran and Tecplot. It also has compilers for the ANSI C/C++ and Fortran 90 languages to allow users to develop their own applications for pre/post processing of FEA data. It also acts as the main backup server for all the CSAF computer systems.

Figure 2 CSAF main FEA server jayted. The system enclosure with console on top is on the left. The scratch disk tower case, CD-ROM tower case and DDS3 tape unit are in the middle. The Viper2 RAID system enclosure containing the home areas is on the right, with the temporary work area disks and a DLT4 tape unit on top.

4.1.2 Secondary AVD server

The secondary CSAF server is an HP 9000 Series 800 K-class Model K260 server called bigted and is shown in Figure 3. It is located in the CSAF Workstation Room in Bldg 3. Its hardware specifications are:

- 2x 180 MHz PA-Risc 8200 CPUs
- 1 GB RAM
- ≈60 GB disk space
- HP-UX 10.20 operating system

The primary function of bigted is to perform PAFEC FEA. It has several levels of the PAFEC suite of FEA applications installed. Note that the current level, 8.6, of PAFEC that is installed on bigted is limited to running under HP-UX 10.20. bigted has no other FEA codes but it does have the PUPPIES pre/post processing software installed, as well as Fortran 77 and Fortran 90 compilers.

Figure 3 CSAF secondary FEA server bigted, showing system enclosure with console on top, and disks and CD-ROM units on the table at left, with bigted's UPS under it. Note that the UPS at right is supporting jayted.

4.1.3 Main AVD graphics workstation

The main CSAF graphics workstation is shown in Figure 4 and is an HP 9000 series 700 C-class model C240 workstation called farted. It is located in the CSAF Workstation Room in Bldg 3. Its hardware specifications are:

- 1x 240 MHz PA-Risc 8500 CPU
- 1.25 GB RAM
- ≈30 GB disk space
- HP-UX 11.0 operating system

The two main functions of farted (see Table 2) are to perform data archiving onto CD-ROM and for the more graphics intensive pre/post processing in MSC.Patran of the larger and more complex MSC.Nastran and ABAQUS models. It does not have any FEA code installed.

Figure 4 CSAF main graphics workstation, farted, showing the user console, with CD writers and DDS tape units sitting on top of the system enclosure.

4.1.4 Other workstations

The other CSAF workstation (see Table 2) is an HP 9000 apollo Series 700 Model 715 workstation called storm. It is located in the System Administrator's office in Bldg 3. Its hardware specifications are:

- 1x 33 MHz PA-Risc 8000 CPU
- 96 MB RAM
- ≈9 GB disk space
- HP-UX 10.20 operating system

The primary function of storm is remote system administration of the CSAF. It has no FEA software or Fortran compiler.

Table 2. Summary of workstation models and names

Workstation make/model	Qty	Registered IP name	Function	Ref
HP C240	1	farted	CD data archiving; FEA pre- and post-processing	4.1.3
HP apollo 715	1	storm	System administration	4.1.4
SGI Octane 2	2	aardvark outback1	FEA pre- and post-processing Fatigue life calculation	4.2.1

4.1.5 X-terminals

The CSAF also maintains a number of X-terminals (see Table 3). Most are located in the CSAF Workstation Room in Bldg 3, while the rest are in users' offices around the Fishermans Bend site. Several of the X-terminals in the CSAF Workstation Room are shown in Figure 5.

Table 3. X-terminal models and names

Make/model	Qty	Registered IP name/s
HP Envizex II	1	orac
HP Envizex – model P	2	heyted, chunted
HP Envizex – model A	3	ricted, bonnie, cowted
HP 700/RX	1	exted
Tektronix x500	1	freya
Labtam – model C320	6	abrizon, tacoma, ifostpx2, fattam, adour, bugs

Figure 5 X-terminals in the CSAF Workstation Room, showing from left to right, cowted (HP Envizex A), heyted (HP Envizex P) and abrizon (Labtam CT320).

4.1.6 PCs with X-emulation software

Many users now access the CSAF from their corporate desktop PCs via X-emulation software such as X-Win32, Excursion or Exceed. The user is responsible for maintaining the X-emulation software on their own PC; the CSAF Team can only provide limited assistance to users with troubleshooting of X-emulation software problems. It is recommended that large monitors be used with MSC.Patran, i.e. minimum of 17" monitor running at a resolution of 1280x1024 pixels. It is also advantageous to communicate to the X-server (on jayted) via a high-speed (100 Mb/s) network connection, particularly when manipulating more complex models within MSC.Patran.

4.2 Hardware managed by F-111 SOP

4.2.1 SGI Octane workstations

The two SGI Octane workstations (see Table 2) are managed by AeroStructures contract engineers working on the F-111 Sole Operator Program (SOP). The workstations are SGI Octane 2 workstations called ardvark (shown in Figure 6) and outback1. They are located in Bldg 3, Rooms 114 and 104, respectively. Their hardware specifications are:

- 1x 200 MHz R10000 CPU each

- 1.25 GB and 512 MB RAM, respectively
- ≈40 GB and ≈20 GB disk space, respectively
- IRIX 6.5 operating system

The primary function of ardvark is the pre/post processing of the large F-111 models using MSC.Patran and TMP Vision/Slim software. It has no FEA codes installed. The primary function of outback1 is to perform computations involving the use of the METLIFE fatigue life management program.

Figure 6 SGI Octane workstation ardvark, with DDS4 tape drive and CD-ROM at far left, shown running TMP Slim software.

4.3 Hardware managed by SCIS

4.3.1 NIC server

The NIC server is an SGI Origin 2000 server called poisson2. It is located in the SCIS server room in Bldg 23. Its hardware specifications are:

- 16x 200 MHz R10000 CPUs
- 4 GB RAM
- ≈160 GB disk space

- IRIX 6.5 operating system

The primary function of poisson2 is to perform large and lengthy numerically intensive computations. It has the MSC.Nastran and ABAQUS codes installed. It is used to run large FEA jobs through a batch queuing system. The policy of use of the NIC, as determined by the NIC Management Committee, is that no post processing of analyses be carried out on this machine. Therefore, the X-display manager, XDM, has been disabled on this machine and MSC.Patran has not been installed.

5. CSAF Software – Unix Systems

This section gives a brief general description of Unix systems and some more specific details about systems associated with the CSAF. The general structure of Unix and some of the similarities and differences of the systems are illustrated.

5.1 Unix systems, shells and environments

The basic program that interacts with users on a Unix system is called a shell. There are several shells used by Unix, common ones being the C-shell (csh), Bourne shell (sh), Korn shell (ksh), Born Again shell (bash) and the Posix shell (sh).

All of the HP systems in the CSAF are set up to use the Posix shell (sh) with the CDE desktop environment. The SGI machines use a Korn shell or an SGI variation of the C-shell, either csh or tcsh, with the SGI Desktop environment (except for the NIC which has had its Desktop environment deliberately disabled). While the Unix environments on each server have many similarities and may look the same, they are actually all different. In particular, the CDE environments on the various CSAF HP machines look almost identical to one another, therefore the user must maintain an awareness of which server or workstation they have logged into and to which machine any particular terminal emulator window on their desktop is connected.

For example: a user may log in to *jayted* through the X-display manager, XDM, on an X-terminal called *heyted*. Once CDE has started the user opens two xterm windows. Both of these windows have their DISPLAY environment variable set to *heyted*. In the first window the user runs MSC.Patran to prepare a data deck for analysis on *poisson2*. In the second window the user does a remote login to *poisson2*. Now in this window the user copies the data deck from *jayted* and then submits the job – on *poisson2*. After the job completes the user tries to run MSC.Patran to review the results. The response is a command not found error – because the current window is connected to *poisson2*, which does not have MSC.Patran. The user must copy the results to *jayted* then switch back to the first window before running MSC.Patran.

5.1.1 The Unix file system

A lot of what you want to do when using Unix involves the manipulation of files. In Unix, groups of files are collected together in directories. The directories are arranged in an upside down tree. The very top of the tree is the root directory. Directories that are contained in other directories are considered to be lower down in the tree. When you log in you find yourself in what is known as your home directory. It has the same name as your login name. For example, if you type zmythe as your user name to login, then your home directory is named zmythe. Your home directory is actually located partway down

the tree, e.g. /usr/people/zmythe on the SGI machines and /home/zmythe on most other systems. The list of directories from a file to the top of the tree is called its full path. An example of a full path is /usr/local/bin.

In Unix the "slash" / is used in a path to separate a directory from the one above it in the tree. The directory that you are working in is called the working directory, the present working directory or the current directory.

5.1.2 Posix shell

The Posix shell (sh) is the default shell used on the CSAF HP machines. The Posix shell is a superset of the Korn (ksh) and Bourne (sh) shells. On the CSAF HP systems the shell prompt has been enhanced to display the machine name, user name and current directory (as illustrated in 5.1.4). This enhancement is done in an environment file, .shrc, which is itself defined within and called from the .profile file for each user. Several common "global" aliases are also defined within the .shrc file. The user may customise this file.

5.1.3 Startup files: .profile, .shrc, ...

There are a number of files that control how various aspects of Unix will work for you. You can put commands into these files and thereby "customise" your environment. The filenames of these files often start with a dot, and often they are collectively referred to as your environment files. They may also be given various other names, like "dot rc" files, startup files, etc. Files that begin with a dot are not listed with the standard **ls** command, so they are sometimes also called "hidden" files. You can get a listing of all these files using **ls –a** or **ll –a**.

For the Bourne, Korn and Posix shells, the .profile file is read when you log in. It controls the messages you get initially, sets the terminal type, etc. (The .login file does the equivalent for C-shells.) As the Posix shell is the default for the CSAF, most of the things you will probably want to do involve putting things into your .shrc file. Commands in the .shrc file are executed every time you start a Posix shell. The .profile is only read when a login shell is started. The system automatically launches a login shell when you initiate a new connection. At any other time you may also cause a new terminal emulator to start up as a login shell. If you want to have various aliases available whenever you are logged in, you may add to the existing list of alias commands in your .shrc file.

5.1.4 What the command line prompt can tell you

Most accounts are set up so that the command line prompt gives the user some information about where they are in the file system and what machine they are connected to. For example, the prompt

[auser@ted SomeDir]$

tells the user that they are logged in as auser on a machine called ted and that the current directory is called SomeDir.

5.1.5 CDE – the Common Desktop Environment

On the CSAF HP systems, the standard login method uses the X-Display Manager, XDM, to start the Common Desktop Environment, CDE. The CSAF CDE has also been customised. The CDE front panel has been extended with the addition of a "CSAF apps" subpanel. Actions (also called dt-actions) for the following applications have been created within this subpanel: Acrobat Reader (**acroread**), **netscape**, **xemacs**, MSC.Nastran Encyclopedia (**iview**), **ghostview**, ghostscript (**gs**) and **tecplot**. Also, in line with the Defence policy for use of the restricted network, all screen savers have been enabled with locking set. The preferred method of access to the CSAF is through the CDE interface. CDE allows the user to perform many "commands" by clicking on icons and actions with the mouse.

5.1.6 On-line help – there is no help command

There is no help command, however there is on-line documentation available for most Unix commands and programs. These documents are called manual pages and are often referred to as man pages. On the front panel of the CDE there is a Help subpanel. Through this subpanel various actions such as the Help Manager and Man Page Viewer may be used to access information. In general, these actions invoke some Motif-based wrapper to aid the user to access system information by putting the information inside a window with a scrollbar and sometimes allowing the user to search for various expressions. For a description of how to use the command line version of the **man** command see section A.1.

5.2 Syntax of commands and parameters

The syntax used within this text to describe the various commands and their parameters is:

command [**opt_name**] [**opt**={ alt1 | alt2 | alt3 … }] (**req_name**) *user_input*

where:

command	is the command name
[ ]	denotes a list of optional parameters or keywords
( )	denotes a list of required parameters or keywords

{ … | … }	denotes a set of mutually exclusive parameters or keywords
req_name	is a required keyword parameter
opt_name, opt	are optional keyword parameters
alt1, alt2,…	are alternate selectable keyword values
default	is underlined to indicate that it is a default value
user_input	is a user supplied input parameter or file name

5.3 Unix commands

The following sections define what is meant in this text by basic and advanced Unix commands. The distinction is fairly subjective – some users may need to use only a few of the very basic Unix commands, while others may find that their needs soon require the regular use of some of the more advanced commands. Descriptions of Unix and explanations and illustrations of a few selected commands are given in Section 5.4. Reasonably comprehensive quick reference lists of basic and advanced Unix commands, together with their key one-line descriptions are provided in Appendix A (in A.2 and A.3 in particular). The command lists are in alphabetical order.

5.3.1 Unix commands – basic

The basic commands are what a user could expect to use on a regular basis while performing FEA on the CSAF. Refer to the one-line descriptions in Appendix A.2.

5.3.2 Unix commands – advanced

The advanced commands are those that a user could expect to use less frequently, but should nevertheless be aware of. These commands, in general, allow the user to perform more sophisticated and complex manipulation of data. Refer to the one-line descriptions in Appendix A.3.

5.4 Unix commands explained

Underneath or behind CDE, which is the HP graphical user interface (GUI), UNIX is still used by typing commands. These are typed after the prompt on what is called the command line. This command line is actually the interface with the (Posix) shell attached to that terminal. If you are connecting to the server using a serial or "dumb" terminal, or are not using a window system, you just type the commands that you wish to use. If you are using any X-window system, e.g. CDE, you type Unix commands in a terminal-like window.

There are basically two types of Unix commands. In one type of command you ask the computer for information. For example, if you type

ls

the computer will respond by printing a list of your files on the screen. In the other type of command, you ask the computer to do something. It does not necessarily need to print any response. These types of commands usually are asking for something to be done to a file. The basic form of these commands is: *command filename*. These tell the machine to perform the action dictated by *command* on the file that is called *filename*. An example is:

rm *model.dat*

which deletes the file named model.dat from the current directory. This will permanently remove the file from the system. Be warned: the shared file systems under Unix mean that it is almost impossible to recover deleted files.

Many Unix commands have options. These are preceded on the command line by a minus sign –. The basic form is: *command –option filename*, or just *command –option*. Two examples are:

rm –i *model.dat*
ls –l

5.4.1 Paths and patterns – ways to refer to files: *, ~ , .., .

The asterisk or wildcard * is very useful in referring to files. It matches any string of text. For example, *dat refers to any files that end in dat. All the files in the current directory can be referred to by *. Thus **cp** **.dat* Models will copy all the files in the present working directory with the .dat extension into the subdirectory called Models.

Beware, doing a command like **rm** *s*.dat* is good for removing a list of files called stuff1.dat, stuff2.dat, ... but it will also remove the file special.dat that you forgot was also there. To be safe, you can always do **ls** *s*.dat* to check what your wildcard is matching before you take action.

One dot . and two dots .. refer to the current directory and the directory one level up, respectively. Thus, cd .. moves you one level up the directory tree. The tilde ~ is used to refer to home directories. For example, **cd** ~ moves you to your home directory and **cd** ~*zmythe* moves you to the home directory of the user zmythe.

When you give a command that refers to a file or a directory, you can always give its full path from your home directory. For directories and files that are contained in your current directory you can just give the name. For example, if you are in a directory called Models

that is contained in your home directory, and you want to move to the directory Reports that is also in your home directory, you can do **cd** *../Reports* or **cd** *~/Reports*.

5.4.2 Listing the contents of a directory: ls

The command **ls** gives a list of the names of the files in a directory. It can also be used to give the names of certain types of files. It has many options. Here are some examples.

ls	List all files in the current directory
ls **.dat*	List all files that end in .dat
ls –a	List all files including hidden ones (files whose names begin with a dot .)
ls –s	Give the sizes of files in kilobytes
ll or **ls –l**	Give the long form with permissions and modification date

5.4.3 Moving, copying and removing files: mv, cp, rm

The **mv** command moves (or renames) files. The simplest form of use is **mv** *oldfilename newfilename*. It will also move directories. If you use it in the form **mv** *filename directoryname* it will move the file into the named directory keeping its old name. To change its name you do **mv** *oldfilename directoryname/newfilename*.

Copying files is similar to moving them but you use **cp** instead of **mv**.

Beware, if you move or copy a file and there already exists a file with your *newfilename* in the target directory, that file will be destroyed.

To remove a file you **rm** *filename*. On Unix systems with shared file systems, once you remove anything it is gone. If you don't like this you can create a safer alias as instructed below. There are also regular backups made of files on the /home area, but an ounce of prevention is worth a pound of cure.

5.4.4 Changing, creating and removing directories: cd, mkdir, rmdir

You can make a new directory using **mkdir** *newdirectoryname*. You can remove a directory using **rmdir** *directoryname*. To remove a directory, you must first remove all the files it contains. To change directories to a directory that is contained in the current directory use **cd** *directoryname*. To move up one directory **cd** .. does the trick. To move to a directory on the same level, i.e. both the current directory and the desired one are both contained in the same directory, use **cd** *../directoryname*.

5.4.5 Looking at the contents of a file: more, head, tail, cat

The command **more** *filename* will display the named file page by page. To view the next page you hit the space bar. To see just the first few lines of a file use **head** *filename*. The last part of the file can be viewed via **tail** *filename*. The command **cat** *filename* will cause the entire file to be printed on the screen. This means that if the file is longer than a single screen full it will just scroll rapidly by. You can, of course, also look at a file in an editor like **dtpad**, **xemacs** or **vi**.

5.4.6 Printing files: lp, lpstat, cancel

To print a text file you do **lp** *filename*. To check the files waiting to be printed you type **lpstat**. This may also tell you if something is wrong with the printer. It may take up to a minute or so for the **lpstat** command to complete its work, as it needs to interrogate various printer queues on the system. If you want to cancel a printing job before it is printed, first type **lpstat**. Under the printer name you will find an ID for your printing job. If this ID is b3prt-23 say, you can remove the job by typing **cancel** *b3prt-23*. To cancel all your jobs queued to a printer, type: **cancel** *(printer)* –**u***(user*).

5.4.7 Comparing and concatenating files: diff, cat

If you do **diff** *file1 file2* you will get a list of the differences between the two files. To take two files and put them end to end in a new file you use **cat** *file1 file2* > *newfile*. To put a file at the end of an existing file type **cat** *file1* >> *existingfile*.

5.4.8 Searching for and inside files: find, grep

The command **find** is used to find files with various attributes. It has many options. This makes it somewhat awkward for just searching for a file by name. For example, the command:

find . **–name** "*.dat" **–print**

finds all the files in the current directory and any directories below it that have a name which ends in .dat and then prints the result of the search at the terminal.

The program **grep** looks inside files for specified text. For example,

grep –e *Analysis model.dat*

will search the file model.dat for any occurrences of the text "Analysis" and then print at the terminal any lines containing the text. If you are searching in more than one file, as in **grep** *Analysis *.dat,* it will print the name of the file in which the text was found.

5.4.9 Changing the permissions on a file: chmod

The permissions on a file act to control who can read and change the file. This is described in full detail in the section on File Permissions (5.4.10). The **chmod** command allows the user to either specify a new set of permissions to activate for a file or to add to the existing access permissions for a file. The command itself is fairly simple, i.e. **chmod** *symbolic_mode_list filename*. However, the multiple ways to specify the permission mode set to be applied can be tricky. There are three levels controlling who has access to a file, these being user, group and other (sometimes called world), and each level has control of three attributes, these being: read, write and execute permissions.

The obsolescent method of specifying a mode set is to use a numeric mode (see section 5.4.10), which uses an octal number to represent the read, write and execute attributes for each of the three levels of control. The now preferred symbolic mode list allows the use of character strings with operators. The character strings may be quite complex, but at least they are close to being human readable.

The example that follows will illustrate several complex character permission masks that may be used to ensure that exactly the following file access permissions are set: the user (you) has all access permissions, users in the same group as you have only read and execute permissions and all other users have read only permissions.

Any of the following commands may be used to achieve the desired outcome:

chmod u=a,g=rx,o=r *filename* or
chmod a=r,ug+x,u+w *filename*

or perhaps the old way, is simpler IF you are sure that you have summed all the bits correctly!

chmod 754 *filename*

5.4.10 Controlling access to your data with file permissions

The **chmod** command (see section 5.4.9) changes the access permissions of one or more files according to the value of symbolic_mode_list (see a) or numeric_mode (see b). You can display the current permissions for a file with the **ls –l** command (see section 5.4.2).

a) Symbolic Mode List

A symbolic_mode_list is a comma-separated list of operations in the following form. Whitespace is not permitted.

[who]op[permission][,...]

The variable fields can have the following values:

who	One or more of the following letters:
u	Modify permissions for user (owner).
g	Modify permissions for group.
o	Modify permissions for others.
a	Modify permissions for all users (a is equivalent to ugo).
op	Required component of mode list; must be one of the following symbols:
+	Add permission to the existing file mode bits of who.
–	Delete permission from the existing file mode bits of who.
=	Replace the existing mode bits of who with permission.
Permission	One or more of the following letters:
r	Add or delete the read permission for who.
w	Add or delete the write permission for who.
x	Add or delete the execute file (search directory) permission for who.
s	Add or delete the set-owner-ID-on-file-execution or set-group-ID-on-file-execution permission for who. Useful only if u or g is expressed or implied in who.
t	Add or delete the save-text-image-on-file-execution (sticky bit) permission. Useful only if u is expressed or implied in who. See Unix man page **chmod(2)**.
X	Conditionally add or delete the execute/search permission as follows: + If file is a directory, add or delete the search permission to the existing file mode for who. (Same as x.) + If file is not a directory, and the current file permissions include the execute permission (**ls –l** displays an x or an s) for at least one of user, group, or other, then add or delete the execute file permission for who. + If file is not a directory, and no execute permissions are set in the current file mode, then do not change any execute permission.

Or one only of the following letters:

u	Copy the current user permissions to who.

g	Copy the current group permissions to who.
o	Copy the current other permissions to who.

The operations are performed in the order specified, and can override preceding operations specified in the same command line.

If who is omitted, the r, w, x, and X permissions are changed for all users if the changes are permitted by the current file mode creation mask (see Unix man page **umask(1)**). The s and t permissions are changed as if a was specified in who.

Omitting permission is useful only when used with = to delete all permissions.

b) Numeric Mode (Obsolescent)

Absolute permissions can be set by specifying a numeric_mode, an octal number constructed from the logical OR (sum) of the following mode bits:

Miscellaneous mode bits:

4000	(= u=s)	Set user ID on file execution (file only)
2000	(= g=s)	Set group ID on file execution (file only)
1000	(= u=t)	Set sticky bit; see chmod(2)

Permission mode bits:

0400	(= u=r)	Read by owner
0200	(= u=w)	Write by owner
0100	(= u=x)	Execute (search in directory) by owner
0040	(= g=r)	Read by group
0020	(= g=w)	Write by group
0010	(= g=x)	Execute/search by group
0004	(= o=r)	Read by others
0002	(= o=w)	Write by others
0001	(= o=x)	Execute/search by others

5.4.11 Getting information: whoami, pwd, finger, env, jobs, ps, history

The command **pwd** prints the name of your current working directory on your screen. Typing **history** will give a list of your most recent commands. This information is stored in the file ~/.sh_history.

Typing **whoami** will tell you whose account you are currently using. The command **finger** will get you information on all users currently logged onto your system. You can also find out about a single person. For example, **finger** *zmythe* will give you information about the

user zmythe. **finger** will make a guess for you, so you can use it to find out e-mail addresses. Thus, **finger** *smyth* will probably give you information on zmythe whose name you had forgotten how to spell. You can also use finger to inquire about people elsewhere on the internet using their e-mail address as in **finger** *teds@dsto.defence.gov.au*. However, this feature is not enabled on the majority of DSTO machines and certainly won't work through the DSTO firewall.

A list of your environmental variables can be obtained via **env**. The commands **ps** and **jobs** tell you the current processes and jobs that are owned by you. To get all processes use **ps –ef** or **ps –el**.

5.4.12 Killing jobs and processes: kill

To kill or terminate a job you type **kill** %*number* where number is the job number as reported by **jobs**. To kill a process, use **kill** *processnumber* where the *processnumber* is reported by **ps**. Watch out! Don't kill a process unless you know what you are doing.

5.4.13 Repeating and changing commands with emacs style editing

The Posix shell on the HP machines has been configured to use emacs style command line editing (**set –o** emacs). It uses many control and escape sequences such as: ^R, ^P, ^N, ^A, ^B, ^D, ^E, ^F, ^H.

You can repeat the command you just gave using ^P^M. The command ^Rtext finds the last command that contained "text". Thus, for example, the command ^Rls finds the previous command that contained ls in your history list (see 5.4.11) and ^Rr finds the last command that contained an "r", for example a pafrun command.

5.4.14 Pipes and redirects: |, >, >>, <

Many UNIX programs take input, transform it, and then give output. This is usually set up so that the input comes from a file. If no filename is given, input comes from what is called standard input. The output goes to standard output. Almost always, standard input is the terminal keyboard and standard output is the terminal screen.

Often you want to control the input and output of programs. This is done with pipes and redirects. What a pipe does is connect standard input to standard output. In other words, it takes the output of one command and makes it the input to the next command. For example,

ll | more

takes the output from **ll** and sends it to the program **more** which displays it page by page.

The redirects work similarly, but they connect standard input or standard output to a file. The greater than sign > sends the output of the previous command to a file. For example,

ls > *file1*

sends the result of **ls** (i.e. a list of the files in the directory) to a file called file1. The double greater than sign >> is similar, but it appends the result to the file.

The less than sign < takes the file that is after it and uses its contents for input to the command that precedes it. An example is

aprogram < *commd.in* > *commd.out*

which takes the contents of the file commd.in and sends it as input to the program aprogram. After aprogram processes the file, the result is put in the file commd.out.

The command **cat** has a special role in all of this. It takes a file and sends it to standard output. Thus **cat** *file* sends the file to the terminal screen and **cat** *file1* >> *file2* puts the contents of file1 at the end of file2. You can put a group of files together in a single file using

cat *file1 file2 file3* > *newfile*

You can combine many commands, pipes and redirects. For example,

ps –ef | grep *patran* > *patran_process.list*

lists all the processes running and pipes this result to **grep** *patran*. This looks for all the lines that contain the word patran and then sends this result to a file named patran_process.list. One of the main organising principles of Unix is to have simple commands that can be easily linked together in various ways to get powerful results.

5.4.15 Compressing files: tar, compress, uncompress, gzip, gunzip

All Unix systems have **tar** and **compress** commands and today most will also have **gzip**. The **tar** command will combine a list of files and directories into an archive, but it does not compress the data. The **compress** command will compress each file it is given. The **gzip** command in general will compress files better than the **compress** command and also has the advantage of being compatible with the zip engines used on other platforms. The

uncompress and **gunzip** commands are used to expand the data after it has been compressed via **compress** and **gzip**, respectively.

The default file extensions for **tar** archives, **compress** and **gzip** files are .tar, .Z and .gz, respectively. Also note that **gunzip** also understands the extension .tgz to be a zipped **tar** archive and .taz to be a compressed **tar** archive.

The tar command saves and restores archives of files on a magnetic tape, a flexible disk, or a regular file. The default archive file is /dev/rmt/0m. See the –f option below. Its actions are controlled by the key argument.

The format of the tar command is:

tar [–]**key** [arg ...] [*file* | –C *directory*] ...

or specifying some of the more commonly used keys and arguments:

tar [c | x | u | t] [v] [f *archive_file*] *file_list…*

where:

c	create archive
x	extract files from archive
u	update or add files to archive
t	test the archive
v	verbose mode
f	specify archive file (default /dev/rmt/0m)
archive_file	name of the archive file to open
file_list…	list of files to be added to or extracted from the archive

The following commands compress and uncompress files and directory subtrees as indicated:

a) **compress** reduces the size of the named files using adaptive Lempel-Ziv coding. If reduction is possible, each file is replaced by a new file of the same name, with the suffix .Z added to indicate that it is a compressed file. Original ownership, modes, access, and modification times are preserved. If no file is specified, or if – is specified, standard input is compressed to the standard output.

b) **uncompress** restores the compressed files to original form. Resulting files have the original filename, ownership, and permissions, and the .Z filename suffix is removed. If no file is specified, or if – is specified, standard input is uncompressed to the standard output.

The usage is:

compress *file_list*

uncompress *file_list*

The command gzip reduces the size of the named files using Lempel-Ziv coding (LZ77). Whenever possible, each file is replaced by one with the extension .gz, while keeping the same ownership modes, access and modification times. If no files are specified, or if a file name is "–", the standard input is compressed to the standard output. The gzip command will only attempt to compress regular files. In particular, it will ignore symbolic links.

By default, gzip keeps the original file name and timestamp in the compressed file. Compressed files can be restored to their original form using gzip –d or gunzip.

The usage of the commands is:

gzip [–acdfhlLnNrtvV19] [–S suffix] [name ...]
gunzip [–acfhlLnNrtvV] [–S suffix] [name ...]

and the more typically:

gzip *file_list*
gunzip *file_list*

5.4.16 Remote login and file copying operations: rlogin, rcp and .rhosts

Remote operations, such as logging in on a remote machine with the **rlogin** command and copying of data with the **rcp** command, depend on the existence and proper configuration of a .rhosts file on the remote host. This remote .rhosts file must contain an entry allowing the current user on the local host to have access to the account on the remote host. The file format is a simple list of entries, one per line, listing each hostname/username pair that is granted remote access to the machine where the file resides. For example, for a user, *goldylocks*, with accounts on three different hosts, *jayted*, *armadillo* and *poison*, the following entries in the .rhosts files on each of the hosts will allow remote login and copy operations to be performed between any of the hosts.

jayted *goldylocks*
armadillo *goldylocks*
poison *goldylocks*

Some hints on setting up your .rhosts file:

a) Some machines require the last entry to be terminated with the new line character, so make sure that it's there.
b) Also some machines, i.e. like SGI's, don't define the IP network domain, and hence will require the host names to be fully resolved.
c) It can be useful to include the local host hostname/username pair in the list, as in the example above, thus making the file "universal" and therefore making it quick to update or recover from/to any of the hosts in your "network" if it is lost or corrupted.

Once the .rhosts file is sorted out, it then becomes a fairly straightforward matter to perform **rlogin** or **rcp** operations. The syntax for **rlogin** is: **rlogin** *hostname*. e.g. using the above example .rhosts file, for a user *goldylocks* on *jayted*, the command

rlogin *armadillo*

will log them into the *goldylocks* user account on *armadillo* without prompting for a password.

The **rcp** command allows the users to copy data between the user accounts on any of the authorised hosts in the .rhosts file using similar syntax to **cp**, the copy command. The format of the **rcp** command is:

rcp *user@host:source_file user@host:/dest_file*

The defaults for the user and host are the current username and hostname. Both the user and host may be omitted where the source or destination is the local host. Using the above example .rhosts file, for the user *goldylocks* logged into jayted the following command:

rcp *armadillo:rem_file .*

will copy *rem_file* from the host *armadillo* to the current directory on *jayted*. An example using the full command syntax is, for the user *goldylocks* on host *jayted*, the command:

rcp *goldylocks@poison:modeldir/datafile armadillo:otherdir/data_from_afar*

which will copy the file *datafile* from the user account *goldylocks* on the remote host *poison* into the file *data_from_afar* in the directory *otherdir* on the remote host *armadillo* under the user account *goldylocks*.

6. CSAF FEA Applications

This section details the specific commands that are required in order to use the various FEA and associated applications on the CSAF. Commands to perform pre- and post-processing are covered in section 7. The commands are separated into standard vendor components and customised CSAF commands. Table 4 shows the matrix of vendor commands and Table 5 shows the custom commands developed by the CSAF that are available on each server. Users are expected to be familiar with these commands. The Section column in each table refers to the relevant section in this text in which the detailed descriptions and syntax for use of these commands is given. Relevant references are listed in sections 9.1, 9.2, 9.3, 9.4, 9.5 and 9.7.

Table 4. Application command matrix

Application	Command	Name of Server					Section
		jayted	bigted	farted	poisson2	ardvark	
MSC.Nastran	nastran	x			x		6.1.1
ABAQUS	abaqus	x			x		6.2.1
PAFEC	pafrun		x				6.3.1

Table 5. CSAF custom application command matrix

Application	Command	Name of Server					Section
		jayted	bigted	farted	poisson2	ardvark	
MSC.Nastran	seenast	x					6.1.3
MSC.Nastran	nastbat				x		6.1.5
MSC.Nastran	fedel	x	[x]		x		6.1.6
ABAQUS	fedel	x	[x]		x		6.1.6
PAFEC	pafdel		x				6.3.2
PAFEC	paflev		x				6.3.4

6.1 MSC.Nastran

6.1.1 Running a finite element analysis – the nastran command

The installed versions of MSC.Nastran are 70.5, 70.7 and 2001. The default version is the latest release available on the system. MSC.Software provides a number of alternate names to access these commands. Assuming that the most recent version is 2001, then the respective lists of commands to use that particular version are:

nastran, msc2001 nastran, nast2001, …

and

msc707 nastran, nast707, …

msc705 nastran, nast705, …

The normal command line to invoke MSC.Nastran is of the form:

nastran (*input_data_file*) [keyword1 = ***value1*** keyword2 = ***value2*** ...]

A more complete listing of the MSC.Nastran 2001 keywords is given in the following section (6.1.2). However, as a starting guide, the following short list contains some of the more routinely used keywords when invoking **nastran** on the CSAF. Note that all keywords are described in section 6.1.2, except for the last, **sys166**, which is described in Table 6:

memory=(*mem*), **old**=no, **news**=no, **scr**=yes, **buffsize=**(*number*)**, smem=**(*number*) and **sys166=0**

Table 6. System Cell Number 166 Summary (from [4], Section 1, Table 1-2.)

System Cell Number	System Cell Name	Function and Reference
166	---	Controls sparse symmetric decomposition. Sum the desired values. (Default=1) 0: No action 1: If insufficient core is encountered, then switch to conventional decomposition and continue. (Default) 2: Print diagnostics 4: Do not issue fatal message if maximum ratios are exceeded. Although high maximum ratios may be printed, they will not cause job termination. This applies to the DCMP, DECOMP, REIGL, and LANCZOS modules.

6.1.2 Descriptions of MSC.Nastran version 2001 keywords

This section lists short descriptions of the **nast2001** command keywords, as given by the command: **nastran help all**. The command output has the keywords divided into the following sections: a) General, b) Database, c) Numeric method, d) Distributed execution, e) Queuing, f) Remote execution, and g) Administrative.

a) General keywords:

after=*time*

UNIX systems only.

Hold execution until the specified time. Also used by "**submit**". See the **at(1)** man page for further information. There is no default for this value.

append={"yes", "no"}

Concatenate the F06, F04, and LOG files. The file type of the concatenated file is "out". The default is "**append**=no".

batch={"yes", "no"}

UNIX systems only.

Execute in background. Overridden by "**aft**" and "**queue**". The default is "**batch**=yes". If running in an NQS/NQE batch job, the default is "**batch**=no".

delete={"yes", "no", "all", "*jid*",*type*}

Comma-separated list of MSC.Nastran output file types to delete at job completion. "**delete**=yes" is equivalent to "**delete**=*f06,f04,log*"; "**delete**=all" deletes all standard output files; "**delete**=*jid*" is ignored if not in server mode. The default is "**delete**=no".

delivery={*pathname*,"MSCDEF"}

Alternate delivery database prefix or "MSCDEF". This keyword overrides all MSC-supplied solution sequences. The file type of the file must be ".MASTERA". If a directory is not specified and the file does not exist in the current directory, the default delivery database directory is assumed. The default is "**delivery**=MSCDEF".

diag=*flag*[,*flag*,...]

Set MSC.Nastran diagnostic flags. The values may be separated with spaces or commas. These diagnostics are set in addition to any diagnostics set via the Executive Control "DIAG" statement in the input data file.

display=*display_name*

UNIX systems only.

Specify a display for XMONAST. This value may also be set with the DISPLAY environment variable. The environment variable overrides the RC files; the command line overrides the environment variable.

executable=*pathname*

Alternate solver executable. This keyword overrides all architecture and processor selection logic. If a directory is not specified and the file does not exist in the current directory, the default executable directory is assumed.

gmconn=*pathname*

Geometric evaluator connection file. There is no default value.

ishellext=type=*processor*[,type=*processor*,...]

Define processor associations. This list defines the processor used by MSC.Nastran to execute an ISHELL program based on its file type. This value may also be set with the MSC_ISHELLEXT environment variable. The environment variable overrides the RC files; the command line overrides the environment variable. The default is operating system dependent.

ishellpath=*path*

Additional path for ISHELL module. MSC.Nastran will attempt to locate ISHELL module programs on this path before searching the standard PATH. The path value may include environment variable references, e.g. "**jidpath**=$HOME/mytpl:$HOME". This value may also be set with the MSC_ISHELLPATH environment variable. The environment variable overrides the RC files; the command line overrides the environment variable. The default is the directory containing the input data file.

jid=*pathname*

Input data file. An input file must be defined on the command line. There is no default value.

jidpath=*path*

Additional path for input data file searching. MSC.Nastran will attempt to locate the input data file on this path if the input data file name does not have a directory component and cannot be found in the current directory. The path value may include environment variable references, e.g. "**jidpath**=$HOME/mytpl:$HOME". This value may also be set

with the MSC_JIDPATH environment variable. The environment variable overrides the RC files; the command line overrides the environment variable. There is no default path.

jidtype=*file_type*

Default file type for the input data file. This keyword may be specified on the command line, or in the system RC files, the HOME RC file, or an RC file explicitly named with the "**rcf**" keyword. The default is "**jidtype**=.dat"; if not defined by the user, "**jidtype**" shall be set to the file type of the actual input data file.

memory={*memory_size*,"estimate"}

Size, in words, of open core. If "**memory**=estimate" is specified, the memory size will be determined by **estimate** (see section 6.1.4). Otherwise, a positive non-zero value must be specified. The default is "**memory**=estimate". Valid memory specifications are nB, nW, nKB, nKW, nK, nMB, nMW, nM, nGB, nGW, nG, n*physical, and n*virtual.

memorydefault=*memory_size*

Default size, in words, of open core. This value is used if and only if "**memory**=estimate" was specified and the **estimate** utility (see section 6.1.4) failed. The default is "**memorydefault**=8MW". Valid memory specifications are nB, nW, nKB, nKW, nK, nMB, nMW, nM, nGB, nGW, nG, n*physical, and n*virtual.

memorymaximum=*memory_size*

Maximum size, in words, of open core. The default is machine dependent; on this platform it is "**memorymax**=0.8*physical" (see Appendix C in the "MSC.Nastran Installation and Operations Guide" [1]). Valid memory specifications are nB, nW, nKB, nKW, nK, nMB, nMW, nM, nGB, nGW, nG, n*physical, and n*virtual. Note: If you refer to physical or virtual memory, the physical or virtual memory must be known, otherwise this limit will be ignored. See also the "**s.pmem**" and "**s.vmem**" keywords.

ncmd=*command_string*

Command to notify user that the job has completed. The default notification will use the Korn shell print command if "**batch**=no" or write(1) otherwise.

news={"yes", "no", "auto"}

Display site's news and release notes in the F06 file. The "**news**=auto" keyword will only print the news file if it has changed since the last time it was printed. The default is "**news**=yes".

notify={"yes", "no"}

Notify user at job completion. Use "**ncmd**" to set the notify command. The default is "**notify**=yes". If the application is running in an NQS/NQE batch job, the default is "**notify**=no".

old={"yes", "no"}

Save existing output files. If "**old**=yes", the existing output files are renamed with a version number; if "**old**=no", the existing output file are deleted. The default is "**old**=yes".

oldtypes=*file_types*

A comma-separated list of user file types, excluding the ".", that will be versioned if "**old**=yes" is specified, or deleted if "**old**=no" is specified. These file types are in addition to the standard file types, e.g., f04, f06, log, etc. This value may also be set with the MSC_OLDTYPES environment variable. The environment variable overrides the RC files; the command line overrides the environment variable. There is no default value.

out=*pathname_prefix*

Alternate output file prefix. If the prefix is a directory, 'jid-basename' is appended. The default is "**out**=./'jid-basename'".

post=*command_string*

User command(s) run at job completion. Every occurrence of "**post**" in an RC file or on the command line will be executed. Setting a null value, i.e., "**post**=", will erase the currently saved commands. There is no default value.

pre=*command_string*

User command(s) run at job start. Every occurrence of "**pre**" in an RC file or on the command line will be executed. Setting a null value, i.e., "**pre**=", will erase the currently saved commands. There is no default value.

processor=*file_type*

File type of alternate executable. This keyword overrides the default processor subtype selection logic. There is no default value.

rcf=*pathname*

File containing additional keywords. Replaces the local RC file. This keyword may only be specified on the command line. There is no default value.

symbol=*name*=*value*

Assign a value to a symbolic name; the name may be used in a filename reference on the command line, in an RC file, or in the data file. All environment variables can be used as symbols. A symbol definition can include an environment variable reference, e.g. "**symbol**=*mysymbol=$HOME/mydir*". The environment variable overrides the RC files; the command line overrides the environment variable.

system=*number*

Assign a value to a system cell. The keyword can be specified as either "**SYSn**=*value*" or "**SYSTEM(***n***)**=*value*" where *n* is the system cell being set. For example, either "**SYS107**=*2*" or "**SYSTEM(107)**=*2*" will set SYSTEM cell 107 to 2. If the latter form is used on the command line, it must be quoted to prevent interaction with the shell. There is no default value.

trans={"yes", "no", "auto"}

Indicates the XDB file is to be converted to neutral format using the TRANS utility if the "**node**" keyword is not specified. Indicates the XDB file is to be copied from the remote node if the "**node**" keyword is specified. If "**trans**=yes", use TRANS/RECEIVE to copy the XDB file from the remote node. If "**trans**=auto", use TRANS/RECEIVE only if needed; otherwise use a binary copy. If "**trans**=no", do not copy the XDB file from the remote node. The default is "**trans**=auto".

version=*version_number*

Selects the version to be run, e.g., "**version**=*2001*". This keyword may only be specified on the command line or in the command initialisation file. The default is the latest installed version.

xmonitor={"yes", "no", "kill"}

UNIX systems only.

Run XMONAST to monitor the job's progress. "**xmonitor**=kill" will terminate XMONAST at job completion. The default is "**xmonitor**=no".

b) Database keywords:

bpool=*number*

Number of GINO blocks allocated to buffer pool (in open core). The default is machine dependent (see Appendix C in the "MSC.Nastran Installation and Operations Guide" [1]). This value may also be set with the "**sys114**" keyword.

buffsize={*number*,"estimate"}

Size, in words, of a GINO buffer. If "**buffsize**=estimate" is specified, the BUFFSIZE will be determined by **estimate** (see section 6.1.4). Otherwise, the value must be one plus a multiple of the disk block size in words. The default is machine dependent (see Appendix C in the "MSC.Nastran Installation and Operations Guide" [1]). This value may also be set with the "**sys1**" keyword.

cpyinput=0 or 1

"Copy Input File" flag, where 0 means "no", 1 means "yes". The default is 0 (no copy). This value may also be set with the "**sys305**" keyword.

dbs=*pathname_prefix*

Alternate user database prefix. Overrides "**scratch**=yes". If the prefix is a directory, 'jid-basename' is appended. The default is "**dbs**=./'jid-basename'". When used for remote processing, this pathname must be valid on the remote system. If "dmparallel" >= 1, unique database directories may set for each task using the ":" character to separate entries; the names will be paired with the tasks in round-robin order.

scratch={"yes", "no", "mini", "post"}

Delete database files at job completion. "**scratch**=yes" is equivalent to the FMS statement "INIT MASTER(S)". Setting the "dbs" keyword in an RC file or on the command line overrides "**scratch**=yes". "**scratch**=mini" selects the "mini database" option, this reduces the size of the database if it will only be used for data recovery restarts. The default is "**scratch**=no".

sdball={*db_size*,"estimate"}

Default size, in GINO blocks, of the DBALL DBset. This number may also be specified in words or bytes if followed by a unit modifier. If "**sdball**=estimate" is specified, the DBset size will be determined by **estimate** (see section 6.1.4). The FMS INIT statement overrides this keyword. The default is machine dependent (see Appendix C in the "MSC.Nastran Installation and Operations Guide" [1]). Valid unit modifiers are B, W, KB, KW, K, MB, MW, M, GB, GW, G.

sdirectory=*pathname*

Directory for scratch files. This is the default directory for user database files if "**scratch**=yes". The default is taken from the TMPDIR environment variable if it is defined, otherwise it is machine dependent (see Appendix C in the "MSC.Nastran Installation and Operations Guide" [1]). When used for remote processing (see "node"), this pathname must be valid on the remote system. If "**dmparallel**" >= 1, unique scratch directories may set for each task using the ":" character to separate entries; the names will be paired with the tasks in round-robin order.

smaster=*db_size*

Default size, in GINO blocks, of the MASTER DBset. This number may also be specified in words or bytes if followed by a unit modifier. The FMS INIT statement overrides this keyword. The default is machine dependent (see Appendix C in the "MSC.Nastran Installation and Operations Guide" [1]). Valid unit modifiers are B, W, KB, KW, K, MB, MW, M, GB, GW, G.

smemory=*number*

Number of GINO blocks of SCRATCH and SCR300 files kept in open core. The default is machine dependent (see Appendix C in the "MSC.Nastran Installation and Operations Guide" [1]).

sscr={*db_size*,"estimate"}

Default size, in GINO blocks, of the SCRATCH and SCR300 DBsets. This number may also be specified in words or bytes if followed by a unit modifier. If "**sscr**=estimate" is specified, the DBset size will be determined by **estimate** (see section 6.1.4). The FMS INIT statement overrides this keyword. The default is machine dependent (see Appendix C in the "MSC.Nastran Installation and Operations Guide" [1]). Valid unit modifiers are B, W, KB, KW, K, MB, MW, M, GB, GW, G.

sysfield=*string*

Specifies a default value for the SYS keyword on all FMS ASSIGN statements associated with DBsets. The string specification is not case sensitive. There is no default value.

use_aio={"yes", "no"}

HP-UX systems only.

Enables the HP AIO library for database I/O. This value may also be set with the USE_AIO environment variable. The environment variable overrides the RC files; the

command line overrides the environment variable. The default is "**use_aio**=no". The CSAF default is "**use_aio**=yes"

c) Numeric method keywords:

config=*number*

Configuration number. This value is used to select the machine timing data. You may need to set this value if USER WARNING MESSAGE 6080 is encountered. See the "MSC.Nastran Installation and Operations Guide" [1] for information on UWM 6080 and generating timing data. The default is machine dependent. This value may also be set with the "**sys28**" keyword.

fbsmem=*number*

Reserves memory for faster solution of Lanczos method of eigenvalue extraction. See the "MSC.Nastran Quick Reference Guide" [4], Section 5, EIGRL Bulk Data entry. This value may also be set with the "**sys146**" keyword.

fbsopt=*number*

Selects Forward-Backward Substitution methods. This value may also be set with the "**sys70**" keyword.

massbuf=*number*

Half the number of buffers to be set aside for storing the mass matrix in memory. This value may also be set with the "**sys199**" keyword.

mpyad=*number*

Selects/deselects multiplication method. This value may also be set with the "**sys66**" keyword.

newhess=*number*

Request complex eigenvalue method. See the "MSC.Nastran Quick Reference Guide" [4], Section 5, EIGC Bulk Data entry, and the "MSC.Nastran Numerical Methods User's Guide" [13]. This value may also be set with the "**sys108**" keyword.

numseg=*number*

Lanczos High Performance Option: number of segments. See also the NUMS field on the EIGRL Bulk Data Entry. On a DMP job, the default is "**numseg**=*dmparallel_value*". This value may also be set with the "**sys197**" keyword.

parallel=*number*

Shared memory parallel (SMP) processing selection. This keyword defines the number of processors to be used in the shared memory parallel modules. There is no default value. This value may also be set with the "**sys107**" keyword.

rank=*number*

Set the rank update and minimum front sizes in the sparse modules. The default is machine dependent (see Appendix C in the "MSC.Nastran Installation and Operations Guide" [1]). This value may also be set with the "**sys198**" and "**sys205**" keywords.

real=*memory_size*

The amount of open core memory that certain numerical modules will be restricted to. This keyword may be used to reduce paging at the potential expense of spilling. Valid memory specifications are nB, nW, nKB, nKW, nK, nMB, nMW, nM, nGB, nGW, nG, n*physical, and n*virtual.

sparse=*number*

Sparse matrix method selection. This value may also be set with the "**sys126**" keyword.

usparse=*number*

Unsymmetric sparse matrix method selection. This value may also be set with the "**sys209**" keyword.

d) Distributed execution keywords (only if "dmparallel" is specified):

dmparallel=*number*

Distributed systems only.

Distributed memory parallel (DMP) processing selection. This keyword defines the number of tasks to be used in a distributed memory parallel analysis. This value may only be set on the command line. The value must be null to cancel distributed processing, or a

number greater than 0 to request distributed processing. See also the "hosts" keyword. There is no default value.

hostovercommit={"yes", "no"}

Distributed systems only.

Allow this job to assign more tasks to a host than processors. This does not prevent other MSC.Nastran jobs or users from using the processors. The default is "**hostovercommit**=no".

hosts={*host*:*host*:...}

Distributed systems only.

List of distributed memory processing hosts. This keyword defines the list of candidate hosts to be used for distributed memory parallel analysis. See also the "**dmparallel**" keyword. The default is machine dependent; on this platform it is the current host (see Appendix C in the "MSC.Nastran Installation and Operations Guide" [1]).

mergeresults={"yes", "no"}

Distributed systems only.

Indicates results are to be merged by master node.

numseg=*number*

Lanczos High Performance Option: number of segments. See also the NUMS field on the EIGRL Bulk Data Entry. On a DMP job, the default is "**numseg**=*dmparallel_value*". This value may also be set with the "**sys197**" keyword.

rcmd=*pathname*

nastran command pathname on the remote or distributed nodes. This keyword may be specified on the command line, RC file, or in the command initialisation file. If specified in an RC file, the "**version**" keyword must be defined on the command line or in the command initialisation file.

sdirectory=*pathname*

Directory for scratch files. This is the default directory for user database files if "**scratch**=yes". The default is taken from the TMPDIR environment variable if it is defined, otherwise it is machine dependent (see Appendix C in the "MSC.Nastran Installation and Operations Guide" [1]). When used for remote processing (see "node"), this pathname

must be valid on the remote system. If "**dmparallel**" >= 1, unique scratch directories may set for each task using the ":" character to separate entries; the names will be paired with the tasks in round-robin order.

slavejob={"yes", "no"}

Distributed systems only.

Specifies the "slave job" is to be used on slave nodes. For MSC.Nastran V69.2 the default (and only legal) value is "**slavejob**=yes". For MSC.Nastran V70.5 and later, the default is "**slavejob**=no".

slaveout={"yes", "no"}

Distributed systems only.

Specifies the output files from slave tasks are to be returned to the local node. The default is "**slaveout**=no".

username=*name*

Alternate username for distributed, remote, or queued jobs. The default is the current user name.

e) Queuing keywords (only if "queue" is specified):

cpulimit=*seconds*

UNIX systems only.

The NQS/NQE per-request CPU limit (–lT). See also "**ppcdelta**". This is the maximum total CPU time allocation for all processes in the job. This value may be input in the form [[hours:]minutes:]seconds[.fractionalseconds] where hours and minutes are integral values; it is always saved as seconds. There is no default value.

ppcdelta=*seconds*

UNIX systems only.

Used to calculate "ppc"; "**ppcdelta**" is subtracted from "**cpulimit**". This value may be input in the form [[hours:]minutes:]seconds[.fractionalseconds] where hours and minutes are integral values; it is always saved as seconds. There is no default value.

ppmdelta=*memory_size*

UNIX systems only.

Used to calculate "ppm"; "**ppmdelta**" is added to "**memory**". If the value is greater than 1000, the value is added to "**memory**". If the value is less than **1000**, it is assumed to be a percentage of the executable size. The default is "**ppmdelta**=*105*". Valid memory specifications are nB, nW, nKB, nKW, nK, nMB, nMW, nM, nGB, nGW, nG, n*physical, and n*virtual.

prmdelta=*memory_size*

UNIX systems only.

Used to calculate "prm"; "**prmdelta**" is added to "ppm". The default is "**prmdelta**=*5120*". Valid memory specifications are nB, nW, nKB, nKW, nK, nMB, nMW, nM, nGB, nGW, nG, n*physical, and n*virtual.

qclass=*string*

UNIX systems only.

May be used to specify a special class for queued jobs. There is no default value.

qoption=*string*

UNIX systems only.

May be used to specify any queue options not otherwise handled. There is no default value.

queue=*name*

UNIX systems only.

Submit job to queue_name. Use "**submit**" to define the queue command. There is no default value.

submit=[*queue_name1*[,*queue_name2*,...]=]*queue_command*

UNIX systems only.

Queue submittal command(s). Required by "**queue**". Zero or more queue names may be specified. A queue name may be up to 15 characters long. *queue_command* is the command that will submit the job to the queue. See the "MSC.Nastran Installation and Operations

Guide" [1] for complete details on this keyword. This keyword can only be set in an RC file. There is no default value.

username=*name*

Alternate username for distributed, remote, or queued jobs. The default is the current user name.

f) Remote execution keywords (only if "node" is specified):

node=*name*

Execute the job on the specified node. Use "**username**" to specify an alternate username on the remote node. This keyword may only be specified on the command line. There is no default value.

rcmd=*pathname*

nastran command pathname on the remote or distributed nodes. This keyword may be specified on the command line, RC file, or in the command initialisation file. If specified in an RC file, the "**version**" keyword must be defined on the command line or in the command initialisation file.

username=*name*

Alternate username for distributed, remote, or queued jobs. The default is the current user name.

xhost={"yes", "no"}

UNIX systems only.

Execute "**xhost** '*node*'" on this node. "**xhost**=yes" may be required if "**xmonitor**=yes" or "**xmonitor**=kill" is specified. The default is "**xhost**=no".

g) Administrative keywords:

accmd=*string*

Command line arguments for the MSC Account logging program, **acct**. This keyword can only be set in the command initialisation file or an RC file.

acct={"yes", "no"}

Perform solution accounting. The default is "**acct**=no".

acdata=*string*

Site defined accounting data. See your system administrator to determine how this keyword is to be used.

acid=*string*

Site defined account ID. See your system administrator to determine how this keyword is to be used.

acvalid={{"f", "w"}*regular-expression*,`*command*`}

Indicates account ID validation is to be performed. See the "MSC.Nastran Installation and Operations Guide" [1] for further information. This keyword can only be set in the command initialisation file or an RC file.

authclear={"no", "msc", "lm", "both", "yes"}

Clear previous authorisation information flag. The value indicates what level of previous "AUTHORIZE" information is to be cleared. The default is "authclear=no".

authinfo=*number*

Information message flag. Values greater than zero indicate additional informational messages are to be displayed. The default is "**authinfo**=0".

authorize={*pathname*,[*port*]@*host*,"*demo*"}

Authorisation file pathname, nodename of network license server, or "demo" to request the demo license. The environment variable MSC_LICENSE_FILE overrides the RC files; the command line overrides the environment variable.

authqueue=*number*

Maximum time (in minutes) to wait for licenses if they are currently in use by another job. The default is "**authqueue**=20".

lock=keyword

Prevents a keyword from being modified. Once a keyword is locked, e.g., "**lock**=acct", any attempt to set the keyword in an RC file or on the command line will be silently ignored. The lock keyword can be locked, i.e., "**lock**=lock".

msgcat=*pathname*

Pathname of binary message catalogue. If a directory is not specified and the file does not exist in the current directory, the default executable directory is assumed.

Users should also see the section 6.1.4 on tuning analyses.

6.1.3 Displaying MSC.Nastran license usage – seenast

This is the CSAF custom command script developed to display current MSC.Nastran license use on the server jayted. The command has no parameters. This command uses the FLEXlm license manager utility to interrogate the license server for the MSC.Nastran feature. Be warned: occasionally license tokens are returned prematurely to FLEXlm from some parts of an MSC.Nastran analysis, which leaves the analysis running in an indeterminate state and also then permits a new MSC.Nastran analysis to be started – even if the number of analyses exceeds the license limit. When this event occurs the "lost" job still continues using CPU, memory and disk resources, albeit very slowly, but it is not certain to complete successfully. It is recommended that a more thorough check be made on the state of MSC.Nastran jobs by using the command **hey** *anal* or **hey** *msc* (see section 7.7.4).

Usage is:

seenast

6.1.4 Tuning analysis for optimal performance on a system – estimate

The **estimate** utility is used to tune an MSC.Nastran analysis for more optimal performance on a particular system. It is more advantageous when applied to a larger analysis. It is important to note that, as both jayted and poisson2 are running 64-bit operating systems, a report generated using the default value of 32 is only of academic interest, hence the user must specify the word=64 option to obtain useful results.

Usage is:

estimate (*input-data-file)* (**word**=64) **[keyword options]**

where:

Command line parameters	Default
input-data-file	none
adapt=sid	none
bpool=value	37
buffsize=value	estimate
estimatedof={yes | no}	no
memory=value	estimate
method=sid	none
mode={estimate | modify | suggest}	suggest
mpc=sid	none
nastrc={yes | no}	yes
output=*output-data-file*	*input-data-file*
pause={yes | no | info | warn | fatal}	no
real=value	none
realdelta=value	12MB
report={normal | keyword | both}	normal
scratch={yes | no | mini}	no
spc=sid	none
suppress=value[,value,...]	none
smem=value	100
verbose={yes | no}	no
version=version	2001.0
wordsize={32 | 64}[,{32 | 64}]	32

6.1.5 Submitting MSC.Nastran jobs to batch queue – nastbat

This is the CSAF custom command script developed for the NIC (poisson2) to simplify the task of submitting an MSC.Nastran analysis to the NIC batch queuing system. The nastbat script will submit your job to the batch queues. At present, MSC.Nastran versions 70.5, 70.7 and 2001 are running on poisson2. This command invokes the system default version of MSC.Nastran, which is version 2001.

Usage is:

nastbat (*job_name*) (*cpu_time*) [*mem*]

where:

job_name your dat file

cpu_time estimated (maximum) cpu time. The batching system uses this value to select the queue to place the job in. The queue names are short, medium and long. The main limits for the queues are: a) short: 7200 seconds (2hours) CPU time, 2 jobs maximum and 500 MB of memory maximum; b) medium: 54000 seconds (15 hours), 8 jobs maximum and 1000 MB memory maximum; and c) long: 720000 seconds (200 hours), 6 jobs maximum and 1000 MB memory maximum.

mem amount of memory to allocate to the solver. May be specified in words (w) or bytes, e.g. 100kw, 900m, 5000k, etc.

To get the general usage statement for this command:

nastbat

See the **nastran** (6.1.1) command for more information on the memory and other MSC.Nastran options.

6.1.6 Cleaning up job output – fedel

This is the CSAF custom command script developed to remove output files from PAFEC, ABAQUS and MSC.Nastran jobs on the servers jayted, bigted and poisson2. The command identifies the type of job by looking for an input file, in the following order of precedence: *job_name*.DAT for a PAFEC job, *job_name*.inp for an ABAQUS job and finally *job_name*.dat for an MSC.Nastran job. If run without specifying a job name it will try to clean up all analysis output files in the current directory. Be warned: using the same job name for multiple analyses will result in the deletion of the input decks (*.dat or *.inp) of at least one of the analyses. This command works by identifying an analysis package by matching the input deck name and extension, i.e. *job_name*.DAT, then deleting ALL files matched by *job_name*.* except for the input deck, e.g. *job_name* *.DAT, and a database file, i.e. *job_name*.db, if it exists. It is important to note that additional files are removed with a PAFEC analysis (6.3.3).

Usage is:

fedel [*job_name*]

6.2 ABAQUS/Standard

6.2.1 Running an analysis with the abaqus command

The versions of ABAQUS/Standard (and Explicit) that are available on both jayted and poisson2 are 6.2-7 and 6.3-1. The default version is now 6.3-1 and it is available via the command **abaqus**. The previous version is available via the command **abq627**. The **abaqus**

command is the master command to invoke all of the ABAQUS options, such as running a job or processing the results. The following is a description of the ABAQUS analysis options for version 6.3-1. Note that some options may be different for version 6.2-7.

Usage is:

abaqus job=*job-name*
[**analysis** | **datacheck** | **parametercheck** | **continue** | **convert**={select | odb | all} | **python**=[*script-file*] | **recover** | **script**=[*script-file*] | **suspend** | **syntaxcheck** | **help** | **information**={environment | local | memory | release | status | system} | **whereami**]
[**input**=*input-file*]
[**user**={*source-file* | *object-file*}]
[**oldjob**=*oldjob-name*]
[**fil**={append | new}]
[**globalmodel**={*results file-name* | *output database name}*]
[**cpus**=*number-of-cpus*]
[**parallel**={loop | domain | supernode | tree}]
[**domains**=*number-of-domains*]
[**memory**=*memory-size*]
[**buffer**=*buffer-size*]
[**interactive** | **background** | **queue**=[*queue-name*][**after**=*time*]]
[**double**]
[**scratch**=*scratch-dir*]
[**startup**=*startup file-name*]
[**noenvstartup**]
[**output_precision** = {single | full}]

6.2.2 Descriptions of abaqus command line options.

All options are order independent. If none of these options is present, the **analysis** option is assumed. The **convert** option is an exception to the mutual exclusion rule: **convert** can appear with any option except **datacheck**, **parametercheck**, and **syntaxcheck**.

help

This option prints a summary of the abaqus command syntax.

information

Typing abaqus **information**=*value* prints helpful information about the installation and the environment that is in effect for this job.

whereami

This option prints the location of the ABAQUS release directory. It cannot be used with any other command line option.

job

The value of this option specifies the name of all files generated during the run and the name of files that are read in the **continue**, **convert**, and **recover** phases.

If this option is omitted from the command line, the user will be prompted for its value except for the informational options.

analysis

This option indicates that a complete ABAQUS analysis (or a restart of an ABAQUS analysis) is to be performed.

datacheck

This option indicates that the run is for data checking only. No analysis will be performed. If this option is used, all files necessary to continue the analysis are saved.

parametercheck

This option indicates that the run is for input parameter checking only (the *PARAMETER option must have been used). No analysis or data checking will be performed.

continue

This option indicates that the run is to begin at the point at which a previous **datacheck** run ended.

convert

The value of this parameter indicates which files will be post-processed.

Results can be converted either immediately following an analysis run, as a separate run subsequent to an analysis run, or while an analysis is running as follows:

1. To run an analysis including a subsequent conversion of the results, use the **convert** option in conjunction with the **job** and **analysis** options.
2. To convert the results of a previously run analysis, use the **convert** option in conjunction with the **job** option.

3. To convert results from a job that is currently running, use the **convert** option in conjunction with the **oldjob** option (to name the running job) and the **job** option (to supply a new name for the files generated by the **convert** option).

If **convert**=select, the ABAQUS/Explicit selected results file (*job-name.sel*) will be converted into a standard ABAQUS results file (*job-name.fil*). If the analysis was run in parallel with **parallel**=domain, the separate selected results files (job-name.sel.n) will be converted into a single results file (job-name.sel) prior to being converted into a standard ABAQUS results file.

If **convert**=odb, the output database (*job-name.odb*) will be converted using the postprocessing calculator (see ABAQUS/Standard User's Manual [24], Section 4.3.1, "The postprocessing calculator" and [26]). This conversion is necessary only if the types of output listed in "The postprocessing calculator," Section 4.3.1, are requested. If the analysis was run in parallel with **parallel**=domain, the separate output database files (job-name.odb.n) will be converted into a single output database (job-name.odb) prior to conversion by the postprocessing calculator.

If **convert**=state, the separate ABAQUS/Explicit state files (job-name.abq.n) will be converted into a single ABAQUS/Explicit state file (job-name.abq) if the analysis was run in parallel with **parallel**=domain.

If **convert**=all, the selected results file and the output database file will be converted.

python

This option indicates that Python scripting is to be done. If this option is not given a value, the Python interpreter is initialised. When this option is set equal to a script file name, the Python interpreter executes the instructions in the script file.

recover

This option applies only to ABAQUS/Explicit. It indicates that an analysis is to be restarted at the last available step and increment in the state file. This capability is available to restart a job after a catastrophic failure, such as exceeding a CPU time limit or a disk quota.

script

This option indicates that a parametric study is to be done. If this option is not given a value, the Python interpreter is initialised by importing the parametric study module. When this option is set equal to a script file name, the parametric study module is imported and the instructions in the parametric study script file are executed. If the script file contains references to other input files, these files must be located in the same directory

as the script file. The files created by the execution of the script file are placed in the directory from which the ABAQUS execution procedure is run.

suspend

This option is used to suspend an ABAQUS analysis job and temporarily release its license tokens to the free-token pool. (See the ABAQUS Site Guide [28] for details.)

syntaxcheck

This option indicates that the run is for checking the syntax of the input file only. This option does not use any license tokens. No analysis will be performed, and the **continue** option cannot be used to continue with an analysis. Only the data (.dat) and output database (.odb) files are generated for viewing.

information

This option specifies that the run includes information extraction. By default, the information is written to the screen. If a job-name is given or if the job is run in the background, the information text is written to the file *job-name.log*. The following information is output for all information requests: the current version, the directory in which ABAQUS is located, and the directory in which the information files are located.

If **information**=environment, the current settings of the environment file options are displayed.

If **information**=local, the local installation notes are output.

If **information**=memory, some suggestions for setting memory parameters for analysis jobs are output.

If **information**=release, information is provided about where to locate the current release notes.

If **information**=status, information is provided about where to locate the status reports.

If **information**=system, information is provided about system software and hardware resources (operating system level, compiler levels, processor type, graphics board, memory, etc).

input

This option is used to specify the input file name, which may be given with or without the .inp extension (if the extension is not supplied, ABAQUS will append it automatically). If this option is not supplied, the procedure will look for an input file called *job-name.inp* in

the current directory. If *job-name.inp* cannot be found, the procedure will prompt for the input file name.

user

This option specifies the name of a FORTRAN source or object file that contains any user subroutines to be used in the analysis. The name of the user routine may contain a path name and may be given with or without a file extension. If an extension is given, the program will take the appropriate action based on the file type. If the file name has no extension, the program will search for a FORTRAN source file. If the source file does not exist, an object file will be searched for instead. The execution procedure creates a shared library using the user subroutine file that is used by ABAQUS/Standard or ABAQUS/Explicit during execution.

oldjob

This option specifies the name of the files from a previous run from which a **restart** or *POST OUTPUT (ABAQUS/Standard only) run is to be started or from which results are to be imported. A file extension is not allowed. This option is required when the *RESTART, *POST OUTPUT, *SYMMETRIC MODEL GENERATION, or *IMPORT option in the input file indicates that data are to be read from the restart or the results file. The *oldjob-name* must be different from the current *job-name*.

fil

This option specifies whether the data from the results file of the *oldjob* specified in a **restart** run are included at the beginning of the new results file (default). If **fil**=new is used, the new results file will contain only the data from the point in the analysis where the restart occurred. This feature is used for ABAQUS/Standard runs to join the output from restarted analyses into a single, continuous results file. Non-restart jobs cannot use this feature to append results file output to an old results file; the ABAQUS/Append utility must be used for this purpose. Setting **fil**=new is not allowed for ABAQUS/Explicit runs.

globalmodel

This option specifies the name of either the global model's results file or the output database from which the results are to be interpolated to drive a submodel analysis. The file extension is required if both the results file and the output database exist. This option is required whenever the *SUBMODEL option or the *BOUNDARY, SUBMODEL option in the input file indicates that data are to be read from the global model's results.

cpus

This option specifies the number of processors to use during an analysis run. This feature is not available on all systems. Use **information**=local to learn about local multiprocessing capabilities. The default value for this parameter can be set in the environment file (see ABAQUS/Standard User's Manual [24], Section 3.4.1, "Using the ABAQUS environment settings" and [26]).

parallel

This option specifies the method to use for parallel processing in ABAQUS/Explicit and for parallel equation solution in ABAQUS/Standard.

For ABAQUS/Explicit the values domain and loop are applicable. If **parallel**=domain, the domain level method is used to break the model into geometric domains. If **parallel**=loop, the loop level method is used to parallelise low-level loops. See "Parallel execution," Section 7.9.1 of the ABAQUS/Explicit User's Manual [26], for more information on these methods. The default value for ABAQUS/Explicit is loop, which can be changed in the environment file (see ABAQUS/Standard User's Manual [24], Section 3.4.1, "Using the ABAQUS environment settings" and [26]).

For ABAQUS/Standard the values supernode and tree are applicable. The default value is supernode, which invokes the version of the parallel sparse solver that solves individual fronts in parallel. If **parallel**=tree, multiple fronts are solved simultaneously in addition to solving individual fronts in parallel.

domains

This option specifies the number of parallel domains in ABAQUS/Explicit. If the value is greater than 1, the domain decomposition will be performed regardless of the values of the **parallel** and **cpus** options. However, if **parallel**=domain, the value of **cpus** must be evenly divisible into the value of **domains**. The default value is 1, which can be changed in the environment file (see ABAQUS/Standard User's Manual [24], Section 3.4.1, "Using the ABAQUS environment settings" and [26]).

memory

This option specifies the number of 64-bit words of memory allocated by the solver input file processor. The default is sufficient for most problems. If an analysis requires more memory than is currently in use by the solver input file processor, it will issue an error message listing the current amount of memory in use. Incrementing the memory by 50% is the recommended approach to increasing memory. The default value for this parameter can be set with the pre_memory parameter in the environment file (see ABAQUS/Standard User's Manual [24], Section 3.4.1, "Using the ABAQUS environment settings" and [26]).

buffer

This option specifies the number of 64-bit words that are used for each internal scratch buffer in the solver input file processor. These scratch buffers are used for internal databases. The default is sufficient for most problems. Insufficient buffer space will cause the solver input file processor to use more external (disk) storage and will affect performance. The default value for this parameter can be set with the pre_buffer parameter in the environment file (see ABAQUS/Standard User's Manual [24], Section 3.4.1, "Using the ABAQUS environment settings" and [26]).

interactive

This option will cause the job to run interactively. For ABAQUS/Standard the log file will be output to the screen; for ABAQUS/Explicit the status file and the log file will be output to the screen. The default run_mode can be set in the environment file (see ABAQUS/Standard User's Manual [24], Section 3.4.1, "Using the ABAQUS environment settings" and [26]).

background

This option will submit the job to run in the background, which is the default. Log file output will be saved in the file *job-name.log* in the current directory. The default method for submitting the job can be set in the environment file by using the run_mode parameter (see ABAQUS/Standard User's Manual [24], Section 3.4.1, "Using the ABAQUS environment settings" and [26]).

queue

This option will submit the job to a batch queue. If the option appears with no value, the job will be submitted to the system default queue. Quoted strings are allowed. The available queues are site specific. Contact your site administrator to find out more about local queuing capabilities. Use **information**=local to see what local queuing capabilities have been installed. The default method for submitting the job can be set in the environment file by using the run_mode parameter (see ABAQUS/Standard User's Manual [24], Section 3.4.1, "Using the ABAQUS environment settings" and [26]).

after

This option is used in conjunction with the **queue** option to specify the time at which the job will start in the selected batch queue. This capability is supported for each individual site through the ABAQUS environment file. (See the ABAQUS Site Guide [28] for details.)

double

This option is used to specify that the double precision executable is to be used for ABAQUS/Explicit. This option is available only on machines where the default length of a single precision, floating point word is 32 bits. This option will run the executable for ABAQUS/Explicit that was built using double precision, floating point word lengths of 64 bits. This capability is also supported through the ABAQUS environment file with the environment variable explicit_precision (see ABAQUS/Standard User's Manual [24], Section 3.4.1, "Using the ABAQUS environment settings" and [26]).

Note that all servers in the CSAF are running 64 bits operating systems, hence this parameter is not required.

scratch

This option is used to specify the name of the directory used for scratch files. On UNIX platforms the default value is the value of the $TMPDIR environment variable or /tmp if $TMPDIR is not defined. On Windows platforms the default value is the value of the %TEMP% environment variable or \TEMP if this variable is not defined. During the analysis a subdirectory will be created under this directory to hold the analysis scratch files. The default value for this parameter can be set in the environment file (see ABAQUS/Standard User's Manual [24], Section 3.4.1, "Using the ABAQUS environment settings" and [26]). If the scratch files are too large for the scratch directory, auxiliary scratch directories can be specified using the aux_scratch parameter in the environment file. The scratch file will be split once split_scratch (also set in the environment file) is exceeded.

startup

This option is available only for ABAQUS/Design. It specifies the name of the file containing Python configuration commands to be run at application startup. Commands in this file are run after any configuration commands that have been set in the environment file.

noenvstartup

This option is available only for ABAQUS/Design. It specifies that all configuration commands in the environment files should not be run at application startup. This option can be used in conjunction with the **startup** command to suppress all configuration commands except for those in the startup file.

output_precision

This option specifies the precision of the nodal output written to the output database (job-name.odb) using the *NODE OUTPUT option. Using **output_precision**=full results in double precision output for ABAQUS/Standard analyses. To obtain double precision output for ABAQUS/Explicit analyses, use the double option in addition to using **output_precision**=full.

6.2.3 Checking the ABAQUS license – abaqus licensing

ABAQUS version 6.2-7 and later uses a version of the FLEXlm licensing software. (ABAQUS versions 6.2-6 and earlier used the ELM licensing software, which was very sensitive to incorrect job termination, i.e. the license manager could hang onto the licence tokens for up to an hour if a job crashed or was otherwise interrupted by the user.)

The status of ABAQUS jobs may be examined with the **abaqus licensing** utility.

Usage is:

abaqus licensing –ru

or

abaqus licensing lmstat –f ABAQUS

6.2.4 Checking ABAQUS license usage – seeaba

Displays all licensed abaqus features and their usage.

Usage is:

seeaba

6.3 PACSYS PAFEC

PAFEC is installed only on the two HP servers, bigted (the HP K260 server) and jayted (the HP J7000 server). PAFEC levels 6.2, 8.1 and 8.5 are installed on bigted and only PAFEC level 8.7 is installed on jayted. Note that level 6.2 has some compatibility issues with the current compiler and operating system version, which cause some analysis features not to function. These issues may well be resolved by a recompilation of the package, but this has been deferred until a sufficient need for this level and these features exists.

6.3.1 Running a PAFEC job – pafrun

This is the command script that sets up the PAFEC analysis run. It allows the user to run a PAFEC analysis interactively only.

Usage is:

pafrun [*job_name*] [*opt*] [*level*]

where:

job_name	base name of input data deck, i.e. without the .DAT suffix. For levels prior to 8.1, the name had to be in UPPERCASE.
opt	1=new analysis, 2=restart, 3=restart
level	PAFEC level to be run, e.g. 62, 81, 85 for levels 6.2, 8.1 and 8.5, respectively on bigted, and 87 only on jayted.

6.3.2 Releasing stale licenses before and after a PAFEC job – rmlock

This is a PAFEC utility program that allows the user to check for and remove stale PAFEC license lock files that were created by any user. It is deemed to be good practice to run it before and after a PAFEC analysis. In the first case, it can remove stale lock files left by any users' previous analysis that failed. In the second case, it ensures that the lock files are removed after your current analysis – irrespective of whether it failed or succeeded.

Usage is:

rmlock [–*y*]

where:

–*y*	remove all stale lock files without prompting

6.3.3 Cleaning up after a PAFEC job – pafdel

This is the CSAF custom command script developed to remove output files from, in order of precedence, PAFEC, ABAQUS and MSC.Nastran jobs on the server bigted. It is identical to fedel (see section 6.1.4), but its name is retained for compatibility with the older PAFEC analysis scripts developed to enhance the standard PAFEC **pafdel** command. It is important to note that when this command identifies a PAFEC analysis that the following additional files that match the base *job name* and the following suffix patterns are also removed:

[0-1][0-9], [0-1][0-9].[f], [0-1][0-9].exe, _ERROR, _err

Usage is:

pafdel [*job name*]

6.3.4 Checking and setting the PAFEC level to run – paflev

This is the command to check and set the global and local levels of PAFEC to run. The global level is defined in the user's home directory in the file ~/.pafeclevel. A different level can be specified for a directory by using the local switch option. This option creates a .pafeclevel file in the current directory. When a PAFEC analysis is run it first checks for this local file and if found uses the PAFEC level specified therein to run the analysis, otherwise it looks for the global definition file in the user's home directory.

Usage is:

paflev [*level*] [**global** | **local**]

where:

level	PAFEC level to be run, e.g. 62, 81, 85 for levels 6.2, 8.1 and 8.5, respectively
global	Set global default run level for all your analyses
local	Set default run level for the current directory (overrides global default)

7. CSAF FEA Pre/Post Processing Applications

This section details the specific commands to use when pre- and post-processing data for the various FEA applications on the CSAF. The commands are separated into standard vendor components and customised CSAF commands. Table 7 shows the matrix of vendor commands and Table 8 shows the custom commands available on each server. Users are expected to be familiar with these commands. The Section column in each table refers to the relevant section in this text in which the detailed descriptions and syntax for use of these commands is given. Relevant references are listed in sections 9.1, 9.2, 9.3, 9.4, 9.5, 9.6 and 9.7.

Table 7. Vendor application command matrix

Application	Command	Name of Server					Section
		jayted	bigted	farted	poisson2	ardvark	
ABAQUS	abaqus viewer	x			x		7.2.1
ABAQUS	abaqus doc	x			x		7.2.3
ABAQUS	abaqus convertenv	x			x		7.2.4
ABAQUS	abaqus upgrade	x			x		7.2.5
MSC.Patran	patran	x	[x]	x		x	7.1.1
PAFEC	pigs		x				7.3.2
PAFEC	puppies		x				7.3.1
PAFEC	plotq		x				7.3.3
tecplot		x					7.4.1
slim						x	7.6.1

Table 8. CSAF custom application command matrix

Command	Name of Server					Section
	jayted	bigted	farted	poisson2	ardvark	
seepat	x		x			7.1.2
fedel	x	[x]		x		6.1.4
pafdel		x				6.3.2

7.1 MSC.Patran

7.1.1 Starting the MSC.Patran gui

There are several versions of MSC.Patran installed on the different servers. Table 9 shows the versions installed on each machine and also shows the default for that machine.

Table 9 MSC.Patran versions available on each server. (=default)*

	jayted	bigted	farted	ardvark
p85	x		*	
p90	x			
p95/p2000	x			x
p2001	*			*

To select the default version on any machine, the relevant commands are:

patran
pat
p3

To select a specific version on any machine, the relevant commands are:

p2001
p2000
p95
p90
p85

Usage is:

patran [*journal_file*]

where:

journal_file is a file containing a list of PCL commands to be performed on start up of the MSC.Patran session.

7.1.2 Displaying MSC.Patran license usage – seepat

This is the CSAF custom command script developed to display current MSC.Patran license use on the server jayted. The command has no parameters. Be warned: occasionally license tokens are returned prematurely to FLEXlm from some MSC.Patran sessions, which leaves the sessions running in an indeterminate state and also then permits new sessions to be started – even if the number of sessions exceeds the license limit. When this event occurs the "lost" session still continues but may have difficulty as it may have to wait until another token becomes available. It is recommended that a more thorough check be made on the state of your MSC.Patran session by using the command **hey** *p3* or **hey** *msc* (see section 7.7.4).

Usage is:

seepat

7.2 ABAQUS

The following is a summary of the some of the more relevant pre- and post-processing options extracted from the output of the command: **abaqus help**

7.2.1 Viewing results of abaqus jobs – CAE

The new ABAQUS tool to process and view results of an analysis is called CAE, which has replaced the viewer tool (see section 7.2.2). Many results may be viewed by using MSC.Patran also.

Usage is:

abaqus cae [**database**=*database-file*] [**replay**=*replay-file*] [**recover**=*journal-file*] [**startup**=*startup-file*] [**noenvstartup**]

where:

database=*database-file*	This option specifies the name of the model database file or output database file to open. To specify a model database file, include either the .cae file extension or no file extension in the file name. To specify an output database file, include the .odb file extension in the file name.
replay=*replay-file*	This option specifies the name of the file from which ABAQUS/CAE commands are to be replayed. The commands in replay-file will execute immediately upon startup of ABAQUS/CAE. If no file extension is given, the default extension is .rpy.
recover=*journal-file*	This option specifies the name of the file from which a model database is to be rebuilt. The commands in journal-file will execute immediately upon startup of ABAQUS/CAE. If no file extension is given, the default extension is .jnl.
startup=*startup-file*	This option specifies the name of the file containing Python configuration commands to be run at application startup.

Commands in this file are run after any configuration commands that have been set in the environment file.

noenvstartup This option specifies that all configuration commands in the environment files should not be run at application startup. This option can be used in conjunction with the **startup** command to suppress all configuration commands except for those in the startup file.

7.2.2 Viewing results of abaqus jobs – viewer

The old tool to process and view results is called viewer. It has the same features as the CAE tool, with the exception that it cannot create new databases. (It will be discontinued in future ABAQUS releases.)

Usage is:

abaqus viewer [**database**=*database-file*] [**replay**=*replay-file*] [**startup**=*startup-file*] [**noenvstartup**]

where:

database=*database-file* This option specifies the database from an analysis to load into the viewer.

replay=*replay-file* This option specifies a session file saved from a previous viewer session to be replayed when viewer starts up.

startup=startup-file This option is available only for ABAQUS/Design. It specifies the name of the file containing Python configuration commands that are to be run at application startup. Commands in this file are run after any configuration commands that have been set in the environment file.

noenvstartup This option is available only for ABAQUS/Design. It specifies that all configuration commands in the environment files should not be run at application startup. This option can be used in conjunction with the **startup** command to suppress all configuration commands except for those in the startup file.

7.2.3 abaqus doc

This command accesses the on-line documentation for ABAQUS. The documentation is in a Hyper-Help format and therefore requires an X-display to view.

Usage is:

abaqus doc

7.2.4 abaqus convertenv

This is the procedure for converting the environment file from a version 5 file to version 6. By default it converts an old abaqus.env file and produces a new abaqus_v6.env file.

abaqus convertenv [*environment-file*]

where:

convertenv	ABAQUS keyword to invoke the env convert function
environment-file	alternate ABAQUS env file to convert

7.2.5 abaqus upgrade

This is the procedure to upgrade an old ABAQUS input file to version 6.2.

abaqus upgrade job=*new-file* **input**=*old-file*

where:

upgrade	ABAQUS keyword to invoke the env convert function
job=*new-file*	name of the new job after upgrading
input=*old-file*	name of the old file to be upgraded

7.3 PACSYS PAFEC

7.3.1 Processing output of PAFEC jobs – puppies

This is a utility, developed by Engineering Analysis Software, to extract data from a PAFEC BS, SS, EE or ET file after a PAFEC job run. **puppies** stands for Pafec User Pre and Post program for the Interactive Evaluation of Structures. There are three versions on bigted, these being 1.93, 8.1 and 8.5. Versions 1.93 and 8.1 are compatible with PAFEC level 8.1 Backing Store (BS) files and version 8.5 for PAFEC 8.5. The commands to access

each version are **pup193**, **pup81** and **pup85** respectively. The default version is also available with the command **pup**. Presently this is linked to the puppies 8.1 version and the version 8.5 is available through a customised executable called **puppies.exe**, which is distributed to users as required.

Usage is:

puppies [*journal_name*]

or

pup [j*ournal_ name*]

where:

journal_name the name of a puppies journal file.

7.3.2 PAFEC Interactive Graphics System – pigs

PIGS is the interactive graphics system to process PAFEC jobs. It can be used to build new models, create new data decks or modules and to process analysis results. It is mostly used for post processing stress, strain, displacement or thermal distribution data after an analysis.

Usage is:

pigs [*%window size*]

7.3.3 Printing plots from pigs – plotq

This is the CSAF custom command script developed to print DOGS plot files generated in **pigs** from PAFEC jobs on the server bigted. Note that the PAFEC plot queues have not been enabled on bigted at present.

Usage is:

plotq [*plot_file*] [*pafec_print_queue*]

where:

plot_file a DOGS plot file created by **pigs**
pafec_print_queue a PAFEC print queue

7.4 Amtec Tecplot

7.4.1 Scientific plotting of data – tecplot

This is a fully featured scientific plotting package. It has many capabilities and features. Tecplot creates and uses either a layout file, *.lay, or a package file, *.lpk. As Tecplot has so many features for a novice, the best option here is to find someone to show you how to use it and then you must be willing to spend some time reading the manual. Tecplot can export plots in Windows compatible formats, such as Windows metafile and bitmaps.

Usage is:

tecplot [*plot_layout_file*]

where:

plot_layout_file may be either a layout file, *job*.lay, or a package file, *job*.lpk

7.5 MSC.Nastran

7.5.1 Creating and printing plots from MSC.Nastran jobs – plotps

This is the MSC.Nastran command to create a PostScript output file from a plot file created during an MSC.Nastran analysis by the XYOUTPUT command section option.

Usage is:

plotps (*input-plot-file*) **output**=[*output-ps-file*] [**begin**=*first-frame-to-plot*] [**end**=*last-frame-to-plot*]

where for **plotps** version 2001.0:

Command line parameters	Default
input-plot-file	(none).plt
begin=*first-frame-to-plot*	1
color={yes | no}	no
cscale=*character-scale-factor*	1.0
dump={yes | no}	no
end=*last-frame-to-plot*	999999
format={binary | neutral}	binary
height=*page-height*	10.0

optimizestrings={yes | no}	yes
output= *output-ps-file*	(none).ps
rotate={automatic | no | yes}	automatic
scale=*plot-scale-factor*	1.0
verbose={yes | no}	yes
width=*page-width*	7.5

The frames output by **plotps** can then be printed with the **lp** command (see section 5.4.6 and Appendix A.2), e.g.: **lp** *outfile*.ps

7.6 TMP SLIM and Vision

7.6.1 Specialised processing of large MSC.Nastran jobs – slim

The **S**tructural **L**oads **I**nterface **M**ethod (SLIM) was developed at General Dynamics, Fort Worth, Texas (July 1992) for the F-22 stress analysis community to allow easy access to internal loads data by individual analysts. The software is an extension of the "NASUU3" software used by the F-16 stress analysis community. Now sold by Third Millennium Productions (TMP), SLIM provides the capability to isolate parts (i.e. select a group of elements/nodes), perform maximum/minimum internal load scans (i.e. stress, strain, force, etc.), summarise results, and generate TMP Vision results files for visualisation of finite element model results.

The software resides on the SGI Octane workstation *ardvark*. SLIM can be initiated by executing the following UNIX command. Note that the user manual [40] is in html format and hence requires a browser to view it.

Usage is:

slim [*mode*] (*dbname*) [(*partname*)]

where:

mode	Operational mode (Default= '–i').	
	–i	Interactive SLIM (command line prompts).
	–porgen	Automated batch portable file generation.
	–viewgen	Automated batch TMP Vision[tm] results file generation.
	–u	Display command line usage.
	–h	Display on-line user documentation (html).
	–r	Software restrictions and limitations.
	–c	Copy custom SLIM template to specified file.
dbname	Name of SLIM data base or condition definition file.	
partfile	Name of part file defining the portion of the model for recovery (valid only for '–porgen' and '–viewgen' modes).	

7.6.2 Specialised processing of large MSC.Nastran jobs – vision

TMP Vision is a visualisation tool designed to visualise data that can be mapped on to a polygonal model. Although a number of features are aimed at visualising finite element models and associated analysis results, any data that can be mapped onto a mesh of polygons can be viewed. TMP Vision is designed using Motif as the graphical user interface. All graphics are drawn in the graphics region using either Iris GL or Open GL depending on the hardware being used.

The software resides on the SGI Octane workstation *ardvark*. Vision can be initiated by executing the following UNIX command. Note that the user manual [41] is in html format and hence requires a browser to view it.

Usage is:

vision

7.7 CSAF utilities

7.7.1 Text editors – dtpad or vuepad

These simple X-windows text editors are available on the HP servers under the CDE or VUE desktop managers, respectively. More specifically, the dtpad utility is a basic editor that supports editing text files in a manner consistent with other common graphical user interface text manipulation and file access mechanisms. Cursor positioning and text selection as well as access to various editing operations can be done via the standard Motif text manipulation mechanisms using the mouse or user-definable key combinations. Text can be cut, copied or pasted, or dragged to and from the Text Editor and/or other compliant application windows. Also, standard dialogs are presented for accessing files and printing text.

The Text Editor also provides the following features:

- Pull down menus for common edit and file operations.
- Undo of the previous edit operation.
- Search and replace.
- Spell checking.
- Simple formatting.
- Wrap-to-fit and overstrike modes.
- Optional status line, allowing cursor positioning by line number.
- Automatic file save on many abnormal termination conditions.
- Mechanism for automatic session save and restore.

It is usually invoked by CDE, but a user may directly start it.

Usage is:

dtpad [*options*] [*file_name*]

Useful option: –standAlone

For example, to launch a new instance of the editor and run it as a background process in the current terminal:

dtpad –standAlone &

7.7.2 Text editor – xemacs

XEmacs is a fully featured editor based on the GNU Emacs package. It has a very large set of features and capabilities, including recording and playback of keystroke macros, as well as its own very extensive built-in help system. A detailed description of its use is beyond the scope of this text. The alias **x** for **xemacs** is pre-defined in the user's .shrc file.

Usage is:

xemacs [*file_name*] [*file_name2*]
or
x [*file_name*] [*file_name2*]

7.7.3 beep

This is the CSAF custom command script developed to ring the bell in a terminal window on the servers bigted and jayted and the workstations farted and storm. It is often used after a string of other commands to give an audible indication when the previous commands have completed.

Usage is:

beep

Example: Tell me by ringing the (terminal) bell, when the system has finished copying the entire contents of a huge directory from the current directory to my /tmpwrk area:

cp –r *VeryBigDirectory /tmpwrk/myuser***; beep**

7.7.4 hey

This is the CSAF custom command script developed to quickly find from the list of processes running on the system which, if any, contain the specified pattern.

Usage is:

hey (*pattern*)

7.7.5 well

This is the CSAF custom command script developed to quickly find, from the list of processes running on the system, those that contain the user's login name.

Usage is:

well

7.7.6 Image capture and manipulation – xv

An image viewer and editor. It is an old X-based application but it is still very useful. It supports many image formats, the most commonly used being TIF, GIF and JPEG.

Usage is:

xv

7.7.7 Image manipulation – imageview

Image viewer and a capture utility. This HP supplied utility has a limited editing ability and supports a limited range of image formats.

Usage is:

imageview

7.7.8 Image manipulation – capture

The **capture** utility works as part of the **imageview** utility, although it may be started independently.

Usage is:

capture

7.7.9 PostScript file viewing and printing – ghostview

Ghostview is an X11 user interface for ghostscript (see section 7.7.10) written by Timothy O. Theisen and provided under the GNU public license. It is a utility to view and print PostScript files.

Usage is:

ghostview [*postscript_file*]

7.7.10 PostScript file editing – ghostscipt

It is a utility written by Aladdin Enterprises, provided under the GNU public license, to edit and print PostScript files.

Usage is:

gs

7.7.11 Web browser – netscape

The web browser supplied with HP computer systems running HP-UX.

Usage is:

netscape [*URL*]

7.7.12 HP documentation viewer – dynatext

It provides on-line help system and programming references and documentation, and is the central part of the HP Instant Information software.

Usage is:

dynatext [*book*]

7.7.13 MSC.Nastran documentation viewer – iview

This provides access to the MSC.Nastran Online Encyclopedia. It covers MSC.Nastran version 70.5, but it has not been updated by MSC because of Year 2000 (Y2K) issues.

Usage is:

iview [*book*]

8. Example Operations

8.1 Copying files to/from another user

This operation requires that both of the users that are involved set the correct security permissions on their files and directories. The particular combination that is required depends upon the group membership of each user. The first two methods listed below are "quick 'n' dirty" ways to perform this operation. The first method provides neither data security, nor any guarantee of data integrity. The second method, whilst providing data integrity, has no data security. The third method has both data security and integrity, but does require careful setting up and continual awareness by both users as to the current group that they are working in.

Method 1: (Not Recommended!)
User A copies the file(s) to /var/tmp
User B copies the file(s) from /var/tmp
User A deletes file(s) in /var/tmp

Method 2: (Not Recommended!)
User A sets global read permissions on the files to be copied and read, and execute permissions on the directory containing the files, then tells User B the path and file names. Note that ANY user can now read the files.
User B copies the files from User A's area.
User A removes the global read access.

Method 3: **(Preferred under the current Defence Security context)**
User A and B check to see if they belong to the same user group. If not, then they request the system administrator to arrange a common group.
If necessary, User A changes to the common group and changes the group that the files and directories belong to.
User A sets the group read (and execute) permissions as appropriate.
User B copies the files.

8.2 Transferring data between Unix servers

Remote copying is done using the **rcp** command, which requires that a properly configured .rhosts file exists on both servers. The .rhosts file must list the remote user name and host name to be permitted access to the local account. For an illustration of this procedure, see the examples provided in Section 5.4.16.

8.3 Transferring data to a PC

It is not possible to directly transfer data from a Unix server to a PC in the vast majority of cases because Windows 95 or 98 are not capable of running an ftp server to accept the connection from the Unix server. (Windows NT, Windows 2000 or Windoze XP can run an ftp server – but they require it to be specifically installed and configured.) The only way most Windows users can transfer data is by using an ftp program from their PC. It is outside the scope of this text to cover the multitude of variations of ftp programs available. Help with setting up a Windows-based ftp program should be sought from your nearest SCIS Computing Help Desk.

8.4 Recommended generic modelling procedures

The following sections will outline recommended generic procedures that may be used to perform FEA using MSC.Nastran, ABAQUS or PAFEC on the CSAF. The aim of these is to give users a general guide as to how FEA should be carried out, to meet CSAF and NAFEMS guidelines, without going into all of the intricate details. The set of common steps follows. After that, specific procedures are given in separate sections for each analysis code, MSC.Nastran in section 8.4.1, ABAQUS in section 8.4.2 and PAFEC in section 8.4.3, analysis code.

The common first steps in the procedure are:

a) Assemble all of the information required, such as the geometry, materials, loads and environmental conditions to be modelled.
b) Define the type and extent of the model, e.g. a 2-D linear static analysis of half of the structure, using elements with mid-side nodes to enable bending stresses to be more accurately analysed.
c) Take the time to verify the boundary conditions. A quick sketch of the model here can save you many hours of post-processing of results trying to calibrate and/or validate the analysis!
d) Select the FEA code to be used. All three FEA codes can perform a wide range of analyses quite well. However, a very loose guide for beginners is to use MSC.Nastran for static analysis and ABAQUS for dynamic analysis, while PAFEC is generally recommended only for users with previous experience with the

application. Note that this choice may also be dictated by external factors such as client requests or compatibility with other work.

e) Create a directory to contain your new analysis using **mkdir**.
f) Create a plain text file in this new directory called *README* to contain a brief description of the work in this directory and give the name of your model index file in this directory.
g) Create a second plain text file, the model index file, which is to contain an index listing of all your databases and analysis decks in this directory, with a short description of each. This file would also be a good place to define any conventions that you have used, such as file names, units, model parameters, etc. An example name is *2D_structure.index.*

8.4.1 MSC.Nastran

These are the generic modelling steps to carry out an MSC.Nastran analysis after performing the common steps in section 8.4 above.

a) Open **patran** and create a new database with the analysis code set to "MSC.Nastran", the analysis types set to "Structural" or "Thermal" as appropriate and the tolerance set to "default".
b) Create the model geometry.
c) Create the finite elements.
d) Create the restraints and loads.
e) Create the materials.
f) Assign the material properties.
g) Optionally define multiple load cases.
h) Create an analysis deck with the appropriate load cases for the desired solution type. Note: CSAF policy is to NEVER run an analysis from within **patran.**
i) Close **patran.**
j) Open the data deck with your preferred editor, e.g. **dtpad**, to verify that the deck has the correct solution type and contains all of the load cases and components of the model you want analysed.
k) Either reopen **patran** to fix any problems and rewrite the data deck *(preferred method)* or directly edit the deck, which is necessary in some cases as MSC.Patran does not support all MSC.Nastran solution types *(but this also means that the patran database will no longer match the data deck)*.
l) For larger MSC.Nastran analyses, run the **estimate** utility to tune memory, buffer size and other **nastran** command parameters for this analysis.
m) If necessary, copy the data deck and any bulk data include files to the remote host, i.e. poisson2.
n) Check (and set) with the **mesg** command, that messages are allowed to be written to the terminal window in which you submit the job.
o) Submit the analysis with the **nastran** command (or the **nastbat** command on poisson2).

p) When the job is completed, check the log, f04 and f06 files for errors and warnings and in particular make sure that the job has completed.
q) Begin post-processing the results – with the naked eye, or with **tecplot** or with **patran** or using any programs you have written or been supplied with for that purpose.
r) Clean up unnecessary files, e.g. log, f0?, xdb, or op2 files, from the analysis either manually or using the **fedel** command, and the *patran.session.** files from the **patran** session.
s) If required, create a new load case or refine the FE mesh in your MSC.Patran database, or create a new database for the next iteration of the analysis.
t) Remember to update the info in your model index file if you have not already done so.
u) If you are satisfied with your results, now write your report.
v) After the report has been done ensure that the essential data is archived.
w) Clean up the directory.

8.4.2 ABAQUS

These are the generic modelling steps to carry out an ABAQUS analysis after performing the common steps in section 8.4 above.

a) Open **patran** and create a new database with the analysis code set to "ABAQUS", the analysis types set to "Structural" or "Thermal" as appropriate and the tolerance set to "default".
b) Create the model geometry.
c) Create the finite elements.
d) Create the restraints and loads.
e) Create the materials.
f) Assign the material properties.
g) Optionally define multiple load cases.
h) Create an analysis deck with the appropriate load cases for the desired solution type. Note: CSAF policy is to NEVER run an analysis from within **patran.**
i) Close **patran.**
j) Open the data deck with your preferred editor, e.g. **dtpad**, to verify that the deck has the correct solution type and contains all of the load cases and components of the model you want analysed.
k) Either reopen **patran** to fix any problems and rewrite the data deck *(preferred method)* or directly edit the deck, which is necessary in some cases as MSC.Patran does not support all ABAQUS solution types *(but this also means that the patran database will no longer match the data deck)*.
l) For larger ABAQUS analyses, run the **abaqus job=***job_name* **info=memory** command to tune memory and buffer size and other **abaqus** command parameters for this analysis.

m) If necessary, copy the data deck and any data include files to the remote host, i.e. poisson2.
n) Check (and set) with the **mesg** command, that messages are allowed to be written to the terminal window in which you submit the job.
o) Submit the analysis with the **abaqus** command.
p) When the job is completed, check the log, out files for errors and warnings and in particular make sure that the job has completed.
q) Begin post-processing the results – with the naked eye, or with **tecplot** or with **abaqus viewer** or with **patran** or using any programs you have written or been supplied with for that purpose.
r) Clean up unnecessary files, e.g. log, out, fil, or odb files, from the analysis either manually or using the **fedel** command, and the *patran.session.** files from the **patran** session.
s) If required, create a new load case or refine the FE mesh in your MSC.Patran database, or create a new database for the next iteration of the analysis.
t) Remember to update the info in your model index file if you have not already done so.
u) If you are satisfied with your results, now write your report.
v) After the report has been done ensure that the essential data is archived.
w) Clean up the directory.

8.4.3 PAFEC

A PAFEC data deck needs to include the following modules or sections:

- A control module.
- PAFBLOCKS, finite elements and nodes.
- The restraints and loads.
- The materials.
- And optionally, multiple load cases.

These are the generic modelling steps to carry out a PAFEC analysis after performing the common steps in section 8.4 above.

a) Either:

(i) Open your favourite editor, e.g. **dtpad**, and create a new data deck.
(ii) Open **pigs** and create a new structure and save the new data deck *(definitely not recommended)*. When using **pigs**, also note that the **rmlock** utility should be employed before and after to ensure that any old PAFEC lock files are cleaned up.
(iii) Create or use a program that writes the complete data deck, or parts of it *(highly recommended for any parametric modelling studies)*.

b) Open the data deck with your preferred editor, e.g. **dtpad**, to verify that the deck has the correct solution type and contains all of the load cases and components of the model you want analysed.
c) Check (and set) with the **mesg** command, that messages are allowed to be written to the terminal window in which you submit the job.
d) Use the **rmlock** utility to clean up any old PAFEC lock files.
e) Submit the analysis with the **pafrun** command.
f) Use the **rmlock** utility to clean up any PAFEC lock files left if your job crashed.
g) When the job is completed, check the log, O0? and MON files for errors and warnings and in particular make sure that the job has completed.
h) Begin post-processing the results – with the naked eye, or with **puppies** or with **pigs** or using any programs you have written or been supplied with for that purpose.
i) Clean up unnecessary files, e.g. log, O0?, BS, or SS files, from the analysis either manually or using the **pafdel** command.
j) If required, create a new load case or refine the FE mesh in your data deck, or create a new data deck for the next iteration of the analysis.
k) Remember to update the info in your model index file if you have not already done so.
l) If you are satisfied with your results, now write your report.
m) After the report has been done ensure that the essential data is archived.
n) Clean up the directory.

9. Overview of Key Manuals and References

The following manuals are available either in the CSAF Workstation Room, or as on-line documentation. Each manual is listed with a brief description of its purpose and contents. The manuals have been grouped by application and have generally been subdivided into User manual/guide and Reference sections.

9.1 MSC.Nastran

The latest information and answers to frequently asked questions (FAQs) may be obtained on the internet from the MSC Software or MSC Software Australia home pages at: http://www.mscsoftware.com or http://www.mscsoftware.com.au, respectively.

Additional help may be found if you go to the MSC Mechanical Solutions website at www.mechsolutions.com and click on **Support**. Here, you can find a wide variety of support resources including examples of applications, technical application notes, available training courses and documentation updates at the MSC.Software Training, Technical Support, and Documentation web page.

9.1.1 Installation and release guides

a) *MSC.Nastran 2001 Release Guide* [1]. This book contains an overview of MSC.Nastran 2001, including enhancements to Aeroelasticity and Parallel Numeric features as well as Miscellaneous enhancements. Also covers Design Optimisation, Dynamics, Elements and Loads, Nonlinear Analysis, Model Checkout and Evaluation Tools, and Upward compatibility of features. (10 chapters, 3 appendices, 500+ pages)

b) *MSC.Nastran 2001 Installation and Operations Guide* [1]. This manual provides instructions on how to install, customise, and use MSC.Nastran Version 2001 on UNIX and Windows systems. This document assumes that you have a working knowledge of the applicable operating environments. The structure of the document is: chapter 2) Installing, 3) Configuring, 4) Using the Basic Functions, 5) Using the Advanced Functions, 6) Using Utility Programs, 7) Building and Using Sample Programs, and Appendices A) Glossary, B) Keywords and Environment Variables, C) System Descriptions and D) Product Timing Data

9.1.2 Reference books

a) *MSC.Nastran Theory Reference (v68)* [3]. This 3-volume set is still the current MSC reference for MSC.Nastran. It covers all the features and solution types that can be undertaken by MSC.Nastran.

b) *MSC.Nastran Quick Reference Guide (version 2001)* [4]. Over 1500 pages divided into 6 chapters with sections covering: 1) the **nastran** command and NASTRAN statement, 2) file management statements, 3) executive control statements, 4) case control commands, 5) bulk data entries and 6) command and resource parameters. Also includes appendices with Item codes and Degree-of-Freedom Sets.

c) *MSC.Nastran DMAP Programmer's Guide (version 2001)* [5]. Explains what DMAP functions are and how to use them with various MSC.Nastran analyses. It is intended for more experienced users.

d) *MSC.Nastran Reference Manual (version 2001)* [6]. The MSC reference for MSC.Nastran provides a full description of all MSC.Nastran commands and statements and parameters.

9.1.3 User's guides

a) *Getting Started (version 2001)* [7]. Guide on how to begin an analysis and the recommended steps required to carry out an analysis with MSC.Nastran.

b) *Linear Static Analysis (version 2001)* [8]. Guide on how to perform linear static analysis with MSC.Nastran.

c) *Basic Dynamic Analysis (version 2001)* [9]. Guide on how to perform basic dynamic analysis with MSC.Nastran.

d) *Advanced Dynamic Analysis (version 2001)* [10]. Guide on how to perform advanced dynamic analysis with MSC.Nastran.

e) *Design Sensitivity and Optimisation (version 2001)* [11]. Guide on how to perform design sensitivity and optimisation analysis with MSC.Nastran.

f) *Thermal analysis (version 2001)* [12]. Guide on how to perform thermal analysis with MSC.Nastran.

g) *Numerical Methods (version 2001)* [13]. Guide on the numerical methods available for use in various analyses with MSC.Nastran.

h) *Aeroelastic Analysis (version 2001)* [14]. Guide on how to perform aeroelastic analysis with MSC.Nastran.

i) *User Modifiable (version 2001)* [15]. Guide on using user routines to modify MSC.Nastran behaviour during various analyses.

j) *Toolkit (version 2001)* [16]. Guide to the MSC.Nastran toolkit.

k) *MSC.Nastran On-line Encyclopedia (v70.5)* [17]. This collection is the last set of on-line documentation supplied by MSC and is now very much out of date with respect to the later release features. However, it does contain the complete MSC.Nastran Reference Manual v68, which is still the current MSC.Nastran reference document. It is still of some use in looking up MSC.Nastran command and data deck options as the majority of the MSC.Nastran package features have not changed. It is also useful for looking up error messages, as many of these tend to be the ones that still occur with the newer versions.

9.2 MSC.Patran

a) *MSC.Patran 2001 Installation and Operations Guide* [18]. This book is intended to guide system administrators and users through the installation of MSC.Patran. It contains all the information you need to install and configure MSC.Patran. It is organised into the following major sections: Chapter 2: Required Hardware and Software Configurations; Chapter 3: Installing MSC.Patran on UNIX and LINUX; Chapter 4: Installing on Microsoft Windows NT; Chapter 5: User Environment; Chapter 6: Problems and Resolutions; Appendix A: Installing FLEXlm; and Appendix B: Testing the Installation.

b) *MSC.Patran Users Reference (versions 9.0 and 2001)* [19]. These online documents, accessible via their respective MSC.Patran Viewer applications, contain information on almost every feature of MSC.Patran.

c) *MSC.Patran PCL Programming Reference (versions 9.0 and 2001)* [20]. These online documents, accessible via their respective MSC.Patran Viewer applications, contain information on almost every feature of PCL (MSC.Patran Command Language).

d) *MSC.Patran Online Documentation (all versions)* [21]. This online documentation, accessible via the respective MSC.Patran Viewer applications, contains information on almost every feature of MSC.Patran including installation, operations, user references and PCL.

9.3 ABAQUS

The latest information and answers to FAQs may be obtained on the internet from the ABAQUS home page at: http://www.abaqus.com.

9.3.1 Training manuals

a) *Getting Started with ABAQUS/Standard (Interactive Version) (v6.2)* [22]. This document is a self-paced tutorial designed to help new users become familiar with using ABAQUS/CAE to create solid, shell and framework models and ABAQUS/Standard to perform static and dynamic stress analysis simulations. It contains a number of fully worked examples that provide practical guidelines for performing structural analyses with ABAQUS.

b) *Getting Started with ABAQUS/Standard (Keywords version)* [23]. This online-only document is designed to help new users become familiar with the ABAQUS/Standard input file syntax for static and dynamic stress analysis simulations. The ABAQUS/Standard keyword interface is used to model the same examples as those included in a) above.

9.3.2 User's manuals

a) *ABAQUS/Standard User's Manual – Volumes I, II and III (v6.2)* [24]. This is the basic reference document for ABAQUS/Standard. It contains a complete description of the elements, material models, procedures, input specifications, etc.

b) *ABAQUS/Viewer User's Manual (v6.2)* [25]. This basic reference document contains a tutorial as well as a complete description of how to use ABAQUS/Viewer to display your model and results.

c) *ABAQUS Online Documentation* [26]. Online manual containing instructions on using the ABAQUS online documentation server to read the manuals that are available online.

d) *ABAQUS Release Notes (v6.2)* [27]. Contains brief descriptions of the new features available in this release.

e) *ABAQUS Site Guide (v6.2)* [28]. Describes how to install and configure ABAQUS. Parts of this information of relevance to users are also provided in the user's manuals.

9.3.3 Example manuals

a) *ABAQUS Example Problems – Volumes I and II (v6.2)* [29]. Contains more than 75 detailed examples designed to illustrate the approaches and decisions needed to perform meaningful linear and non-linear analysis. It is generally useful to look for relevant examples in this manual and to review them when embarking on a new class of problem.

b) *ABAQUS Benchmarks Manual* [*30*]. Online only document, containing over 200 benchmark problems and standard analyses used to evaluate the performance of ABAQUS; tests are multiple element tests of simple geometries or simplified versions of real problems. Includes NAFEMS benchmark problems.

c) *ABAQUS Verification Manual* [*31*]. Online only volume, containing more than 5000 basic test cases providing verification of each program feature (procedures, output options, MPCs, etc.) against exact calculations and other published results. It may be useful to run these problems when learning to use a new capability. In addition, the supplied input data files provide good starting points to check the behaviour of elements, materials, etc.

9.3.4 Reference manuals

a) *ABAQUS Keywords Manual (v6.2)* [*32*]. Contains a complete description of all the input options that are available in ABAQUS/Standard and ABAQUS/Explicit.

b) *ABAQUS Theory Manual* [*33*]. Online only volume, contains detailed, precise discussions of all theoretical aspects of ABAQUS. It is written for readers with an engineering background.

c) *ABAQUS Input Files* [*34*]. Online manual, contains all the input files that are included with the ABAQUS release and referred to in the ABAQUS Example Problems Manual, the ABAQUS Benchmarks Manual and the ABAQUS Verification Manual. They are listed in the order in which they appear in the manuals, under the title of the problem that refers to them. The input file references in the manuals hyperlink directly to this book.

9.4 PACSYS PAFEC

9.4.1 User's manuals

a) *PAFEC Data Preparation Manual – Parts 1–3 (Level 8)* [35]. Contains 9 sections covering: 1) General description of data; 2) Data modules; 3) CONTROL module; 4) Element types; 5) System of units; 6) Data validation and element geometry; 7) Types of analysis; 8) Standard material properties; 9) Examples. These are a must read for all PAFEC users.

b) *PAFEC Interactive Graphics Reference Manual – Parts 1–3 (Level 8)* [36]. Contains 10 sections in over 870 pages covering the various ways to use PIGS, including examples.

c) *PAFEC PIGS Option Selector (Level 8)* [37]. The absolutely essential quick reference card to have at hand when using PIGS.

9.4.2 Reference manuals

a) *PAFEC Theory Manual* [38]. A brief history of PAFEC FE, the notation used and an extensive section on the theory used in the development of PAFEC FE and a very long list references.

9.5 Amtec Tecplot

a) *Amtec Tecplot User's Manual Version 10.0* [39]. Detailed descriptions of Tecplot capabilities covered in 32 chapters, 6 appendices and over 600 pages.

9.6 NAFEMS references

The National Association of Finite Element Method Standards (NAFEMS), UK, publishes standards for the conduct of FEA. The NAFEMS membership pack contains a set of guidelines and procedures to ensure the quality assurance of FE analysis. They include standard benchmarks and examples of various analyses such as linear, non-linear, dynamic, thermal, acoustic analysis. The modelling of composite materials, creep, fatigue and plasticity are also covered. Fracture mechanics in 2-D and 3-D are also covered.

Note that the 1998 report on quality assurance for finite element work by Callinan and Sanderson [92] identified deficiencies in some published NAFEMS standards and benchmarks at that time. (As an aside, it also demonstrated the reason why NAFEMS continually review and update their publications.) This highlights the benefit of, and even the need to, check the consistency and accuracy of your work against multiple sources of reference.

The NAFEMS publications are categorised under the following headings:

9.6.1 The 3-year pack textbooks

The topics in this section cover primers to FEA [42] and FE dynamics [43], an introduction to non-linear FEA [44] and a background to benchmarks [45]. Recent additions include an explicit primer [46] and FE based fatigue calculations [47].

9.6.2 Why do / how to series

The "why do" publications in this section are: "Why do FEA?" [48], "Why do non-linear FEA?" [49], "Why do design optimisation?" [50] and "Why do computational fluid dynamics?" [51].

The "how to" publications include topics such as how to: choose an FE System [52], get started with FE [53], get started in acoustics analysis [54], choose a pre/post processor [55], plan an FE analysis [56], interpret FE results [57], understand FE jargon [58], buy FE Services [59], undertake fracture mechanics analysis [60], tackle non-linear FEA [61], analyse composites [62], undertake FE based geotechnical analysis [63], understand CFD jargon [64], use elements effectively [65], and plan a CFD analysis [66].

9.6.3 Benchmarks and reports

The topics related to benchmarks in this section cover: non-linear FEA [67], standard NAFEMS benchmarks [71], FE pre-processors [72], forced vibration [73], tests for creep [74], material non-linearity [75], composite benchmarks [76, 77], and acoustic radiation and scattering [78].

The other topics in this section cover: NAFEMS workbooks of examples [68, 69], and the International Journal of CFD Case Studies (Vol. 3, July 2001) [70].

9.6.4 Management guides

The topics in this section cover quality (to ISO 9001) and management of FEA. The titles are: "Quality System Supplement to ISO 9001 relating to Finite Element Analysis" [79], "Guidelines to the Management of FEA" [80], "Safe Structural Assessment Quick Reference Manual" [81], "Safe Structural Assessment Management Guidelines" [82], and "Safe Structural Assessment Manual" [83].

9.7 Unix manuals

There are almost countless manual pages [84, 87], as well as several browser-style desktop help managers [85, 86, 88] on the HP and SGI UNIX systems. Also almost every package that is installed will add its own manual pages, its own browser-style help system and/or have built-in help [19, 20, 21, 26, 39, 89, 90, 91].

9.8 Other references

As well as the many on-line Unix manuals mentioned in section 9.7, there are many "UNIX How To" and "UNIX Quick Reference" type publications available in print and on the web, which cover all levels of users, ranging from the most experienced, to the freshest novice. Many books on UNIX and FEA can be found in the DSTO Research Library.

(And now a reminder to also see the AVD CSAF FE report catalogue, which is available at: http://jayted/CSAF/reports/catalogue.html.)

10. References

To make the following list of references more readable and also to save space, the following abbreviations have been used for the respective company publications:

MSC.Software	**MSC.Software Corporation**, 815 Colorado Boulevard, Los Angeles, CA 90041-1777, USA.
HKS	**HIBBITT, KARLSON & SORENSEN, INC.,** 1080 Main Street, Pawtucket, RI 02860-4847, USA.
ABAQUS	**ABAQUS, INC.,** 1080 Main Street, Pawtucket, RI 02860-4847, USA. *(Formerly HKS.)*
SER	**SER SYSTEMS LTD**, 39 Nottingham Road, Stapleford, Nottingham NG9 8AD, United Kingdom.
PACSYS	**PACSYS LTD**, Strelley Hall, Nottingham, NG8 6PE, United Kingdom. *(Formerly SER.)*
NAFEMS	**NAFEMS Ltd,** Whitworth Building, Scottish Enterprise Technology Park, East Kilbride, Glasgow G75 0QD, United Kingdom.
TMP	**Third Millennium Productions**, 303 Main Street, Fort Worth, Texas 76102, USA.
HP	**Hewlett-Packard Co**, 3000 Hanover St., Palo Alto, CA 94304, USA.
SGI	**Silicon Graphics Inc**, 1600 Amphitheatre Pkwy, Mountain View, CA 94043, USA.
GNU	GNU General Public License as published by the **Free Software Foundation, Inc**., 675 Mass Ave, Cambridge, MA 02139, USA.

10.1 MSC.Nastran

1. **MSC.Software**, *MSC.Nastran 2001 Release Guide,* Version 2001.

2. **MSC.Software**, *MSC.Nastran 2001 Installation and Operations guide*, Version 2001.

3. **MSC.Software**, *MSC.Nastran Theory Reference*, Version 68.

4. **MSC.Software**, *MSC.Nastran 2001 Quick Reference Guide*, Version 2001.

5. **MSC.Software**, *MSC.Nastran 2001 DMAP Programmer's Guide*, Version 2001.

6. **MSC.Software**, *MSC.Nastran 2001 Reference Manual*, Version 2001.

7. **MSC.Software**, *Getting Started*, Version 2001.

8. **MSC.Software**, *Linear Static Analysis*, Version 2001.

9. **MSC.Software**, *Basic Dynamic Analysis*, Version 2001.

10. **MSC.Software**, *Advanced Dynamic Analysis*, Version 2001.

11. **MSC.Software**, *Design Sensitivity and Optimisation*, Version 2001.

12. **MSC.Software**, *Thermal analysis*, Version 2001.

13. **MSC.Software**, *Numerical Methods*, Version 2001.

14. **MSC.Software**, *Aeroelastic Analysis*, Version 2001.

15. **MSC.Software**, *User Modifiable*, Version 2001.

16. **MSC.Software**, *Toolkit,* Version 2001.

17. **MSC.Software**, *MSC.Nastran On-line Encyclopedia*, Version 70.5.

18. **MSC.Software**, *MSC.Patran Installation and Operations Guide,* Version 2001(r3).

19. **MSC.Software**, *MSC.Patran Users Reference*, Versions 9.0 and 2001.

20. **MSC.Software,** *MSC.Patran PCL Programming Reference*, Versions 9.0 and 2001.

21. **MSC.Software,** *MSC.Patran Online Documentation*, All MSC.Patran Versions.

10.2 ABAQUS

22. **ABAQUS,** *Getting Started with ABAQUS/Standard (Interactive Version)*, Version 6.2.

23. **ABAQUS,** *Getting Started with ABAQUS/Standard (Keywords version)*, Version 6.2, (online only document).

24. **ABAQUS,** *ABAQUS/Standard User's Manual – Volumes I, II and III*, Version 6.2.

25. **ABAQUS,** *ABAQUS/Viewer User's Manual*, Version 6.2.

26. **ABAQUS,** *ABAQUS Online Documentation*, Version 6.2 (online manual).

27. **ABAQUS,** *ABAQUS Release Notes*, Version 6.2.

28. **ABAQUS,** *ABAQUS Site Guide*, Version 6.2.

29. **ABAQUS,** *ABAQUS Example Problems – Volumes I and II*, Version 6.2.

30. **ABAQUS,** *ABAQUS Benchmarks Manual*, Version 6.2 (online only document).

31. **ABAQUS,** *ABAQUS Verification Manual,* Version 6.2 (online only volume).

32. **ABAQUS,** *ABAQUS Keywords Manual*, Version 6.2.

33. **ABAQUS,** *ABAQUS Theory Manual*, Version 6.2 (online only volume).

34. **ABAQUS,** *ABAQUS Input Files*, Version 6.2 (online manual).

10.3 PACSYS PAFEC

35. **PACSYS**, *PAFEC-FE Data Preparation Manual – parts 1-3*, Level 8.

36. **PACSYS**, *PAFEC-FE Interactive Graphics Reference Manual – parts 1-3,* Level 8.

37. **PACSYS**, *PAFEC-FE PIGS Option Selector*, Level 8.

38. **PACSYS**, *PAFEC-FE Theory Manual*.

10.4 Amtec Tecplot

39. Amtec Engineering, Inc, 13920 SE Eastgate Way, Suite 220, Bellevue, WA 98005, USA, *Tecplot User's Manual*, Version 10.0.

10.5 TMP SLIM and Vision

40. **TMP**, *SLIM Users Manual,* Version 4.5, Larry Collins, June 14, 2003. (Located on ardvark in file /app/SLIM/4.50a/docs/SLIM_User_Guide.html.)

41. **TMP**, *TMP Vision Users Manual*, Version 4.50a, Larry Collins, June 26, 2003. (Located on ardvark in file /app/Vision/4.50a/docs/Guide/guide.html.)

10.6 NAFEMS references

42. **NAFEMS**, A finite element primer.

43. **NAFEMS**, A finite element dynamics primer.

44. **NAFEMS**, An introduction to non-linear FE analysis.

45. **NAFEMS**, Background to benchmarks.

46. **NAFEMS**, An explicit finite element primer, P Jacob and L Goulding, 2002.

47. **NAFEMS**, Finite element based fatigue calculations, Dr N W M Bishop and Dr F Sherrat, 2000.

48. **NAFEMS**, Why do FE analysis.

49. **NAFEMS**, Why do non-linear FE analysis.

50. **NAFEMS**, Why do design optimisation. David Spicer, 2002.

51. **NAFEMS**, Why do computational fluid dynamics? Dr C T Shaw, 2002.

52. **NAFEMS**, How to choose a FE System.

53. **NAFEMS**, How to get started with FE.

54. **NAFEMS**, How to get started in acoustics analysis.

55. **NAFEMS**, How to choose a pre/post processor.

56. **NAFEMS**, How to plan an FE analysis.

57. **NAFEMS**, How to interpret FE results.

58. **NAFEMS**, How to understand FE jargon.

59. **NAFEMS**, How to buy FE services.

60. **NAFEMS**, How to undertake fracture mechanics analysis.

61. **NAFEMS,** How to tackle non-linear finite element analysis.

62. **NAFEMS**, How to analyse composites, Dr W Marsden and Dr D J Irving, 2002.

63. **NAFEMS**, How to undertake finite element based geotechnical analysis, Andrew Mar, 2002.

64. **NAFEMS**, How to understand computational fluid dynamics jargon, Althea de Souza, 2003.

65. **NAFEMS**, How to use elements effectively, T Helen, 2003.

66. **NAFEMS**, How to plan a CFD analysis, AC de Souza, 2002.

67. **NAFEMS**, Understanding non-linear finite element analysis through illustrative benchmarks.

68. **NAFEMS**, NAFEMS workbook of examples.

69. **NAFEMS**, NAFEMS advanced workbook of examples and case studies (Volume 2), 2003.

70. **NAFEMS**, International Journal of CFD Case Studies (Vol. 3, July 2001)

71. **NAFEMS**, The standard NAFEMS benchmarks.

72. **NAFEMS**, Benchmarks for FE pre-processors.

73. **NAFEMS**, Selected benchmarks for forced vibration.

74. **NAFEMS**, Fundamental tests for creep.

75. **NAFEMS**, Selected benchmarks for material non-linearity.

76. **NAFEMS**, Composite benchmarks.

77. **NAFEMS**, Benchmarks for composite delamination. GAO Davies. R0084, Issue 1, 2002.

78. **NAFEMS**, Benchmarks for radiation and scattering of sound. Alfred J Svobodnik, Günter Hofstetter and Otto von Estorff. R0083, Issue 1, 2003.

79. **NAFEMS**, Quality system supplement to ISO 9001 relating to finite element analysis.

80. **NAFEMS**, Guidelines to the management of FEA.

81. **NAFEMS**, Safe structural assessment quick reference manual.

82. **NAFEMS**, Safe structural assessment management guidelines.

83. **NAFEMS**, Safe structural assessment manual.

10.7 Unix manuals

84. **HP**, HP man pages (on-line document, may be printed).

85. **HP**, HP Help Manager (CDE) (on-line document, may be printed).

86. **HP**, HP Instant Information (dynatext) (on- line document, may be printed).

87. **SGI**, SGI man pages (on-line document, may be printed).

88. **SGI,** SGI Help Manager (Insight) (on-line document, may be printed).

89. **GNU,** Xemacs Info, on-line reference and users guide (on-line document).

90. **GNU,** Ghostview user guide (on-line document), Copyright (C) 1992, Timothy O. Theisen.

91. **GNU,** GNU Ghostscript 5.50 users guide, (on-line document), Copyright (C) 1998, Aladdin Enterprises, Menlo Park, CA, USA.

10.8 Miscellaneous

92. Callinan, R.J. and Sanderson, S., Quality assurance for finite element work within Airframes and Engines Division, Aeronautical and Maritime Research Laboratory, DSTO-TN-0166, June 1998.

93. Anon, Untitled, reference on using Unix.

11. Acknowledgements

I also wish to acknowledge the assistance and guidance provided by Mr Witold Waldman, Mr Adam Barylko, Mr Richard Callinan and Dr Manfred Heller in preparing and shaping this document. I would also like to acknowledge the assistance from Aerostructures contractors Mr Nic Maan and Dr Norman Freund with information about the TMP products.

Appendix A Unix Commands Quick Reference

A.1. Commands lists

The following sections provide a reasonably comprehensive list of basic and advanced Unix commands with their key one-line descriptions. The command lists are provided in alphabetical order. The basic commands are what a user could expect to use on a regular basis during FEA on the CSAF. The advanced commands are those that a user could expect to use less frequently. The distinction is fairly artificial – some users may not find a need to use more than a few of the very basic Unix commands; others may find that their needs require the regular use of the so-called advanced commands.

It should also be noted that there are a number of commands that are built into specific command shells. The names of some of these commands are also common to several different shells, but their behaviour is dependent upon the particular shell that the user has invoked.

Help on a Unix system is found in manual pages. The manual page for a command is accessed using the man command. It has two forms of usage:

man [section] <command>

man –k <keyword>

The first form accesses the manual entry for a specific command. For example, to look in section 1M for the manual page entry for the bdf command:

man 1M bdf

The second form will search the manual page database for entries containing <keyword>. e.g.

man –k elephants

A.2. Unix commands "one-line" descriptions – Basic

alias	(sh builtin) create or display alias (shortcut) to a command
bdf(1M)	report number of free disk blocks (Berkeley version)
bg, fg	(sh builtin) resume a suspended process in background or foreground
cat(1)	concatenate, copy, and print files
cd(1)	change working directory
clear(1)	clear terminal screen
compress, uncompress, zcat(1)	compress and expand data
cp(1)	copy files and directory subtrees
df (generic)(1M)	report number of free file system disk blocks
dtmail(1)	the desktop mailer
dtpad(1)	edit text files
dtterm(1)	emulate a terminal window
du(1)	summarise disk usage
echo(1)	echo (print) arguments
exit, _exit(2)	terminate process
export	(sh builtin) set a sh shell environment variable
file(1)	determine file type
ftp(1)	file transfer program
gzip, gunzip, zcat(1)	compress or expand files
hostname(1)	set or display name of current host system
lp, lpalt, cancel(1)	print/alter/cancel requests on an LP printer or plotter
lpstat(1)	report line printer status information
ls, lc, l, ll, lsf, lsr, lsx(1)	list contents of directories
man(1)	find manual information by keywords; print out a manual entry
mesg(1)	permit or deny messages to terminal
mkdir(1)	make a directory
more, page(1)	file perusal filter for crt viewing
mv(1)	move or rename files and directories
newgrp(1)	switch to a new group
passwd(1)	change login password and associated attributes
pg(1)	file perusal filter for soft-copy terminals
ps(1)	report process status
pwd(1)	working directory name
quota(1)	display disk usage and limits
rcp(1)	remote file copy
rm(1)	remove files or directories

rmdir(1)	remove directories
sh(1)	overview of various system shells
sh, rsh(1)	Bourne shell, the standard/restricted command programming language
sh, rsh(1)	standard and restricted POSIX.2-conformant command shells
top(1)	display and update information about the top processes on the system
whence	(sh builtin) locate a program file including aliases and paths
which(1)	locate a program file including aliases and paths
who(1)	who is on the system
whoami(1)	print effective current user id
xterm(1)	terminal emulator for X

A.3. Unix commands "one-line" descriptions – Advanced

accept, reject(1M)	allow/prevent LP printer queuing requests
at, batch(1)	execute batched commands immediately or at a later time
awk(1)	pattern-directed scanning and processing language
bdiff(1)	diff for large files
chmod(1)	change file mode access permissions
chown, chgrp(1)	change file owner or group
cmp(1)	compare two files
comm(1)	select or reject lines common to two sorted files
command(1)	execute a simple command
cpio()	copy file archives in and out; duplicate directory trees
cron(1M)	timed-job execution daemon
crontab(1)	user job file scheduler
csh(1)	a shell (command interpreter) with C-like syntax
cut(1)	cut out (extract) selected fields of each line of a file
date(1)	display or set the date and time
diff(1)	differential file and directory comparator
diff3(1)	3-way differential file comparison
elm(1)	process electronic mail through a screen-oriented interface
enable, disable(1)	enable/disable LP printers
env(1)	set environment for command execution
fbackup(1M)	selectively back up files
find(1)	find files
finger(1)	user information lookup program
fold(1)	fold long lines for finite width output device
frecover(1M)	selectively recover files
grep, egrep, fgrep(1)	search a file for a pattern
groups(1)	show group memberships
gs(1)	Ghostscript (PostScript and PDF language interpreter and previewer)
head(1)	give first few lines
id(1)	print user and group IDs and names
kill(1)	send a signal to a process; terminate a process
ksh, rksh(1)	shell, the standard/restricted command programming language
ln(1)	link files and directories
login(1)	sign on; start terminal session
mail, rmail(1)	send mail to users or read mail
mailx(1)	interactive message processing system
make(1)	maintain, update, and regenerate groups of programs
makedepend(1)	create dependencies in makefiles

mc(1M)	media changer manipulation utility
merge(1)	three-way file merge
model(1)	print detailed hardware model information
mt(7)	magnetic tape interface and controls for stape, tape1 and tape2
mvdir(1M)	move a directory
news(1)	print news items
nice(1)	run a command at nondefault priority
nohup(1)	run a command immune to hangups
nroff(1)	format text
paste(1)	merge same lines of several files or subsequent lines of one file
ping(1M)	send ICMP Echo Request packets to network host
pr(1)	print files
printf(1)	format and print arguments
rcancel(1M)	remove requests from a remote line printer spooling queue
rcp(1)	remote file copy
read(1)	read a line from standard input
remsh, rexec(1)	execute from a remote shell
renice(1M)	alter priority of running processes
rev(1)	reverse lines of a file
rlogin(1)	remote login
rlp(1M)	send LP line printer request to a remote system
rlpstat(1M)	print status of LP spooler requests on a remote system
sdiff(1)	side-by-side difference program
sed(1)	stream text editor
shar(1)	make a shell archive package
sleep(1)	suspend execution for an interval
sort(1)	sort or merge files
split(1)	split a file into pieces
strings(1)	find the printable strings in an object or other binary file
strip(1)	strip symbol and line number information from an object file
tabs(1)	set tabs on a terminal
tail(1)	deliver the last part of a file
tar(1)	tape file archiver
tee(1)	pipe fitting
telnet(1)	user interface to the TELNET protocol
tftp(1)	trivial file transfer program
time(1)	time a command
timex(1)	time a command; report process data and system activity
tset, reset(1)	terminal-dependent initialisation
touch(1)	update access, modification, and/or change times of file
umask(1)	set or display the file mode creation mask
uniq(1)	report repeated lines in a file
uptime, w(1)	show how long system has been up, and/or who is logged in and what they are doing

uuencode, uudecode(1)	encode/decode a binary file for transmission by mailer
vi, view, vedit(1)	screen-oriented (visual) text editor
vis, inv(1)	make unprintable and non-ASCII characters in a file visible or invisible
wall(1M)	write message to all users
wc(1)	count words, lines, and bytes or characters in a file
whereis(1)	locate source, binary, and/or manual for program
write(1)	interactively write (talk) to another user
xargs(1)	construct argument list(s) and execute command
xdm(1M)	X Display Manager with support for XDMCP
xjdm(1)	monitor network printing devices
xjmore(1)	display a file in a scrollable Motif window
xkill(1)	kill a client by its X resource
xlsclients(1)	list client applications running on a display
xmkmf(1)	create a Makefile from an Imakefile
xpr(1)	print an X window dump
xrefresh(1)	refresh all or part of an X screen
xwd(1)	dump an image of an X window
zcmp, zdiff(1)	compare compressed files
zgrep(1)	search possibly compressed files for a regular expression
zmore(1)	file perusal filter for crt viewing of compressed text
znew(1)	recompress .Z files to .gz files

Appendix B New user mail message

B.1. New User mail message template

Outlook 2000 template file: jayted New account8.oft Date: 2005-01-11

(This and similar templates for bigted, etc., are located in the following shared folder on \\Luniss\public\MSoffice\Template.)

Sent to: Xxx
CC: AVD CSAF Help; 'root@jayted.dsto.defence.gov.au'; TaskManager
Subject: jayted user account setup for Xxx

Dear Xxx,

Welcome to the **Air Vehicles Division Computational Structural Analysis Facilities.**

Name:	xxx
Task manager:	xxxman
Your task no:	aaaaa
Your task W/As:	aaaaa

You now have a new account on jayted. Your home directory is /home/username.

user name:	username
user id:	iiii
group name:	xxxxx
group id:	iiii
login shell:	sh – posix
password:	**********

Notes:

1) There are two user areas and two scratch areas:
   a) /home (118GB) - You live here & work on **small**-medium jobs. (Data IS backed up.)
   b) /tmpwrk (136B) - You work on **larger** jobs here. (Data is NOT backed up.)
   c) /tmpfe/abaqus (68GB) - all of your abaqus FE scratch files go into here by default. (Data is removed regularly.)
   d) /tmpfe/msc (68GB) - all of your nastran FE scratch files go into here by default. (Data is removed regularly.)

2) Access to CSAF is usually via XDMCP logins using either X-Win32 software or HP, Labtam or Tektronix X-terminals.

3) CDE is the desktop window manager.

4) Software available (quick list – subject to change):

MSC.Patran v 2001& 2003	commands: p2001 & p2001, patran, p3
MSC.Nastran v 2001 & 2004	commands: nast2001& nast2004, nastran
ABAQUS v 6.2-7, 6.3-1	commands: abq627 & abq631, abaqus
Xemacs v21.1	command: xemacs
Netscape v4.79	command: netscape
Mozilla v1.6	command: mozilla
Tecplot v9.0	command: tecplot
Zencrack v7.3	command: runzcr73

5) The present software license agreements allow us to run up to following number of simultaneous jobs on each server:

FEA Software	Licences on Server jayted	poisson2	bigted
patran	10	0	0
abaqus	2	1	0
nastran	2	2	0
pafec	2	0	2
zencrack	0	1*	0

*Note that zencrack uses the abaqus seat on poisson2

6) Anything else: If you have *ANY* problems, questions or requests in relation to the CSAF then please mail them to: **AVD CSAF Help** (avdcsafhelp@dsto.defence.gov.au)

7) You have also been added to the **AVD CSAF Users** mail list, through which you will receive regular updates on the AVD CSAF.

8) As a matter of high priority, please familiarise yourself with the DSTO POLICY on Information System Security Practice & Procedures (IS-SPP) For the DSTO RESTRICTED Network:

http://web-vic.dsto.defence.gov.au/workareas/SCIS/networks/dsto-sec-003.pdf

Good luck & happy computing!!

bfn,

SS :-)
AVD CSAF Sys Admin
Voicemail: 9626-7278
Facsimile: 9626-7089

DISTRIBUTION LIST

Air Vehicles Division Computational Structural Analysis Facilities Policy and Guidelines for Users

S. Sanderson

AUSTRALIA

DEFENCE ORGANISATION

	No. of copies
S&T Program	
Chief Defence Scientist	Shared
FAS Science Policy	
AS Science Corporate Management	
Director General Science Policy Development	
Counsellor Defence Science, London	Doc Data Sheet
Counsellor Defence Science, Washington	Doc Data Sheet
Scientific Adviser to MRDC, Thailand	Doc Data Sheet
Scientific Adviser Joint	1
Navy Scientific Adviser	Doc Data Sht & Dist List
Scientific Adviser – Army	Doc Data Sht & Dist List
Air Force Scientific Adviser	Doc Data Sht & Dist List
Scientific Adviser to the DMO	Doc Data Sht & Dist List
Platforms Sciences Laboratory	
Director, PSL	Doc Data & Exec Summary
Chief of Air Vehicles Division	Doc Data Sht & Dist List
Research Leader – Aerospace Materials	Doc Data Sht & Dist List
Research Leader – Flight Systems	Doc Data Sht & Dist List
Research Leader – Aircraft Structures	Doc Data Sht & Dist List
Research Leader – Propulsion Systems	Doc Data Sht & Dist List
Head Structural Mechanics: Manfred Heller	1
Task Manager: Garry White	1
Author: Stephen Sanderson	4
Adam Barylko	1
Richard Callinan	1
Greg McKenzie	1
Witold Waldman	1
AVD Task Managers	
Ross Antoniou	Doc Data Sht & Dist List
Pud Baburamani	Doc Data Sht & Dist List
Richard Bartholomeusz	Doc Data Sht & Dist List
Robert Boykett	Doc Data Sht & Dist List
Paul Callus	Doc Data Sht & Dist List

Jan Drobik	Doc Data Sht & Dist List
Shane Dunn	Doc Data Sht & Dist List
Phil Jackson	Doc Data Sht & Dist List
Dom Lombardo	Doc Data Sht & Dist List
Marcus McDonald	Doc Data Sht & Dist List
Lorrie Molent	Doc Data Sht & Dist List
Nic Rajic	Doc Data Sht & Dist List
Frank Skidmore	Doc Data Sht & Dist List
Roger Vodicka	Doc Data Sht & Dist List
Kevin Walker	Doc Data Sht & Dist List
Chun Wang	Doc Data Sht & Dist List
Kevin Watters	Doc Data Sht & Dist List
Bryon Wicks	Doc Data Sht & Dist List
Albert Wong	Doc Data Sht & Dist List

WSD Task Managers

Vinod Puri	Doc Data Sht & Dist List

AVD CSAF Users

Rohith Amaratunga	Doc Data Sht
Sameera Bandara	Doc Data Sht
Jonathan Blight	Doc Data Sht
Ronald Braemar	Doc Data Sht
Madeleine Burchill	Doc Data Sht
Guangxia Chen	Doc Data Sht
David Conser	Doc Data Sht
Tim Cooper	Doc Data Sht
Ben Dixon	Doc Data Sht
Rebecca Evans	Doc Data Sht
John Faragher	Doc Data Sht
Soon-Aik Gan	Doc Data Sht
Trenton Gilbert	Doc Data Sht
Noel Goldsmith	Doc Data Sht
Rebecca Gravina	Doc Data Sht
Alex Harman	Doc Data Sht
John Huynh	Doc Data Sht
Shane Hill	Doc Data Sht
Jianfu Hou	Doc Data Sht
Weiping Hu	Doc Data Sht
Mladen Ignjatovic	Doc Data Sht
Robert Kaye	Doc Data Sht
Luther Krake	Doc Data Sht
Oleg Levinski	Doc Data Sht
Qianchu Liu	Doc Data Sht
Paul Marsden	Doc Data Sht
Leo Mirabella	Doc Data Sht
Trung Nguyen	Doc Data Sht
Matthew Phillips	Doc Data Sht
Colin Pickthall	Doc Data Sht

Susan Pitt	Doc Data Sht
Frank Polanco	Doc Data Sht
Benjamin Semple	Doc Data Sht
Khan Sharp	Doc Data Sht
Vijay Sridhar	Doc Data Sht
John Thornton	Doc Data Sht
Yu Chee Tong	Doc Data Sht
Stuart Trezise	Doc Data Sht
Kelly Tsoi	Doc Data Sht
Anthony Walley	Doc Data Sht
John Wang	Doc Data Sht
Ron Wescott	Doc Data Sht
Michael Young	Doc Data Sht
Wyman Zhuang	Doc Data Sht

DSTO Library and Archives

Library Fishermans Bend	Doc Data Sheet
Library Edinburgh	1
Defence Archives	1

Capability Development Group

Director General Maritime Development	Doc Data Sheet
Director General Capability and Plans	Doc Data Sheet
Assistant Secretary Investment Analysis	Doc Data Sheet
Director Capability Plans and Programming	Doc Data Sheet

Chief Information Officer Group

Deputy CIO	Doc Data Sheet
Director General Information Policy and Plans	Doc Data Sheet
AS Information Strategy and Futures	Doc Data Sheet
AS Information Architecture and Management	Doc Data Sheet
Director General Australian Defence Simulation Office	Doc Data Sheet
Director General Information Services	Doc Data Sheet

Strategy Group

Director General Military Strategy	Doc Data Sheet
Director General Preparedness	Doc Data Sheet
Assistant Secretary Governance and Counter-Proliferation	Doc Data Sheet

Navy

Maritime Operational Analysis Centre, Building 89/90 Garden Island Sydney NSW	Doc Data Sht & Dist List
Deputy Director (Operations)	
Deputy Director (Analysis)	
Director General Navy Capability, Performance and Plans, Navy Headquarters	Doc Data Sheet
Director General Navy Strategic Policy and Futures, Navy Headquarters	Doc Data Sheet

Air Force

SO (Science) – Headquarters Air Combat Group, RAAF Base, Williamtown NSW 2314	Doc Data & Exec Summary
Air Movements Training and Development Unit, RAAF Base, Richmond NSW 2755	
Chief Engineer – CENGR	1
Senior Engineering Officer – SENGO	1 (PDF)

Army

ABCA National Standardisation Officer	e-mailed Doc Data Sheet
Land Warfare Development Sector, Puckapunyal	
SO (Science) – Land Headquarters (LHQ), Victoria Barracks NSW	Doc Data & Exec Summary
SO (Science), Deployable Joint Force Headquarters (DJFHQ) (L), Enoggera QLD	Doc Data Sheet

Joint Operations Command

Director General Joint Operations	Doc Data Sheet
Chief of Staff Headquarters Joint Operations Command	Doc Data Sheet
Commandant ADF Warfare Centre	Doc Data Sheet
Director General Strategic Logistics	Doc Data Sheet

Intelligence and Security Group

DGSTA Defence Intelligence Organisation	1
Manager, Information Centre, Defence Intelligence Organisation	1 (PDF)
Assistant Secretary Capability Provisioning	Doc Data Sheet
Assistant Secretary Capability and Systems	Doc Data Sheet
Assistant Secretary Corporate, Defence Imagery and Geospatial Organisation	Doc Data Sheet

Defence Materiel Organisation

Deputy CEO	Doc Data Sheet
Head Aerospace Systems Division	Doc Data Sheet
Head Maritime Systems Division	Doc Data Sheet
Chief Joint Logistics Command	Doc Data Sheet

Defence Libraries

Library Manager, DLS-Canberra	Doc Data Sheet

OTHER ORGANISATIONS

National Library of Australia	1
NASA (Canberra)	1
State Library of South Australia	1
Aerostructures	1
Manager FB site: Robert Koning	

UNIVERSITIES AND COLLEGES

Australian Defence Force Academy

Library	1
Head of Aerospace and Mechanical Engineering	1
Hargrave Library, Monash University	Doc Data Sheet
Librarian, Flinders University	1

OUTSIDE AUSTRALIA

INTERNATIONAL DEFENCE INFORMATION CENTRES

US Defense Technical Information Center	1 PDF
UK Dstl Knowledge Services	1 PDF
Canada Defence Research Directorate R&D Knowledge & Information Management (DRDKIM)	1
NZ Defence Information Centre	1

ABSTRACTING AND INFORMATION ORGANISATIONS

Library, Chemical Abstracts Reference Service	1
Engineering Societies Library, US	1
Materials Information, Cambridge Scientific Abstracts, US	1
Documents Librarian, The Center for Research Libraries, US	1
SPARES	5

Total number of copies: Printed: 33 PDF: 4

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION
DOCUMENT CONTROL DATA

1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)

2. TITLE

Air Vehicles Division Computational Structural Analysis Facilities Policy and Guidelines for Users

3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION)

Document	(U)
Title	(U)
Abstract	(U)

4. AUTHOR(S)

S. Sanderson

5. CORPORATE AUTHOR

Platforms Sciences Laboratory
506 Lorimer St
Fishermans Bend Victoria 3207 Australia

6a. DSTO NUMBER	6b. AR NUMBER	6c. TYPE OF REPORT	7. DOCUMENT DATE
DSTO-GD-0435	AR-013-392	General Document	May 2005

8. FILE NUMBER	9. TASK NUMBER	10. TASK SPONSOR	11. NO. OF PAGES	12. NO. OF REFERENCES
M1/9/1363	RDI 03/058	CAVD	102	93

13. URL on the World Wide Web

http://www.dsto.defence.gov.au/corporate/reports/DSTO-GD-0435.pdf

14. RELEASE AUTHORITY

Chief, Air Vehicles Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for public release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS Yes

18. DEFTEST DESCRIPTORS

Finite element analysis, military aircraft, structural integrity, airworthiness, data management, computer programs, computer systems hardware

19. ABSTRACT

This document presents the operating policies and guidelines for the use of the Computational Structural Analysis Facilities (CSAF) within Air Vehicles Division (AVD). It has been created to assist users of the CSAF to perform professional, high quality finite element analysis (FEA). FE analysts from many tasks within AVD are using the facilities to conduct FEA with respect to the assessment of structural integrity and technical airworthiness of ADF aircraft. This document covers key issues such as data and resource management, as well as providing a comprehensive overview of the hardware and software facilities that are available for use. It also outlines recommended generic FEA modelling procedures and provides an overview of key FEA manuals and references. The specific FEA application software that is covered includes MSC.Nastran, MSC.Patran, ABAQUS, PAFEC and PIGS.