

Advanced Graphical User Interface to the MAAP/Fugen Simulator System

January, 1998

FUGEN NUCLEAR POWER STATION
POWER REACTOR AND NUCLEAR FUEL DEVELOPMENT CORPORATION

Enquiries about copyright and reproduction should be addressed to:

Technical section, Fugen Nuclear Power Station,
Power Reactor and Nuclear Fuel Development Corporation
3 Myojin-cho, Tsuruga, Fukui-Ken 914-8510, Japan

Copyright ©1998

Power Reactor and Nuclear Fuel Development Corporation

January, 1998

Advanced Graphical User Interface to the MAAP/Fugen Simulator System

Per Christer LUND*

Abstract

A new and improved Graphical User Interface (GUI) to the Modular Accident Analysis Program for FUGEN (MAAP/FUGEN) has been developed and implemented at Fugen. The new user interface is a superset of the existing GUI to MAAP - the MAAP/FUGEN/GRAAPH - in the meaning that it contains all the features of the GRAAPH, but in addition offers a number of new features.

The new interface, named MAAP-PICASSO is based on the Picasso-3 technology developed by Institutt for Energiteknikk/OECD Halden Reactor Project. The main difference between the MAAP-PICASSO and MAAP-FUGEN-GRAAPH GUIs is that the MAAP-PICASSO GUI is completely decoupled from the numerical simulator. This gives a far higher flexibility regarding enhancement of the GUI, connection to other, external software and user friendliness. It also includes techniques for presenting 2 byte character set - i.e. displaying text in Japanese characters. A special software has been developed for automatic extraction and reuse of the graphical plant information provided in MAAP/GRAPH into Picasso language. This software has been demonstrated not only on the Fugen plant data, but also other Nuclear Power Plant picture definitions provided by Fauske Inc.

The new GUI requires a minimal modification of the MAAP code itself. However, these modification is only for parameter communication and is not intrusive to the numerical computations of MAAP itself.

The GUI has been developed using modular and object-oriented programming techniques, which makes it relatively easy to extend and modify to fulfill present and future requirements from the users at Fugen, and makes it compatible with future versions of the MAAP code. MAAP-PICASSO is developed on and currently running only on HP UNIX workstations. However, a new NT-based version of Picasso-3 will be released from the Halden Project in February 1998. This will further enhance the applicability and usability of the MAAP-PICASSO GUI.

* PNC international fellow on leave from Institutt for Energiteknikk/ OECD Halden Reactor Project,
PSR Group, Fugen Nuclear Power Station, Power Reactor and Nuclear Fuel Development Corporation

1 9 9 8 年 1 月

MAAP/Fugenシミュレータシステムの 先進的グラフィカルユーザインターフェイス

ペル・クリスティル・ルンド*

要 旨

ふげん版モジュール型事故解析プログラム(MAAP/Fugen)の新たに改善されたグラフィカル・ユーザー・インターフェイス(GUI)が開発され、実際にふげんで使用された。新しいインターフェイスは、既存のMAAPのGUI—MAAP/FUGEN/GRAAPH—の上位版である(このインターフェイスが、GRAAPHのすべての機能を有し、加えていくつかの新機能を提案しているという意味で)。

MAAP-PICASSOと名付けられた新しいインターフェイスは、ノルウェーエネルギー技術研究所/OECDハルデン・リアクター・プロジェクトによって開発された、Picasso-3の技術を基盤にしている。MAAP-PICASSOとMAAP-FUGEN-GRAAPHのGUIの主な違いは、MAAP-PICASSOのGUIが数値シミュレータから完全に分離していることである。このことは、GUIの向上、外部ソフトウェアや他との接続、そしてユーザーとの親和性を考えると、さらに高い自由度を与える。また、2バイト文字セット一すなわち日本語文字のテキストを表現する技術も含んでいる。MAAP/GRAAPHに与えられるグラフィカルプラント情報をPicassoプログラミング言語で自動的に取得・再利用するために、特別のソフトウェアが開発された。このソフトウェアは、「ふげん」のプラントデータ上だけでなく、Fauske社から与えられた他の原子力プラントの画面でも実証された。

新たなGUIは、MAAPコード自身の最低限の修正が必要になる。しかしながら、それらの修正はパラメータ通信のためだけで、MAAP自身の数値計算をじやますものではない。

GUIは、モジュール型オブジェクト指向プログラミング技術を使用して開発され、「ふげん」のユーザーからの現在及び将来の要求に応える拡張と変更が比較的容易に行える他、MAAPコードの将来バージョンとも互換性を有する。MAAP-PICASSOは、HP UNIXワークステーション上でのみ、開発され現在稼動している。しかしながら、新しいPicasso-3のNT版が、1998年2月にハルデンプロジェクトから公開されるだろう。これは、MAAP-PICASSOのGUIの適用性と使いやすさをさらに高めることになろう。

* ノルウェーエネルギー技術研究所/OECDハルデン・リアクター・プロジェクトより、国際特別研究員として動力炉・核燃料開発事業団 新型転換炉ふげん発電所 P S R グループに在籍

CONTENTS

1. MAAP-FUGEN and MAAP-FUGEN-GRAAPH	1
1.1. General	1
1.2. The MAAP-FUGEN-GRAAPH	1
1.3. MFG input files	4
2. Picasso-3	5
2.1. Key Points	5
2.2. The Picasso-3 Concept	5
2.3. Graphics Editor	5
2.4. Run Time Manager	7
2.5. Picasso-3 special features	7
2.6. On-line compilation	7
2.7. The Picasso-3 language pTALK	8
2.8. Dialogues	8
2.9. Run-time creation	8
2.10. Reuse of Interface Components class concept	8
2.11. Class behavior	9
2.12. Reuse of classes	9
2.13. Integration and Communication	9
2.14. File Formats	11
3. Differences between MAAP-FUGEN-GRAAPH and Picasso-3	12
3.1. Simulator and GUI integration	13
3.2. Modifications of the GUI	13
3.3. Interfacing with other software	14
3.4. Portability	14
3.5. Japanese characters	14

4. Components of the MAAP-PICASSO GUI	15
4.1. Modifications of the original MAAP code	15
4.2. The communication between MAAP and Picasso	16
4.3. Extension and modification of the code	19
4.4. The Real Time Manager Data Base (shared variables)	20
5. The MAAP-PICASSO Graphical User Interface (MG)	21
5.1. The main window	21
5.2. Starting the MAAP simulator	22
5.3. The main window Menus	24
5.4. The Compartment window	25
5.5. The Variable Trend window	26
5.6. Displaying other plant pictures	28
6. Automatic conversion of MAAP-FUGEN-GRAAPH picture files to Tdoc code	29
6.1. Reading the GRAAPH Picture File:	29
6.2. Producing Tdoc picture objects:	30
6.3. Library objects	31
6.4. Structure of the "m2p" program	31
6.5. Making the compartment window	33
6.6. Finishing up	33
7. References	34
Appendix 1: Directory listing and file descriptions	35
Appendix 2: Source codes.....	37

LIST OF FIGURES

Figure 1. The MAAP-FUGEN-GRAAPH Graphical User Interface	3
Figure 2. The Picasso-3 GED during editing of the Fugen picture	6
Figure 3. System layout for the two alternative GUIs	13
Figure 4. The main window at startup of MAAP-PICASSO	22
Figure 5. The main window during MAAP simulation	24
Figure 6. Compartment window	25
Figure 7. The normal trend window	26
Figure 8. Polar Plot mode of the trend window	27
Figure 9. MAAP-PICASSO applied on a PWR	28
Figure 10. MFG picture file to Tdoc files translation program	29

LIST OF TABLES

Table 1. MAAP-FUGEN-GRAAPH input files	4
Table 2. Modified MAAP FORTRAN files	15
Table 3. C files and functions included into MAAP	17
Table 4. The RTM resource and database files	20

1. MAAP-FUGEN and MAAP-FUGEN-GRAAPH*

MAAP-FUGEN and MAAP-FUGEN-GRAAPH are developed by Fauske & Associates, Inc., Illinois, USA.

1.1. General

The Modular Accident Analysis Program for the Fugen plant (MAAP-FUGEN) is a computer code that can simulate the response of the Fugen power plant during severe accident, including actions taken as a part of accident management. These quantitatively predict the evolution of a severe accident starting from full power conditions given a set of system faults and initiating events through events such as core melt, reactor coolant system failure, calandria tank failure, shield tank failure, and containment failure. Furthermore, models are included to represent the actions that could stop the accident by cooling the debris in the calandria tank or containment.

MAAP-FUGEN also includes a graphical interface, MAAP-FUGEN-GRAAPH, to increase the capability for analysis and review of the results. Specifically, the user can interactively interface with the code during execution and directly observe the results.

MAAP-FUGEN models thermal hydraulics and fission product behavior in the reactor coolant system loop 1 and 2, calandria tank, shield tank, and containment. Models are included for engineered safeguard system logic and performance. Also, operator actions are simulated by specification of intervention conditions and responses.

1.2. The MAAP-FUGEN-GRAAPH

The MAAP-FUGEN-GRAAPH code (MFG) is a Graphical User Interface integrated with the MAAP-FUGEN code. The MFG software uses the MAAP-FUGEN code physical models to display the thermal-hydraulic and fission product behavior during postulated accident condition at nuclear power plants along with providing the user with an enhanced interface to interact with the MAAP-FUGEN code.

The MFG features include:

1. *Data Input Procedures.* With MFG, the user can initiate an MAAP-FUGEN sequence by interactively answering a series of questions to define the sequence. The option of utilizing a pre-defined MAAP-FUGEN input deck to start an accident simulation is also available. MFG provides pulldown menus, enabling the user to interactively modify accident states and plant conditions as the sequence progresses.

* This chapter is copied from the MAAP User Manual (1)

2. *Graphical Representation of Component Control.* MFG provides the user with control over pumps, valves, and other components modeled by MAAP-FUGEN via the computer's mouse. To turn on a pump/valve using MFG, the user has to select the Plant Components pulldown menu option, position the mouse over the pump/valve and click left mouse button to highlight the component and associated piping. If the user clicks on the same component again, a pop-up window will appear allowing the user to select the operating mode for the pump/valve.
3. *Graphical Representation of Output Parameters.* The MFG graphical display provides the user with a color graphical schematic representation of the Fugen plant, depicting the reactor coolant system, calandria tank, shield tank, containment, and the engineered safety features (EFSs). MFG displays the following information:
 - *Digital clock showing the current transient time*
 - *Fugen schematic representation*
 - *Color coding to indicate status (on/off) and quantities (water/steam/hydrogen)*
 - *Water levels*
 - *Hydrogen mass fraction in the reactor coolant system*
 - *Containment sprays*
 - *Concrete attack on the containment floor if calandria tank failure occurs and debris is accumulated in the containment.*
4. *Graphic Representation of Core Heatup and Transport of Containment.* MFG provides a graphical representation of the core fuel region showing a color (representing temperatures) and a size (representing mass) for each of the individual core nodes.
5. *Stop and Restart Control.* At any time during an accident situation, the user can terminate the sequence via a menu option. Restart a previous analyzed sequence can also be initiated using an MAAP-FUGEN restart file. In addition, MFG offers a time-lapse restart mode allowing selective viewing of the MAAP-FUGEN generated restart file to determine when the simulation should be restarted.
6. *Text Display.* By using the MAAP-FUGEN "Report" function, a pre-defined text window display of the key plant parameters can be activated.
7. *Zoom.* The user can expand any area of the MFG display to reveal default and user-defined information about the area of interest. The user can display a large number of MAAP-FUGEN calculated variables with this option.
8. *Real Time Variable Plotting.* Via the MFG pulldown menus, on-screen plots of the MAAP-FUGEN calculated variables can be displayed. When using MFG, X-Y plots of MAAP-FUGEN calculated variables can be displayed on the screen.

9. *Movie Playback.* A movie playback mode is also included to display the results from a previously executed MFG sequence. The movie playback feature enables long accident or CPU time sequences to be displayed and reviewed, in their entirety, within a short time.
10. *Graphical Representation of Calculated Containment Nodalization Quantities.* By using an outlined containment nodalization, the user can display containment nodal quantities such as fission product distribution and hydrogen concentrations.
11. *View Spots.* Buttons known as view spots can be placed on the MFG display to provide the user with the ability to X-Y plot a selected set of variables.

Figure 1. shows the MFG user interface.

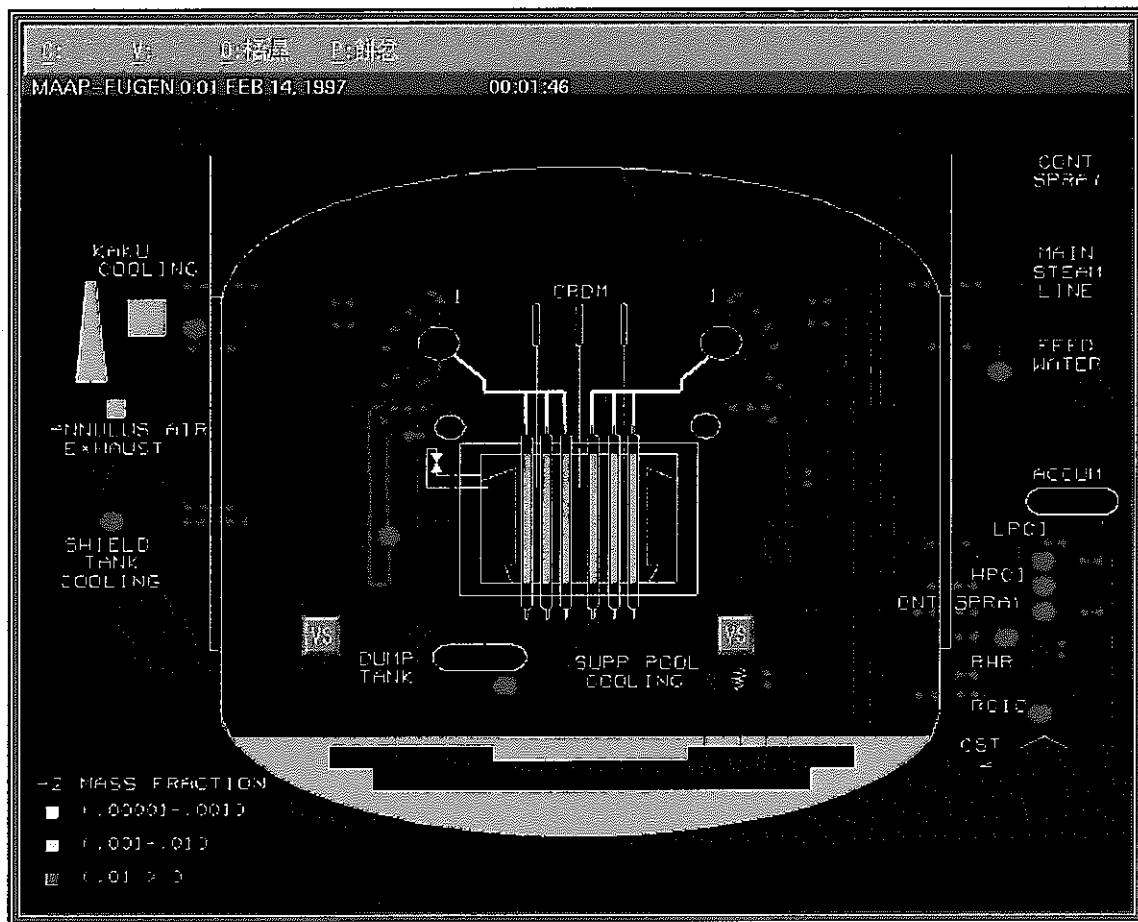


Figure 1. The MAAP-FUGEN-GRAAPH Graphical User Interface

1.3. MFG input files

The MFG input consists of a set of text files, provided by the user. These files are listed below:

Name	Req.	Description	Ext.
Parameter file	Yes	defines the plant and consists mainly of plant-specific data that does not change from one run to the other	
Picture file	Yes	Plant-specific description used by MFG to graphically represent the plant. This file contains all the coordinates for producing the shapes on the MFG display.	.pic
Text file	Yes	Plant-specific file corresponding to the MFG picture file. The text contains descriptions and coordinates for each of the systems represented on the MFG display.	.txt
Variable file	Yes	Plant-specific file corresponding to the MFG picture file. Contains a default set of variables and locations to be written to MFG screen.	.dat
Include File	No	Contains job control instructions.	
Input Deck	No	Can be used to initiate MFG.	
Label File	No	used for X-Y Plot and View Spots	
MFG parameter file	Yes	Contains information for defining the containment-nodalized quantities, view spot locations and variables, and a time delay for component control.	

Table 1. MAAP-FUGEN-GRAAPH input files.

2. Picasso-3[†]

Picasso-3 is a User Interface Management System (UIMS) for developing Graphical User Interfaces (GUIs). An interactive graphics editor can be used to test a user interface immediately in order to verify that dynamic behavior works as intended. Moreover, GUI components can be created, modified, and deleted at run-time. The Picasso-3 system is designed to operate in a network environment. It can run on one workstation while the application programs that it services can be distributed over several workstations.

2.1. Key Points

Important features and characteristics of the Picasso-3 system include:

- Picasso-3 is the third generation of the Picasso project. It is built on more than 20 years of experience in development of graphical user interfaces. Earlier versions of Picasso have been used in more than 40 industrial applications.
- Picasso-3 is being developed using C++ and object-oriented programming techniques.
- An Interactive Graphics Editor makes development and testing of user interfaces more efficient. This allows Picasso-3 to be used as a prototyping tool for creating user interfaces independently of actual applications.
- Separation of central algorithms from the graphical presentation makes maintenance less expensive.
- The design of user interfaces can be optimized using the object-oriented class concept offered by the system.

2.2. The Picasso-3 Concept

The Picasso-3 system is a flexible and powerful tool for developing and testing graphical user interfaces. Furthermore it is an environment for executing applications. It is the third generation of User Interface Management Systems (UIMS) developed as part of the OECD Halden Reactor Project since 1984.

2.3. Graphics Editor

Picasso-3 system offers an interactive design/test environment that makes development and testing more efficient. The Graphics Editor (GED) is used in this process. GED has two modes of operation;

- Edit-mode is for drawing graphics and the creation of dynamics and dialogues
- Test-mode is for testing dynamics on-line.

Switching between the two modes is done from a menu in GED. In this way, switching between the design and testing of the interfaces can be done quickly and efficiently. Using this facility, Picasso-3 can be used as a powerful prototyping tool.

[†] This chapter is copied from the Picasso-3 User Manual (2)

Even though savings in this phase of an application's lifecycle can be considerable, the real potential for savings can be found in the maintenance phase.

This is when the application is installed and is used for its intended purpose. In this phase, errors are usually found and corrected, but when Picasso-3 is used, correcting them is not the main activity. In fact, most of the developers time is likely to be spent improving and enhancing the user interface functionality.

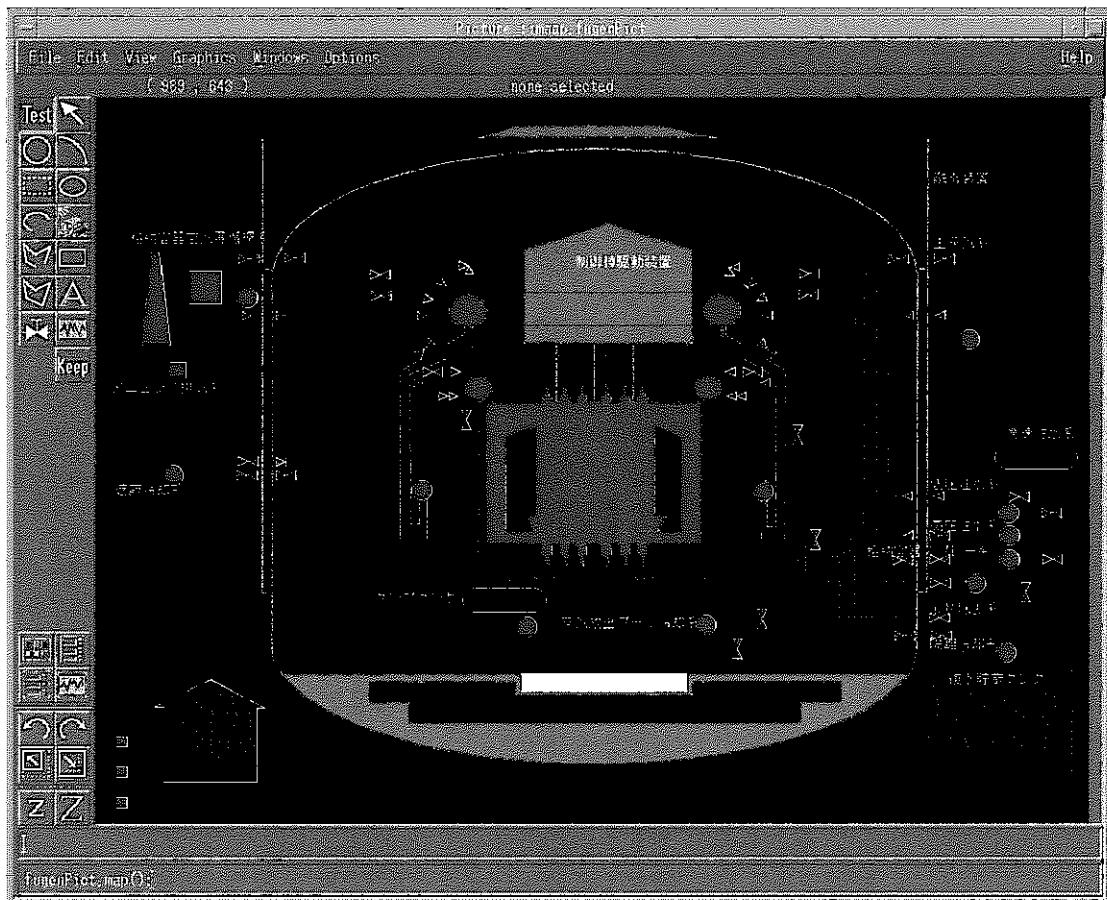


Figure 2. The Picasso-3 GED during editing of the Fugen picture.

Most changes in functionality involve presenting information that is already stored in a system in new or alternative ways. The best approach to reducing the maintenance costs is therefore to *separate the user interface from the application code and information storage*. A modification of the interface will then require minimal changes to the application's code. A major requirement of the Picasso-3 system is to offer a complete separation of the user interface from the application code.

The Picasso-3 system is based on industry standards. It runs under X-Windows on standard UNIX workstations and has been developed using the C++ programming language and object oriented programming techniques. Building software applications today, is much about choosing the right tool for the right job. This means that application developers must pick from the best of the breed and that Picasso-3 must be able to operate in a distributed computing environment. It must cooperate with other running tasks and support client-server architecture.

2.4. Run Time Manager

The central part of the system is the Run Time Manager (RTM), which realizes the user interface.

API Communication is performed through a set of high-level interface routines that are linked with every program. These Application Programmer's Interface (API) routines give application programs access to all functionality in the RTM.

Picasso-3 classes Building graphical user interfaces is a time consuming process. A major feature of the Picasso-3 system is that the user can develop reusable high-level user interface components. These components are building blocks that represent real world objects such as valves, tanks and pumps of different kinds. In Picasso-3 terminology, the representation of these real world objects are called Picasso-3 Data Presentation Central Algorithms and Data Storage

Most changes in applications are done in the presentation of data that is already stored in the application. The separation of central algorithms from the presentation part is a key issue in reducing application-developing time.

2.5. Picasso-3 special features

The Graphics Editor is used to create the look (appearance) and feel (behavior) of graphical objects. These can be primitive shapes or instances of high-level classes.

The design and implementation of graphical user interfaces is typically an iterative process. This means that the application designer design and develop the interface, test it out, and then redesign it and test again etc. The design tool should therefore support the designer with methods that simplify this iterative process.

The Picasso-3 Graphics Editor offers general editing functionality similar to that which one would expect to find in a general purpose graphics editor, as well as more specialized functionality for creating dynamic graphics.

2.6. On-line compilation

User interface development is traditionally carried out in two steps. First, to use some off-line tools to specify the interface and somehow compile this specification and then to run an on-line system that uses the compiled user interface specification. In Picasso-3 there is no distinction between off-line and on-line. All Picasso-3 tools operate on-line and therefore no off-line re-compilation and restarting of the system is necessary. The editing functionality is an integrated part of the Picasso-3 system, and hence editing operations are available at all times. This is utilized in the graphics editor by allowing the user interface designer to easily switch between the design and test modes.

In the test mode, the system runs in its real run-time environment and all parts of the system can be tested. If the actual application is available, it can be connected to the Picasso-3 system while the user interface is being designed and edited, enabling the designer to test the interface using real application data.

A user interface consists of more than static graphics. In order to be able to reflect changes in an application program, it must be possible to connect the interface dynamically to application program data. In Picasso-3, the end user can interact with the system through dialogues, describing what actions are to be executed as a response to a user's input.

2.7. The Picasso-3 language *pTALK*

A key feature of the graphics editor is the functionality for the creation of dynamic graphics, for connecting graphics to application data. Dynamics are created using Picasso-3's user interface language, *pTALK*. *pTALK* is a C/C++ like language with additional constructs for graphics manipulation. The language offers a powerful and flexible way of creating dynamics. The language is compiled by the RTM at run-time, and hence new *pTALK* code can be added and tested/executed immediately during the design as well as during the execution of an interface.

2.8. Dialogues

Another important feature of the Picasso-3 system is the ability to create powerful and flexible dialogues. In Picasso-3, a dialogue is defined as the interactive part of a graphic element, and is integrated with the graphic element itself. Dialogues can be defined at different levels in an interface. User interfaces developed in Picasso-3 consist of three levels: from top to bottom, the application level, the picture level, and the element level. This hierarchical structure allows the user interface designer to create local as well as more global dialogues.

An example of the use of a dialogue attached to a valve object would be to toggle the state of the valve whenever the user clicks on it using the left mouse button. A dialogue consists of two parts: an event description and an action to be executed whenever the event occurs. Both the event and the action are described using the *pTALK* language. The action part of a dialogue is a *pTALK* statement, and can thus contain calls to standard functions as well as user-defined functions. Since both the event and the action descriptions are user defined, highly specialized dialogues can be developed easily using the *pTALK* language.

2.9. Run-time creation

As edit functionality is an integrated part of the Picasso-3 system, graphical objects can be manipulated at run-time. This can be used in applications where a dynamic data structure is represented graphically, and hence the structure of the graphics will change as the data structure changes. A typical example application is a network application where the number of nodes and the connectivity between nodes change over time. By using standard functions in *pTALK*, new graphic network nodes and connection lines could easily be created to reflect the changes in the network application. Another application area where run-time manipulation of graphics is needed is in editor-like applications. The Picasso-3 graphics editor is itself an example of an application that uses the run-time editing functionality provided by the Picasso-3 Run-Time Manager.

2.10. Reuse of Interface Components class concept

User interfaces are often built to depict real world components. The Picasso-3 class concept allows the developer to model these components in the interface. The components (valves, pumps, multiplexers, etc.) are often drawn as a combination of simple shapes. A valve can be represented by a drawing composed of a polygon, a line, and an ellipse.

Using the class concept offered in Picasso-3, instead of drawing each component from scratch each time can optimize the design and maintenance of user interfaces.

Any changes made to the design of a class will be reflected in all instances of it, thus reducing maintenance effort.

2.11. Class behavior

A real world component often has state information and a behavior associated with it. A valve has a state: it is either open or closed. It also has a behavior: it can be opened or closed. In the Picasso-3 system, the state information for a class is stored as attributes of the class. The component's behavior is also part of the class, and is implemented as pTALK functions and dialogues. The fact that not only the appearance of the real world components, but also the state and behavior associated with the components are part of the classes, is one of the object oriented features of the Picasso-3 system.

A Picasso-3 class is composed of simple shapes such as rectangles, circles and lines, and of attributes, functions, and dialogues. A class can also contain instances of other classes.

2.12. Reuse of classes

A class can be designed independently of its surroundings, making it possible to reuse components in different user interfaces, and different parts of a user interface. A class can be viewed as a small user interface in itself, and this is taken advantage of in the Picasso-3 system; classes are constructed in the same way as ordinary user interface pictures, i.e. using the interactive environment provided by the Picasso-3 Graphics Editor.

Classes can be stored in libraries for reuse. Preferably, classes describing related components should be stored in the same library. To make libraries independent of context, other interface elements can be stored in them, such as colors, fonts, and functions.

Another use of libraries is to easily change the appearance of interface components in an application. Switching between libraries containing classes with identical names can do this. This can be useful if an application needs to conform to different standards for the look and feel of components. An example could be a picture containing instances of two classes called 'Valve' and 'Pump'. These classes could be stored in a library named 'ISO_lib'. Equivalent components could also be stored in a library called 'ANSI_lib'. Simply switching libraries can then do switching between the two standards.

2.13. Integration and Communication

Picasso-3 supports several mechanisms to enable coexistence with other software components in order to create a complete integrated system. Integration mechanisms fall into two categories, namely integrating by communication, and integrating user interfaces. The first category deals with message passing in a computer network, while the second deals with cooperation in a graphics environment.

Since Picasso-3 encourages distributed systems, communication is a keyword in the integration process. Picasso-3 allows other programs to communicate with a user interface through an Application Programmer's Interface. The API is a routine library that puts the functionality of Picasso-3 within reach of any C-callable program.

Data sharing is important in a distributed system. Data has a source, which is the program that owns the data and controls it. However, the data is typically needed in other programs. The data generated in an acquisition system must, for example, be available to a separate program for analysis, and to the user interface system for presentation.

A program containing variables can make them available to the user interface by calling selected API routines. Variables can be simple floating-point numbers, integers, or complex record structures. After variables have been declared to Picasso-3, they can be referenced in any pTALK expression as part of the user interface logic.

The Picasso-3 API hides the error-prone process of packaging and interpreting bytes, which all too often becomes a major activity when programming a message based system. When a variable is declared to Picasso-3 using the appropriate API calls, the API routines take care of sending and interpreting the necessary messages. Whenever a variable is changed by a pTALK expression, the new value must be sent from the Run-Time Manager to the application program. When the program changes a variable, the new value must be sent to the Run-Time Manager.

In addition to data sharing, there is often a need to issue commands to other programs. The goal is to make the program look and feel like one integrated system, even though they are actually separate programs running on different computers. Calling remote procedures, i.e. routines in other programs, is one way of achieving this, and Picasso-3 supports this in both directions.

An application program can call a pTALK function, and actually execute any legal pTALK expression or statement. Since pTALK is an interpretative language[†], the source code can be built at runtime, and even be entered by the end user.

Routines in an application program can be called from within a pTALK expression. This requires that the application program declares the routines to Picasso-3 prior to a call, in much the same way as with variables. This means that the user interface can easily issue commands to a program, for example to tell a database system to save its data to file.

Calling API routines is trivial as long as the application program's source code is available. When we want to integrate with another commercial product, a different strategy is required. Unfortunately there is no accepted standard for high-level inter-process communication within the UNIX world, and a program cannot possibly support the protocols of all other available systems. A solution is therefore to put a filter program between the two systems, which translates between the two protocols. The filter program will typically include both systems' different API libraries. This strategy can be used to integrate Picasso-3 with commercial products such as database management systems, spreadsheets, or Geographical Information Systems (GIS).

Picasso-3 communicates using TCP/IP, and supports both synchronous and asynchronous message passing.

[†] Although pTALK is interpretative, the source code is still compiled transparently within the Run-Time Manager to increase execution speed

2.14. File Formats

Binary files

In most software systems there is a need to save and load the whole system or parts of it. This is done in Picasso-3 using *binary* files. Applications are usually built step by step. That is, starting from scratch gradually building up to a complete application.

The process of saving this application, or just parts of it, would require that only those parts that are actually changed during a work session are saved. Instead of saving an application as one big chunk, we have defined four levels of saving:

- Global application data, such as resources, global attributes and functions. The extension for these files is **.pctx**
- Libraries, containing information global to the library as well as graphical primitives and definitions of classes. The extension for library files is **.plib**
- Data, known both to the RTM and external processes, is saved in process database files. Extension for data files is **.pdat**
- Pictures together with relevant data are stored in files with the extension **.ppic**

Binary files are generated from the RTM by selecting the save choice in the file menu of the Graphics Editor.

Tdoc file format

Reading the binary file format is easy for machines, but not for human beings. Therefore, we also have a human-readable file format in Picasso-3. The extension for these files is **.Tdoc**.

The documentation file format is generated from the RTM by selecting the documentation choice in the file menu in the Graphics Editor. *Tdoc* files can also be produced using ones favorite text editor like *vi* or *emacs*.

pcc

The human-readable *Tdoc* file format can be converted into a binary format using a conversion tool called *pcc*, which is supplied as part of the Picasso-3 system.

From the RTM, choosing the save item from GED's file menu can produce binary files. Choosing the document item can produce *Tdoc* files.

In Picasso-3, as in real life, there is more than one road leading to Rome! *pcc* and the *Tdoc* format is one example of this. Besides using *pcc* and the *Tdoc* format, objects can be added to the application using the Graphics Editor, that is the most common method, and by inter-application communication via the Application P

3. Differences between MAAP-FUGEN-GRAAPH and Picasso-3

The differences between the MAAP-FUGEN-GRAAPH (MFG) and MAAP-PICASSO (MP) GUI can be concluded from the descriptions given in the previous chapters. The main differences are summarized here:

<i>Feature</i>	<i>MAAP-FUGEN-GRAAPH</i>	<i>MAAP-PICASSO</i>
<i>Program execution</i>	Numerical simulator and GUI are integrated	Numerical simulator and GUI are decoupled
<i>GUI dynamics</i>	Colors and shapes	Colors, shapes, numbers and texts
<i>GUI modification by user</i>	Not possible	Possible, even online during execution of simulator. Graphical editor and textual input are available
<i>Interaction with other software</i>	No, only indirectly through text files	Yes, provided the other software communicates with Picasso RTM
<i>2 byte (Japanese character) display</i>	No	Yes
<i>Number of windows</i>	1, plus an separate xy-plot program	Unlimited
<i>Robustness</i>	The simulator and GUI are the same code	Independent execution, crash in one does not influence the other
<i>Scalability, compatibility</i>	New versions of the GUI must be provided whenever MAAP code is updated	The GUI is independent of MAAP code version. Updates of the GUI is easy thanks to the object-oriented developing environment.
<i>Portability</i>	New versions of the GUI must be provided if MAAP is ported to new hardware.	Since the GUI is expressed in Tdoc language, it can be ported directly to any hardware platform that is supported by Picasso-3 (SUN, HP, Windows NT etc.)
<i>CPU performance. Test case: 100 sec. transient time simulation of a MLOCA accident.</i>	533.9 sec real CPU time 97% CPU load	423.7 sec. real CPU time 62% CPU load

3.1. Simulator and GUI integration.

The MFG GUI is completely controlled by the MAAP executable. In fact, there is only one executable program, which has to perform the numerical computation of MAAP itself, plus the update of the GRAAPH graphics. The MG GUI, on the contrary, is completely de-coupled from the execution of MAAP. This means that the user can do modification of the GUI during running of the MAAP code, and further that the numerical workload of the MAAP code is significantly reduced. Since the MAAP code and the Picasso-3 very well can run on different workstations, this will contribute to a faster execution of the MAAP code. This concept is illustrated below:

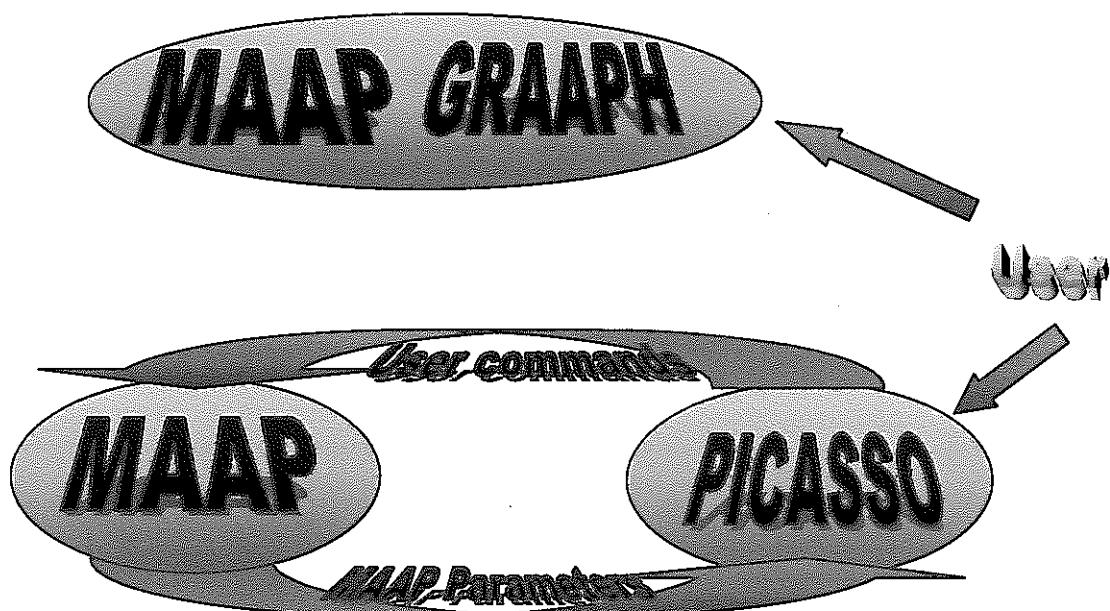


Figure 3. System layout for the two alternative GUIs.

3.2. Modifications of the GUI

An efficient User Interface to any computer code should, to as large as possible extent, be tailored to the specific users needs and preferences. The MAAP code is designed to assist advanced scenario evaluations and parametric analysis of severe accidents at the Fugen NPS. A flexible and tailored GUI is therefore of large importance for the usability of the MAAP code, and it should be possible for the user himself to modify the GUI to fulfill his requirements and preferences.

Modification of the GRAAPH GUI involves a substantial amount of modification of the MAAP code itself, and can probably only be performed by skilled MAAP programmers from Hitachi and Fauske.

The Picasso-3 GUI, however, can easily be modified and immediately tested (even during the execution of MAAP) by programmers without detailed knowledge of the MAAP code details.

3.3. Interfacing with other software

As mentioned above, the MFG is a totally integrated and autonomous process, which means that the only interface with the environment is through the GUI or through produced text and binary files. That implies that whenever the user wants to interface to other software, some filter program must be written which takes the input or output files from MAAP and produces input files to the other software. This is in fact being done under Windows, where a filter program, which takes MAAP output data files, produces files in Windows Excel format.

Using Picasso-3, however, connection to other applications such as Excel, GNUpot, Word, HTML-code etc. can easily be developed. Since the data of interest already are produced by MAAP and are available for all clients connecting to the RTM, any application can directly utilize these data. That is, given the application can communicate with the Picasso RTM.

3.4. Portability

As computer hardware rapidly is updated and changed, any software with expected lifetime of more than a few years should be prepared for hardware and operating system portability.

In the case of MAAP and MFG, completely new versions of both the simulator and GUI must be delivered from Fauske. New versions of MAAP/MARS will be developed for Windows NT, and this requires development of a completely new GUI (based on VisualBasic).

On the contrary, the Picasso-3 GUI, expressed in the pTALK language, is portable between platforms. Given that the necessary Picasso software is installed on the various hardware, the GUI is then completely portable between the hardware and operating system versions.

3.5. Japanese characters

Support for Japanese characters (or any 2 byte character system) is not implemented in the MFG. The Picasso-3 GUI can present 2 byte character strings, such as Japanese characters. It is easy to implement functions for language “toggling”.

4. Components of the MAAP-PICASSO GUI

A new GUI for Fugen NPS has been developed using Picasso-3 as the software platform. This chapter describes the various components of this new GUI and how it is operating together with the underlying MAAP-FUGEN simulator.

4.1. Modifications of the original MAAP code

Since the MAAP code was not designed for interaction with any other application than its own GUI, some modifications of the code are required. The modifications are done in the MAAP FORTRAN code provided by Fauske. Changes are basically disabling ("commenting out") some subroutine calls, and introducing other subroutine calls. A few new subroutines are written in C, and these are called by the FORTRAN main program. A new "Makefile" has been made to make the proper compilation and link calls.

The key points in the modifications are as follows:

- Disable the original GUI. This is simply done by removing subroutine calls.
- Declare communication with the Picasso RTM. This is done by letting the main program call one C subroutine, which sets up the communication.
- Change of the main loop control. For scheduling and timing reasons it has been found more efficient to let Picasso control the main iteration loop of MAAP. This is done by disabling the looping in the main MAAP program, but let Picasso make repeated call to a FORTRAN subroutine which contains the main loop body.
- Preparation and transmission of the MAAP computed parameters. This is handled by a set of C subroutines, which communicates with the FORTRAN Common Blocks where these parameters are stored.
- Receiving and handling user interactions. This is done by receiving data from the Picasso RTM, updating some common parameter vectors, and letting MAAP handle the changes.

The MAAP FORTRAN files, which are modified, are listed below:

Name	Description	Modifications
matr.f	Main program	<ul style="list-style-type: none"> • EXTERNAL declaration for the C subroutine and parameters used for the Picasso call. • Call of the C subroutine INIT_PIC. Argument is the user defined total simulation time; return value is a status flag. If the status flag equal 0 (everything ok), jump to the closing procedures, else proceed with a "normal" MAAP execution (not using the Picasso GUI). <p>Note that the INIT_PIC subroutine embeds</p>

		all Picasso interactions <u>and</u> the main loop.
<i>picloop.f</i>	Implementation of the PICLOOP subroutine.	A new subroutine, which contains the main loop, body. This body is an exact copy of the main loop body as found in <i>matr.f</i> . This subroutine is called by Picasso as the "new" main loop.
<i>pic_api.f</i>	Contains a modified APIC1(..) function of MAAP.	This routine normally provides the interface between MAAP and GRAAPH. The modifications includes: <ul style="list-style-type: none"> • declaration of three new COMMON blocks with data used by the Picasso interface • removal of the call to WMOVIE() (the GRAAPH interface)

Table 2. Modified MAAP FORTRAN files.

The remaining communication between MAAP and Picasso is taken care of outside of these routines, and does not in any way modify or interfere with the MAAP code or execution of MAAP itself.

4.2. The communication between MAAP and Picasso.

The communication routines of MAAP and Picasso are implemented in a set of C structures and functions. All are declared in the header file *picasso.h* and implemented in the three files *picasso.c*, *parameter.c* and *compartment.c*.

The communication is basically performed as follows:

1. Interface with the Picasso-3 RTM is established.
2. A set of *subscribed* MAAP variables are defined. For computation and communication load purposes is only a small subset of all available MAAP parameters (more than 20.000) communicated between MAAP and the GUI. The subscribed parameters include:
 - time data; elapsed and total transient time.
 - flags; a vector of dimension 350 which is used for updating statuses of graphical shape objects such as pumps, valves and pipes. These values are used both ways; updating the status of the GUI object; and for sending user interactions back to MAAP.
 - core node data; an array of dimension 8x20x10 defining the temperature and mass fraction of the nodalized core elements.
 - water and corium levels; a vector of dimension 350 defining the normalized water and corium levels.
 - interrupt controls
 - information and title strings
 - compartment data; an array of dimension 4x40 which defines up to 4 different parameters for each of the nodalized compartments
 - scalar parameters; and vector of dimension 4 which defines up to 4 scalar parameters to be trended in the trend window of the GUI.

3. The main loop is started which performs the following tasks:
 - Update time
 - Checks for possible user interactions
 - Calls the main MAAP loop body
 - Updates the subscribed variables
 - Sends the subscribed variables to the GUI
4. Continues with step 3 until the terminal time has been reached, or the user terminated execution through the GUI.

The C files and their contents are described in the table below:

Name	Contents	Description
<i>picasso.h</i>	Header (declaration) file	<ul style="list-style-type: none"> • Includes the necessary system and Picasso API header files. • Declares some useful constants, such as color codes, maximum values etc. • Declares C structs for more "object oriented" data structuring. • External declaration of C functions and FORTRAN subroutines to be called from C.
<i>picasso.c</i>	init_pic()	Initializes the Picasso-3 RTM communication.
	connectToPicasso()	Defines all parameters and structures to be communicated between MAAP and the RTM
	disconnect()	Called when the connection terminates.
	repeater()	<p>Routine called for each "iteration" (loop call). This routine does the following:</p> <ul style="list-style-type: none"> • updates the current transient time • checks if the user has paused execution • calls the MAAP main loop body (FORTRAN subroutine PICLOOP()) • calls the MAAP API update subroutine APIC1() • computes the updated core node width (normalized) and colors (according to a color code) • computes the nodalized compartment colors

	<ul style="list-style-type: none"> • checks if the user has change parameter to be presented as either compartment variable or in the trend plots^{\$} • informs the RTM about the changed parameters • prints some information to standard output. 	
printOutput()	Print some execution status to standard output, such as elapsed real and transient time, expected remaining time, number of iterations etc.	
restart()	Called if the user has halted the execution and is restarting it.	
halt()	Called if the user halts the execution.	
setupCompVar()	Defines the nodalized compartment variables to be communicated to the GUI, i.e. <i>subscribed</i> to. The variables are represented as vectors of dimension 40 (of which only 13 compartment nodes are active).	
setupMaapVar()	Defines a vector (dimension 4) of variables to be trended (<i>subscribed</i>) in the GUI. The user can choose to trend <u>any</u> four of all available parameters in MAAP (of which there are 3.920 - many of them vectors and matrices).	
Parameter.c	listCommonBlock()	Lists all the common blocks defined in MAAP. Utilizes the MAAP FORTRAN subroutine COM()
	listParameters()	Lists all parameters used in MAAP. Utilizes the MAAP FORTRAN subroutine COMVAR()
	listScalars()	Lists all scalar parameters used in MAAP. Utilizes the MAAP FORTRAN subroutine VARM1()
	listCompVector()	Lists all compartment node parameter vectors used in MAAP. Utilizes the MAAP FORTRAN subroutine VARM1()
	getVariable()	Finds an internal MAAP variable and its value identified by name. Utilizes the MAAP FORTRAN subroutines COMVAR() and COMVAL()

^{\$} These features are not part of the MFG, and will be described later.

	updateMaapVar()	Updates the <i>subscribed</i> variables. This is done by calling the COMVAL() subroutine.
	resetMaapVar()	Called when the user changes the subscribed variable to another MAAP parameter.
<i>Compartment.c</i>	getVector()	Gets the compartment data vector.
	resetVector()	Resets the compartment data vector.
	updateVector()	Updates the compartment data vector.
	setLimits()	Computes the min/max values for the compartment data vector.

Table 3. C files and functions included into MAAP.

4.3. Extension and modification of the code

It should be noted here that not all parameters necessary for the dynamics in the GUI has been implemented, in particular the control rod positions. However, to include these data or other parameters are relatively straightforward. To do this, the following steps should be performed:

- Declare a suitable C data structure for the parameter to be communicated to the GUI. For the control rod position this could be a vector of N_{rod} normalized float values. Declare this in the *picasso.h* file.
- Make an Picasso *record* or *array* containing these values using the *PfBeginRecord()* or *PfCreateArray()* which points to the declared structures. Do this in the *setupMaapVar()* function in file *picasso.c*.
- Find in the MAAP FORTRAN code the name of the parameters to be communicated. This may imply some "detective" work by guessing parameter names and browsing through source codes and comment strings, unless you get some hints from the MAAP code developers.
- Use the MAAP FORTRAN subroutines COM(), COMVAR() and COMVAL() to retrieve the identifier and values of the parameters. This can be done in the *updateMaapVar()* function in *parameter.c*.
- Inside the main loop, update the Picasso variables with the values from MAAP and inform the RTM about the updates. This should be done in the *repeater()* function in *picasso.c*.
- Finally; update the GUI to receive and act corresponding to the new parameters.

4.4. The Real Time Manager Data Base (shared variables)

The Picasso-3 Real Time Manager is the communication central between the MAAP simulator and the GUI. All parameters that is communicated between these two *client programs* has to be declared by the RTM. The RTM contains, in addition, a database of resources used by the GUI, such as definition of colors, fonts and shapes. A specific resource file is defined for this application and has to be loaded into the RTM at startup. This file contains a pointer to a variable declaration file.

Name	Description	Contents
<i>maap.pctx</i> / <i>maap.Tdoc</i>	RTM resource and startup instructions file	Contains a list of resources such as colors, shapes and fonts. It also contains definitions of the main windows in the GUI and the instruction for stating up the GUI.
<i>MaapData.h</i>	Variable and C/Tdoc struct declarations	Contains definitions for the shared parameters between MAAP and GUI. Should be in accordance with the <i>picasso.h</i> header described above**.

Table 4. The RTM resource and database files

** Ideally, these two files should be merged into one header file. However, the current Picasso Tdoc version and standard C is not 100% compatible in this respect. Tdoc does not allow some standard C declaration such as compiler instructions (#include and #define). On the other hand, C does not allow initialization of variables.

5. The MAAP-PICASSO Graphical User Interface (MG)

The MG consists of a set of windows and menus, which offers the same features as the MFG described in chapter 1.2. ^{††}. In addition, the MG contains a number of features, which is not offered by the MFG. This chapter describes the features and usage of the MG.

It is important to stress here again that the GUI operates independently of the MAAP simulator. This means that we can start the GUI before the simulator is started.

Invoking the Real Time Manager with the maap.pctx resource file starts MAAP-PICASSO[#]:

```
{fteshp01} maapuser> cd $MAAPHOME
{fteshp01} maapuser> rtm -r picasso/maap.pctx &
```

5.1. The main window

The main window as illustrated below will automatically pop up.

The window in Figure 4. consists of three frames;

- the top frame provides a set of pull-down menus, and a status text area. Since we have not yet started the MAAP simulator, the status text reads "Maap not started".
- the main process layout frame containing the Fugen ATR picture. Since the simulator has not started, and thus no parameter values are set, all colors and shapes are set to their default values. This frame can also display other plant pictures, such as BWR and PWR.
- the zoom and scroll frame. This frame provides buttons and sliders for zooming and scrolling the main Fugen picture.

By default, all menu texts and picture component texts are displayed with Japanese characters. The language can be toggled between Japanese and English by using a dedicated menu option.

^{††} Some specific features, such as control rod positions, are, due to time limitations, not implemented. However, as explained above, this is a straightforward task.

[#] It is assumed that the environment variable MAAPHOME has been defined to point at the directory where MAAP-PICASSO is installed and that the user has set up his PATH variable accordingly.

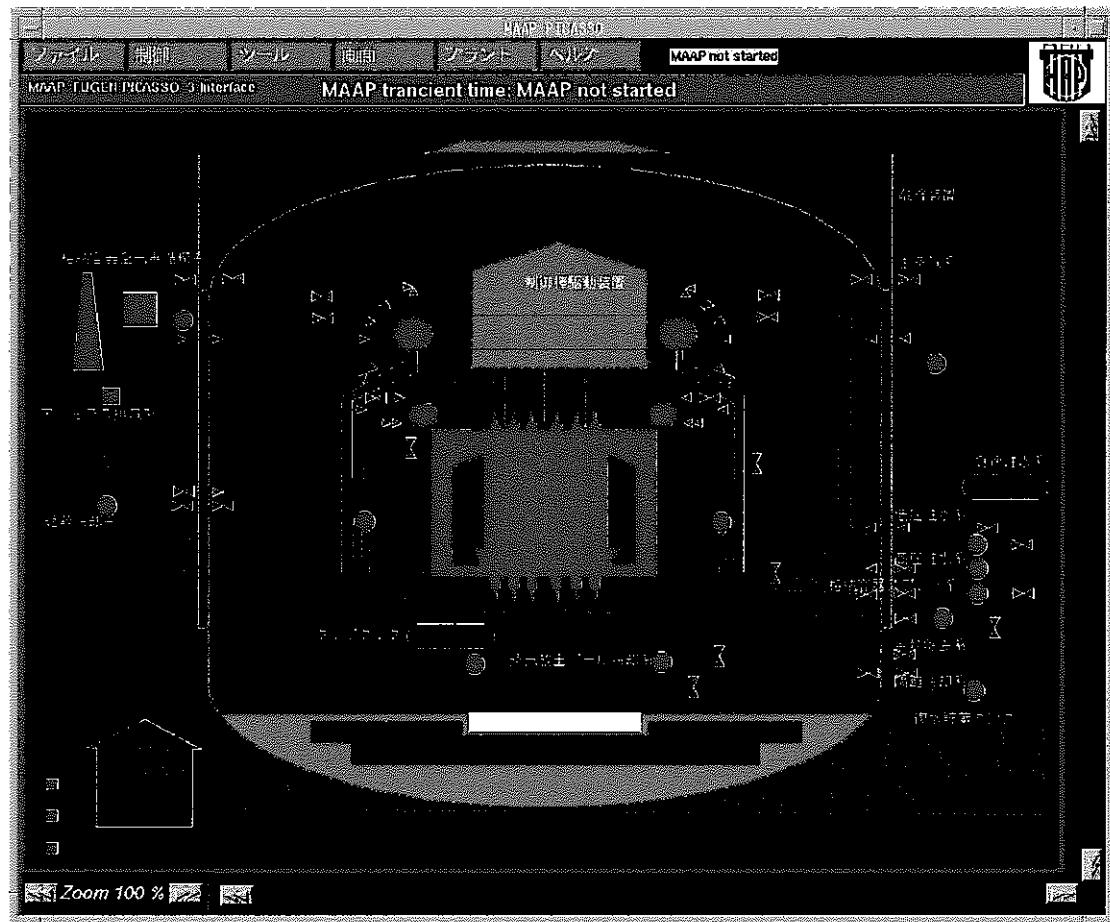


Figure 4. The main window at startup of MAAP-PICASSO

5.2. Starting the MAAP simulator

After starting the GUI the MAAP simulator should be started. An attempt of starting the simulator before the GUI will fail because the Picasso communication routines in the simulator will fail to find the RTM.

The MAAP simulator starts up as under a normal MFG run, but then initiates the Picasso RTM connection, creates the parameters to be communicated and then goes into a "wait" state. If the user chooses "Run" from the "Command" menu, the simulator goes into the main loop.

For each 20th iteration, some status information is printed to the standard output, as indicated in the text box above. This information is:

- Real time: the actual CPU time consumed
- MAAP time: the simulation transient time
- MAAP CPU usage: the percentage of the CPU usage used by the MAAP simulation. The remaining percentage is the usage for the Picasso communication.
- MAAP/Realtime: The *speedup* from realtime for the MAAP simulator. A number less than 1 shows that MAAP is slower than real time.

- estimated time left: is given in hours:minute:seconds. It should be noted that this value is computed based on the current speedup, which improves significantly as the transients are smaller and time steps longer.

```
[fleshp01] maapuser> cd $MAAPHOME/test/output
[fleshp01] maapuser> maap mloca.inp fugen.par

MAAP-FUGEN 0.01 FEB 14, 1997
MAAP PICASSO

Input file ..\mloca.inp opened for reading
Input file name used for this run is MAAPCA
Log file mloca.log opened for writing
Parameter file ..\fugen.par opened for reading
Input parameter file ..\FUGEN.PAR read

Initial processing finished, beginning code execution.

Calling PICASSO init program
Initialized PICASSO-3 API process "maap"
Entering the main loop
Creating PICASSO-3 API variables
Defined vector TGRB [K] (40.00 -> 290.00)
Defined vector MH2RB [KG] (0.00 -> 25.00)
Defined vector PPSTRB [PA] (0.00 -> 25000.00)
Defined vector MFSTRB [I] (0.00 -> 0.25)
Defined variable TGRB(13) value 313.00 [K]
Defined variable MH2RB(13) value 0.00 [KG]
Defined variable PPSTRB(13) value 3629.60 [PA]
Defined variable MFSTRB(13) value 0.02 [I]

*** Iteration 1
Real time: 3.02 - MAAP time 0.00 [sec] (0.00%), MAAP CPU usage 97.98%
MAAP Realtime: 0.00 [sec], estimated time left 4:58:30.00

*** Iteration 21
Real time: 22.10 - MAAP time 1.23 [sec] (0.12%), MAAP CPU usage 99.70%
MAAP Realtime: 0.06 [sec], estimated time left: 4:57:52.75
```

After starting the MAAP simulator, the main window gets real values and automatically updates the colors and shapes according to the parameter values provided by the MAAP simulator. The main window changes as illustrated in Figure 5.

The top frame indicates the new status. The main frame shows the continuously updated colors and shapes and the current transient time.

This image has a more detailed core node temperature and mass fraction discretisation than the standard MFG. The view is otherwise identical to the MFG.

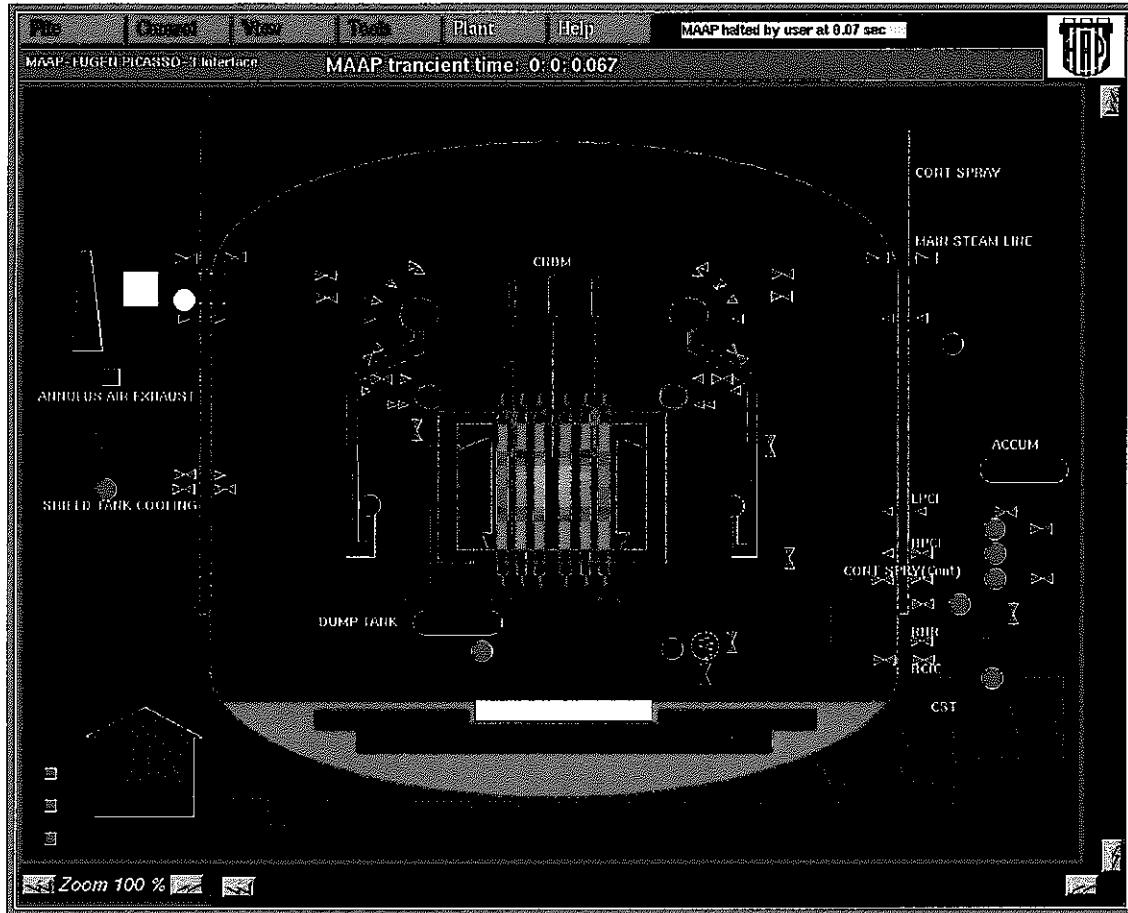


Figure 5. The main window during MAAP simulation

5.3. The main window Menus

The main window contains a set of pulldown menus. These represent the following tasks:

- **File:** currently not implemented, but can provide file functions (save, restore, movie playback etc.) and an exit button.
- **Control:** buttons for starting, halting and restating the MAAP simulation. It is required that the user halts the execution before interactions with clickable objects such as pumps and valves.
- **View:** this menu provides a set of toggle buttons for showing and hiding graphical components. For example can all piping be hidden, or hydrogen concentrations can be shown or hidden. All graphical components are classified into 13 categories of which each visibility can be toggled.
- **Tools:** contains buttons for various tools such as:
 - language toggle
 - showing compartment parameter window
 - showing trend window
- **Plant:** displays the graphical picture of other nuclear power plants provided by Fauske. This is purely for demonstration purpose, as there has not been established any link between these pictures and the numerical MAAP simulators for these plants. The purpose was to demonstrate the flexibility of

the program that translates the MFG picture files into Tdoc format. This will be presented later.

5.4. The Compartment window

By selecting **Compartment** from the **Tool** pulldown menu, a new window pops up, as illustrated below:

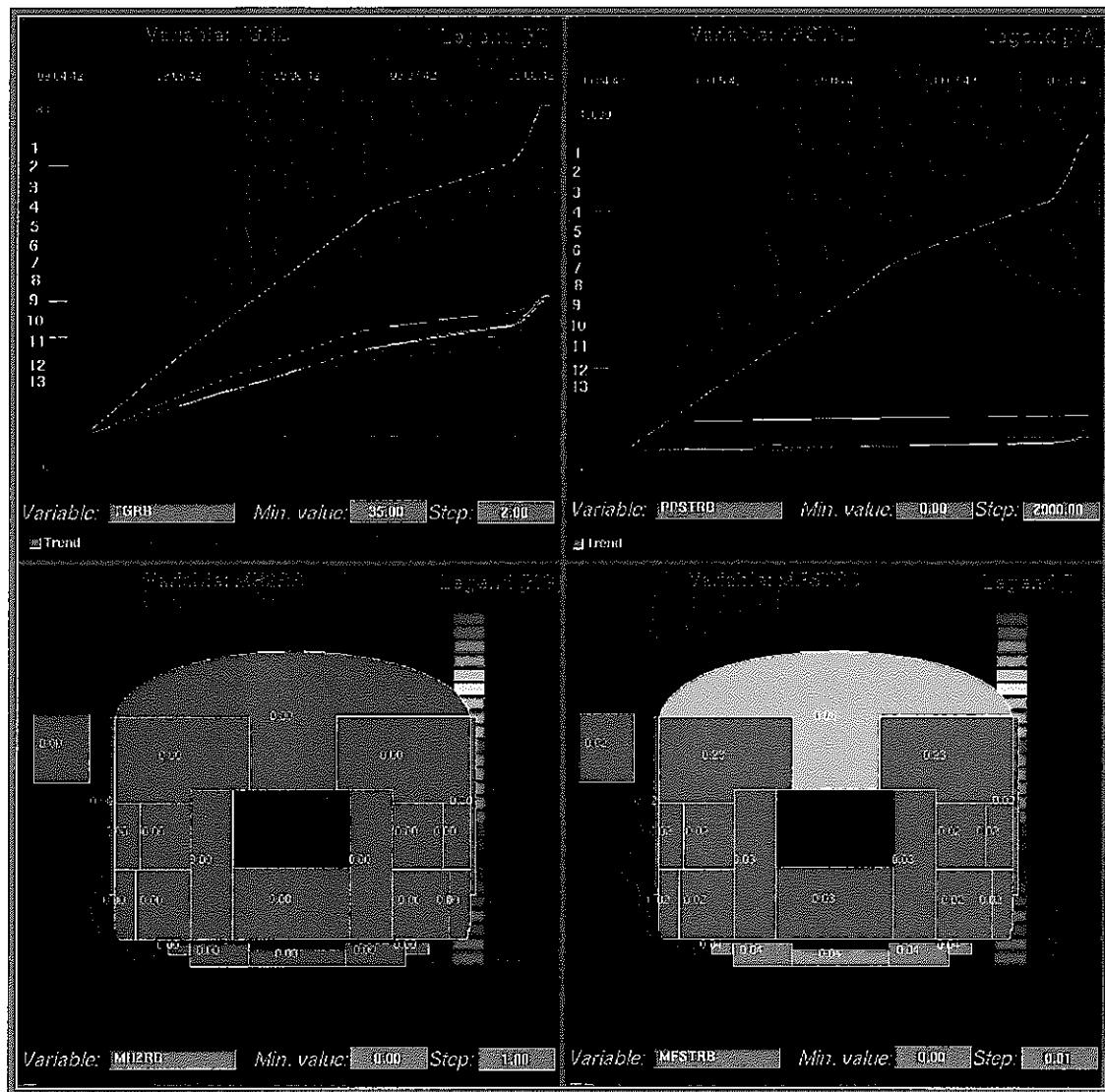


Figure 6. Compartment window

The compartment window is divided into four frames, each representing a variable vector. Each vector contains variable values for each of the 13 nodalized compartment areas of the Fugen NPP. The variable (parameter) name as it is defined in MAAP is displayed, together with the parameter dimension, at the top of each frame. The smallest and largest parameter vector element is displayed at the bottom of each frame.

The parameter vector values may be displayed in one of two modes, and the user can toggle between these modes by clicking the **Trend** button:

- **Trend plot mode.** Trending the variables in a time-series plot. Each variable is given an unique color.
- **Color code and numerical value mode.** In this mode the Fugen NPP compartments are displayed, and given a color according to the value of the variable presented. There is a color code scale displayed to the right of each frame. In addition, the numerical value of the parameter is displayed.

When this window pops up, four default compartment parameter vectors are displayed. However, the user can change this parameter any time simply by entering the name of the variable into the **Variable** input field. The parameter name must have been defined in MAAP^{§§}. When a new parameter name is entered, this new name is transferred to the MAAP simulator and checked for validity. If the parameter is not defined in MAAP, or if it is not a compartment node vector, then the old parameter is restored and the user is informed that the name is invalid. However, if the name is valid, then MAAP changes focus to this variable and new values are transferred to the GUI for presentation. However, the trends are not cleared, so the curves may have a notable "jump" when the new values are trended.

There is not automatic re-scaling of the Y-axis when some values exceed the scale. However, the system recognizes that some values exceed the scale and a message suggesting manual re-scaling is displayed. Instead of entering minimum and maximum Y-axis limits, the user can enter minimum value and an increment value. There are a total of 20 increments (discrete colors) so the increment value is 1/20th of the difference between maximum and minimum value.

5.5. The Variable Trend window

By selecting **Trend** from the **Tool** pulldown menu, a new window pops up, as illustrated below:

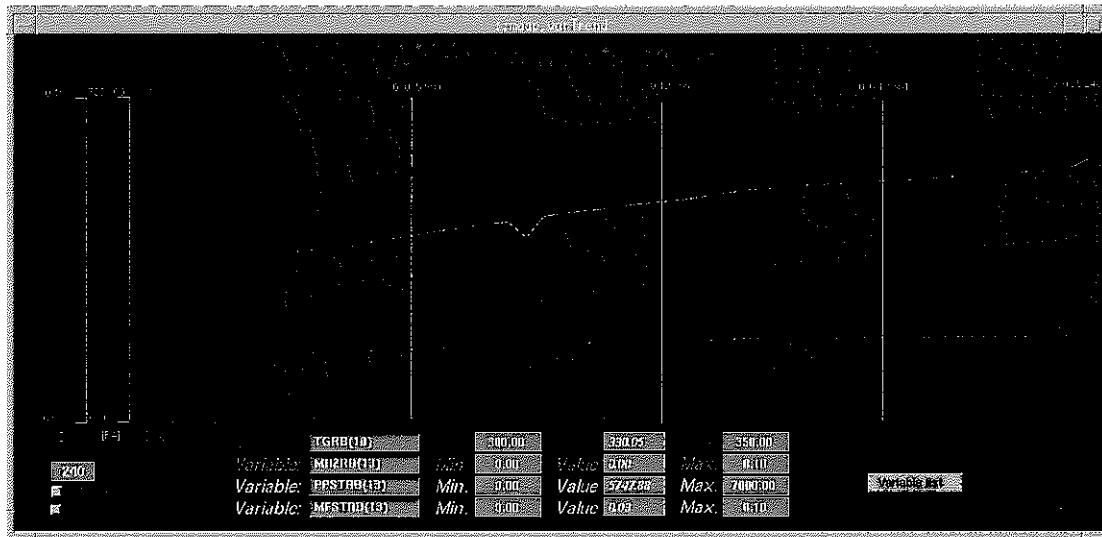


Figure 7. The normal trend window

^{§§} Finding parameter names in MAAP can be a bit of an detective work unless one knows the MAAP code in detail. It includes searching MAAP FORTRAN source files and documentations.

The trend window consists of a trend area and a parameter input area. The trend area displays four parameters simultaneously. Each parameter is associated with a unique color. Individual Y-axis scales are displayed to the left.

The parameter-input area provides the user methods for changing the parameters to be displayed. As for the compartment vector parameters described above, the user can enter any valid scalar MAAP parameter. If the parameter is an element of an vector or matrix, the vector/matrix index must be entered, for example the temperature in compartment node 10 as TGRB(10). A future enhancement would be to provide a method for browsing all MAAP parameters (more than 3000 parameters and 20.000 scalars), and let the user select the parameter graphically.

Entering a minimum or maximum Y-axis scale value can alter the scale for each parameter. The current value of the parameter is displayed between the extreme points.

The time scale is given as the MAAP transient time. However, since the time step in MAAP changes during execution, the time scale is not linear, i.e. all iteration point are plotted with same distance, even though the transient time between them varies. The user can change the overall time span.

By clicking the Polar Plot button, the trend frame changes into a polar plot mode. This mode can be useful for illustrating cyclic behavior of parameters.

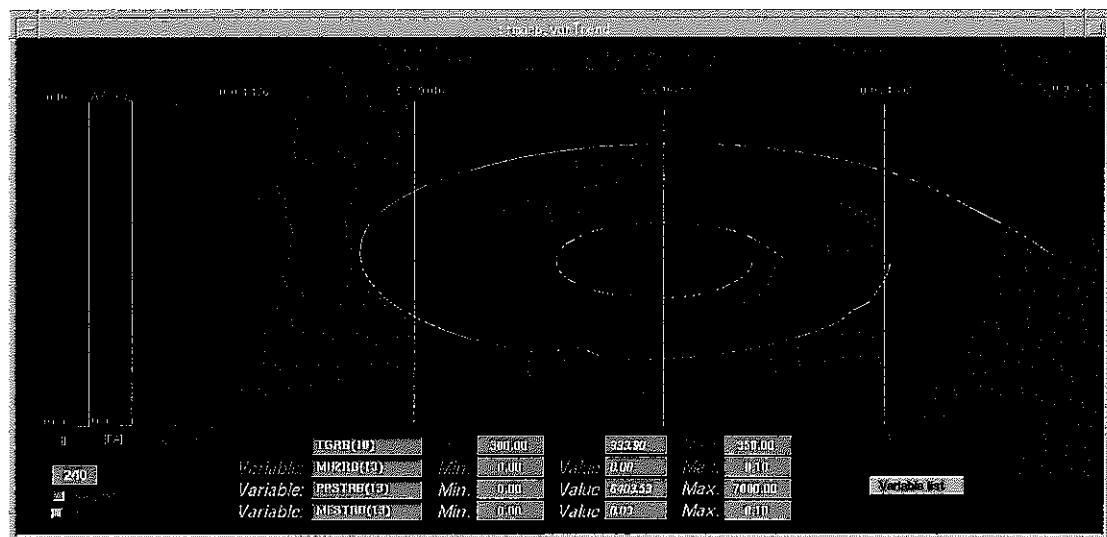


Figure 8. Polar Plot mode of the trend window

5.6. Displaying other plant pictures

For demonstration purposes, a number of other MAAP-GRAAPH power plant picture files has been translated into Picasso Tdoc format. These pictures can be displayed in the main window by selecting them from the Plant menu.

An example of a PWR picture is presented below:



Figure 9. MAAP-PICASSO applied on a PWR.

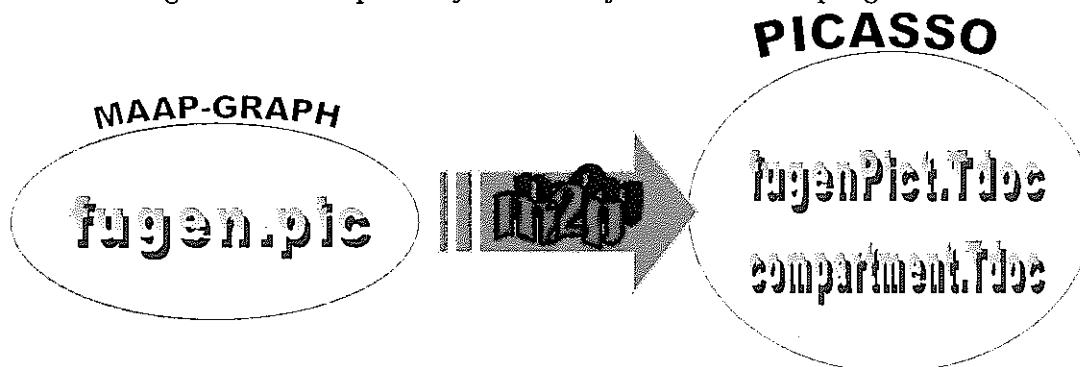
6. Automatic conversion of MAAP-FUGEN-GRAAPH picture files to Tdoc code

A small C++ program that translates GRAAPH pictures to Picasso Tdoc format, called “m2p” was made. This program does the following:

- reads the GRAAPH Picture File (for example “fugen.pic”)
- reads an additional Picasso specific definition file for the picture (“fugen.pdef”)
- writes a Tdoc file for the main picture (“fugenPict.Tdoc”)
- writes additional pictures for example for compartment parameters (“compartmentPict.Tdoc”)
- invokes the Picasso “pcc” compiler to produce Picasso code from the Tdoc files

The central part of the “m2p” program is an object class “Converter” which has methods for performing the tasks above.

Figure 10. MFG picture file to Tdoc files translation program.



6.1. Reading the GRAAPH Picture File:

The GRAAPH pictures are defined in a Picture File “<name of plant>.pic” . The first lines of the “fugen.pic” file is shown here (line numbers included):

```

1: SIZE=
2: 0-9 10.085
3: 21 921 27 34 41 49 67 82 122 125 128 159 140 141 142 143 152 154 161
4: 135 310
5: 1 77 /CONTAINMENT WALL
6: 11 16 0 0 0-1
7: 8.279 1.579
8: 8.279 5.960
9: 8.279 5.008
9: 8.267 6.099

```

Line1 and 2: The SIZE keyword (omitted in some Picture Files) defines the world size coordinates for the GRAAPH window, which all pictures relate to. Line 3: The following list of integer is, for now, not used. Line 4: Number of pictures, also not used. Lines 5+: For each picture, the following data is read:

- object number and coordinates for the line drawing of the picture. The object number plays an important role as an identifier for the so called MAAP events.
- a descriptive text of the picture. This text plays is important when classifying the picture.
- line 6: category number, and 5 parameters which is not yet used. These parameters may contain some useful information.
- a list of coordinates for the line drawing of this picture

From the information in the Picture File, Picasso Tdoc pictures can be made directly as line objects. However, by utilizing knowledge about the different categories of pictures, we can do far better than this.

6.2. Producing Tdoc picture objects:

A graphical object is in the Tdoc language, defined as in *instance* of a standard Picasso object, such as *line*, *polygon*, *circle* etc., or an instance of a user defined graphical object. In the "m2p" program we use both. For all graphical objects, a set of *attributes* must be declared, as illustrated below.

```
line pipe15 //PIPING TO MODULAR CIRCULATION SYSTEM
{
    numberOfPoints = 5;
    point[0] = (354, 572);
    point[1] = (354, 374);
    point[2] = (524, 374);
    foregroundColour = eventColor[event[36],1];
    foregroundFillColour = eventColor[event[36],1];
    x = 0;
    y = 0;
    visibility = visFlag[1];
    backgroundColour = -1;
    lineWidth = 1;
    linePattern = "SolidPattern";
    fillPattern = "SolidPattern";
    lineStyle = "SolidPattern";
    xReference = 439;
    yReference = 473;
    xScaleFactor = 1;
    yScaleFactor = 1;
    rotationAngle = 0;
}
```

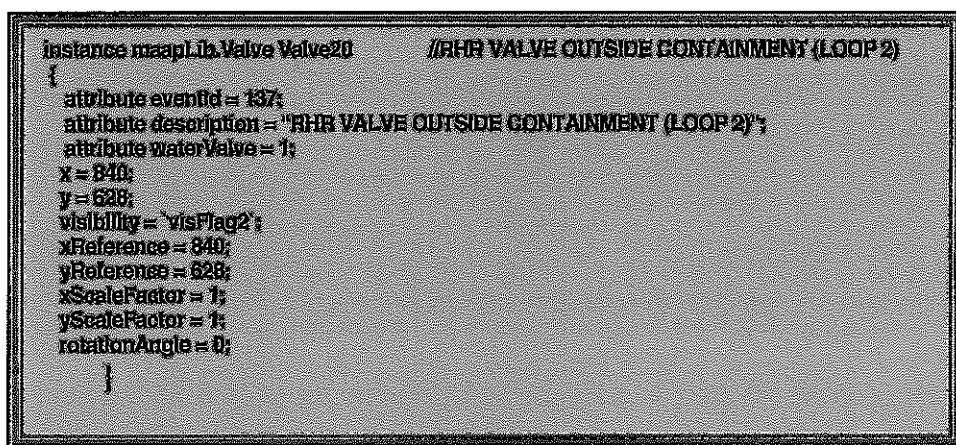
In this example, the " PIPING TO MODULAR CIRCULATION SYSTEM" is declared as a standard *line* object. The standard attributes defines the line's position (i.e. the points), its line width etc. Note that the foreground color, fill colors and visibility are linked to the Picasso variables *ievent[]* and *visFlag[]*.

The example above shows the *standard* Tdoc representation of a GRAAP picture. As can be seen, not that much functionality related to the knowledge about the picture is imbedded, except for the following (which is default for all categories except *compartments*):

- the *visibility* of the picture on the display can be toggled by changing the value of the *visFlagx* variable.
- the color of the object is changed according to the picture's *event number*^{***}.

6.3. Library objects

In addition to using the standard Picasso objects such a line, polygon, circle etc, the user can define her own graphical picture classes. These classes are stored in a library file; in this example in *maapLib.Tdoc*. This library contains special classes for pumps, valves and heat exchangers. An example of the use of a user-defined picture object:



6.4. Structure of the "m2p" program

For each pictures read from the GRAAPH picture file a special *PictObject* object (instance) is made, or rather, an instance of a subclass of *PictObject*. "m2p" reads the picture category number (ranges from 1—23) and then creates a *picture object* according to the category number. The picture objects are implemented as special *subclasses* of a generic C++ picture class. Each such class inherit the same method for reading the GRAAPH parameters (coordinates etc.), but have their own specific method for producing Picasso objects. The generic read() method simply reads the coordinates and stores them in a coordinate list object. For example, when "m2p" reads a picture which it identifies to be a pump type, it creates an instance of the special class "Pump". Class definitions can be found in the file "category.h".

^{***} Each picture in the GRAAP Picture file is assigned a unique event number which corresponds to the internal EVENT[] matrix in MAAP. MAAP uses this EVENT matrix for updates and user interaction of graphical objects in GRAAPH.

The compartment class

In our example, we need to focus on the "Compartment" class. This is implemented as shown above. Note the two statements to connect the color and text value to the shared variables declared in MAAP and in the Picasso resource file.

```

Compartment::Compartment(Converter* cd) : FlatObject(cd)
{
    carNr = COMPARTMENT;
    name = "Compartment";
    eventObj = 0;
    isWater = 0;
    noVis = 1;
}

//-----
ErrorCode Compartment::read(ifstream& inp, SDI::string d, int nc, int end, int snr)
{
    eventObj = car;
    objNr = snr;
    desc = d;
    objNr = atoi(strtok(NULL, " ")); // Extract the compartment number from the description text
    coords = new Guard(conv, nc); // Make a coordinate object
    return coords->read(inp); // Feed the coordinates
}

//-----
ErrorCode Compartment::write(ofstream& outp)
{
    auto << " polygon W" << name << objNr << " " << desc << endl;
    outp << " I" << endl;
    coords->write(outp); // Write the coordinates for this polygon
    outp << " visibility = 1;" << endl; // Always visible
    outp << " foregroundColour = compStreamColor" << objNr << "?" << endl; // Map to compStreamColor
    writeStandard(outp); // Generic method to write the standard attributes
    outp << " Y" << endl;
    return writeText(outp); // Write a text object that is centered within this polygon
}

//-----
ErrorCode Compartment::writeText(ofstream& outp)
{
    xPos = coords->xRef + 29; yPos = coords->yRef;

    outp << " text" << endl;
    outp << " I" << endl;
    outp << " format = V%2H%" << endl;
    outp << " theText = compStreamColor" << objNr << "]" << endl; // Map to compStreamColor
    outp << " foregroundColour = white;" << endl;
    outp << " theFont = font2;" << endl;
    writeStandard(outp, 2);
    outp << " M" << endl;

    return L_OK;
}

```

6.5. Making the compartment window

The compartment window, containing the compartment objects (and optionally a color legend, leading text etc.) is done by writing a specific Tdoc file, "compartment.Tdoc".

The code for doing so is shown below:

```
PErrorCode Converter::writeCompartment(SD::string fname) { // Name of the output file
    // Open file
    ofstream cout(fname);

    // Write header
    if ((status = writeHeader(cout, "compartment", 1000, 900)) != L_OK) return status;

    // Write picture objects
    for(int i = 0; i < nobj[COMPARTMENT]; i++)
        if ((status = pobj[COMPARTMENT][i] > write(cout)) != L_OK) return status;

    // Write footer
    if ((status = writeCompartmentFooter(cout, 1000, 900)) != L_OK) return status;

    // Close file
    cout.close();
    return status;
}
```

6.6. Finishing up

After modifications to some code in the "m2p" source directory, the program must be recompiled using the "make" command. By invoking the "m2p" program on the GRAAPH picture file, the new "compartment.Tdoc" file will be produced and automatically compiled by "pcc".

7. References

- (1) MAAP-FUGEN-GRAAPH User's Manual for HP/UNIX, Fauske & Associates, Inc. March 1997.
- (2) Picasso-3 User Manual. Institutt for energiteknikk, OECD Halden Reactor Project, June 1997.

Appendix 1: Directory listing and file descriptions

All files are situated in subdirectories under the MAAP-PICASSO installation directory (/users/pclund/maap).

<i>Sub Directory</i>	<i>File Name</i>	<i>Languag e</i>	<i>Lines</i>	<i>Description</i>
include	Comdata.h	C++	51	Header file for declaration of common data structures
analyze	Makefile		66	Makefile for the MAAP variable extraction and analyze program
	Analyze.h	C++	46	Declaration of analyze classes
	Main.c	C++	207	Implementation of the analyze program
m2p	Makefile		74	Makefile for the maap2picasso picture file translation program
	Category.h	C++	320	Declaration of the various picture categories in MAAP-GRAAPH
	Category.c	C++	513	Implementation of the various picture categories in MAAP-GRAAPH
	Convert.c	C++	466	Implementation of the m2p conversion program
	M2p.c	C++	78	Main file
	Maap2picasso.h	C++	117	Declaration of the classes in the m2p program
	WriteButton.c	C++	87	Methods for writing category visibility flags
maapsrc	Makefile		77	Makefile for compiling and linking the modified MAAP program
	Compartment.c	C	157	Implementation of compartment variable vector functions
	Fugen0.lib	binary	-	Library of standard MAAP files
	Matr.f	FORTRAN	336	Main MAAP subroutine
	Parameter.c	C	181	Parameter functions
	Pic_api.f	FORTRAN	1.243	Modified API functions
	Picasso.c	C	438	Main Picasso communication functions
	Picasso.h	C	126	Declaration of the common data structures and constants
picasso	Picloop.f	FORTRAN	102	Main MAAP loop body
	Makefile		71	Makefile for making the MAAP-PICASSO GUI. Calls make recursively for all subdirectories
	Maap.Tdoc	Tdoc	805	Main resource and startup file
	Maap.pctx	binary		Compiled version of maap.Tdoc

picasso/ data	Makefile		50	Makefile
	CompartVars.Tdo c	Tdoc	116	Compartment variable trend data declarations
	MaapData.h	C/Tdoc	88	Declaration of common variables
picasso/ lib	ButtonLib.Tdoc	Tdoc	408	Library of button classes
	InputFields.Tdoc	Tdoc	2.764	Library of input fields
	MotifCtrLib.Tdoc	Tdoc	3.523	Library of Motif type objects
	MotifLib.Tdoc	Tdoc	1.073	Library of Motif type objects
	Makfile		49	Makefile
	MaapLib.Tdoc	Tdoc	4.316	Maap-Picasso specific objects
picasso/ menu	Makefile		51	Makefile
	ControlP.Tdoc	Tdoc	137	Menu classes
	FileP.Tdoc	Tdoc	175	Menu classes
	HelpDiaP.Tdoc	Tdoc	470	Menu classes
	HelpP.Tdoc	Tdoc	99	Menu classes
	PlantP.Tdoc	Tdoc	193	Menu classes
	PulldownP.Tdoc	Tdoc	99	Menu classes
	ToolsP.Tdoc	Tdoc	118	Menu classes
	ViewP.Tdoc	Tdoc	509	Menu classes
picasso/ pict	Makefile			Makefile
	CompartmentPict. Tdoc	Tdoc	4.879	Compartment window definition
	MaapMain.Tdoc	Tdoc	534	Main GUI window
	VarTrend.Tdoc	Tdoc	107	Variable trending window
picasso/ plant	Makefile			Makefile
	Fugen.pdef	Text	80	FUGEN picture definition file
	FugenPict.Tdoc	Tdoc	12.060	FUGEN picture file, automatically generated by the m2p program

Appendix 2: Source codes

1997年 10月 30日 14時 23分 /users/pclund/maap/include/comdata.h ページ 1

```
*****  

File name: comdata.h  

Author: Per Christer Lund, PNC  

Creation date: November 1997  

Description: Declaration of common data structures used for transfer of data  

between applications suchs as MAAP/PICASSO, DataServer, FugenTrend  

etc.  

*****  

//ifndef COMDATA_H  

#define COMDATA_H  

/* Data structure for Compartment Parameters (13 compartments) */  

struct CompartData  

{  

    char name[32];           /* Parameter name */  

    char unit[8];            /* Dimension */  

    int indx[40];            /* Pointer index, used by MAAP */  

    float data[40];           /* List of data */  

    int color[40];           /* Color code for each compartment */  

    float minValue;          /* User set minimum color code for plotting */  

    float stepVal;            /* User set color code change increment */  

    float minValue;          /* Minimum numerical value for plotting */  

    float maxValue;           /* Maximum numerical value for plotting */  

    int iunit;                /* Unit flag (1 = SI, 2 = BR), used by MAAP */  

    int icf;                  /* Conversion factor, used by MAAP */  

    int chFlag;                /* Change flag; indicate a user interaction */  

    int failFlag;              /* Fail flag; indicates illegal user input */  

    char message[100];         /* Message or error string */  

};  

/* Scalar variable */  

struct MaapVariable  

{  

    char name[32];           /* Variable name */  

    char unit[8];             /* Dimention */  

    int indx;                 /* Pointer or channel index */  

    float data;                /* Parameter numerical value */  

    float minValue;           /* Minimum value for plotting */  

    float maxValue;            /* Maximum value for plotting */  

    int iunit;                  /* Unit flag (1 = SI, 2 = BR), used by MAAP */  

    int icf;                    /* Conversion factor, used by MAAP */  

    int chFlag;                  /* Change flag, indicate a user interaction */  

    int failFlag;                /* Fail flag; indicates illegal user input */  

    char message[100];          /* Message string */  

};  

//endif  

***** END OF FILE comdata.h *****
```

1997年 12月 25日 13時 37分 /users/pclund/tmp/wc2 ページ 1

50 85 763 data/Makefile
71 133 1147 data/Makefile~
116 931 9080 data/compartVars.Tdoc
0 320 8368 data/compartVars.ptrv
88 442 3582 data/maapData.h
85 418 3408 data/maapData.h~
408 967 7239 lib/ButtonLib.Tdoc
81 395 7396 lib/ButtonLibplib
2764 6228 51954 lib/InputFields.Tdoc
745 3109 57252 lib/InputFieldsplib
49 88 827 lib/Makefile
3523 8065 69171 lib/MotifCtrLib.Tdoc
476 2181 59092 lib/MotifCtrLibplib
3862 8673 70650 lib/MotifLib.Tdoc
1073 4196 73608 lib/MotifLibplib
4316 10058 85486 lib/maapLib.Tdoc
200 3146 72300 lib/maapLibplib
51 92 845 menu/Makefile
71 133 1147 menu/Makefile~
137 343 2610 menu/controlP.Tdoc
12 121 2424 menu/controlP.pic
175 446 3158 menu/fileP.Tdoc
10 156 3476 menu/fileP.pic
470 1111 8217 menu/helpDiaP.Tdoc
99 237 1816 menu/helpP.Tdoc
10 87 1752 menu/helpP.pic
193 489 3920 menu/plantP.Tdoc
193 489 3877 menu/plantP.Tdoc~
14 182 3356 menu/plantP.pic
99 245 1815 menu/pulldownP.Tdoc
10 97 1688 menu/pulldownP.pic
118 286 2206 menu/toolsP.Tdoc
8 112 3096 menu/toolsP.pic
509 1571 12392 menu/viewP.Tdoc
40 613 12932 menu/viewP.pic
45 79 757 pict/Makefile
71 133 1147 pict/Makefile~
4879 11796 94233 pict/compartmentPict.Tdoc
22 3738 77828 pict/compartmentPict.pic
534 1440 10996 pict/maapMain.Tdoc
103 642 14120 pict/maapMain.pic
107 305 2594 pict/varTrend.Tdoc
4 111 3588 pict/varTrend.pic
47 87 809 plant/Makefile
63 6116 124324 plant/bwPict.pic
4402 12834 103351 plant/compartmentPict.Tdoc
80 457 3323 plant/fugen.pdef
4170 11624 102246 plant/fugen.pic
12060 38823 299894 plant/fugenPict.Tdoc
226 8985 250604 plant/fugenPict.pic
52 4923 100788 plant/markiPict.pic
45 4137 101768 plant/markiiPict.pic
72 5252 106916 plant/markiiiPict.pic
54 6994 133112 plant/wicePict.pic
48 6569 123212 plant/wldryPict.pic
47210 181290 2307660 合計

1997年 09月 22日 14時 18分 /users/pclund/maap/analyze/Makefile ページ 1

```

#
# Makefile for the MAAP to PICASSO filer program "m2p" at HP UX workstations
# Written by Per Christer Lund, Feb. 1997
#
##### System configuration #####
#
# Programs and flags
# Use GNU C++ compiler
CC = c++
CFLAGS = -O
STRIP = strip

# Directories
SRCDIR = .
BKKDIR = ./backup

# Libraries and headers
LIBS = -lm -L$(HOME)/lib -lutility
INCS = -I. -I$(HOME)/include

##### End of system configuration #####
##### Program definitions #####
PROG = ana

.c.o:
    $(CC) -c $(CFLAGS) $(INCS) $<

OBJS = main.o

INCF = analyze.h

##### Make program #####
all: ${PROG}

${PROG}: ${OBJS}
    $(CC) $(CFLAGS) -o $@ ${OBJS} ${LIBS}
    ${STRIP} $@

# Dependencies
main.o: $(INCF)

# Cleaning and backup
clean:
    rm -f *~ core *.log

veryclean:
    rm -f *~ core *.log *.o ${PROG}

backup:
    cp *.c *.h Makefile ${BKKDIR}

```

1997年 09月 22日 14時 18分 /users/pclund/maap/analyze/Makefile ページ 2

#===== end of file Makefile =====

1997年 09月 22日 16時 45分 /users/pclund/maap/analyze/main.c ページ 1

```
//=====

#include <fstream.h>
#include <string.h>
#include <ctype.h>
#include "analyze.h"

#define VARDEF "varsort.lst"
#define VARDEF2 "maapvars.txt"
#define VARDEF3 "fugen.par"
#define OUTFILE "vardef.out"

void main(){
    int ivar;

    cout << "---- MAAP variable analyzer program ----\n" << endl;
    VarList* vlist = new VarList();

    // Read main def
    ifstream inp(VARDEF);
    cout << "Reading variable list " << VARDEF << endl;
    if (!inp.is_open() || !(ivar = vlist->readDef(inp))) {
        cout << "Error reading " << VARDEF << endl;
        return 0;
    }
    cout << "Read " << ivar << " vars. \n" << endl;
    inp.close();

    // Reading additional defs
    inp.open(VARDEF2);
    cout << "Reading FORTRAN code definitions (CDEF) from " << VARDEF2 << endl;
    if (!inp.is_open() || !(ivar = vlist->readCDEF(inp))){
        cout << "Error reading " << VARDEF2 << endl;
        return 0;
    }
    cout << "Updated " << ivar << " variables \n" << endl;
    inp.close();

    // Reading additional defs
    inp.open(VARDEF3);
    cout << "Reading parameter definitions from " << VARDEF3 << endl;
    if (!inp.is_open() || !(ivar = vlist->readPara(inp))){
        cout << "Error reading " << VARDEF3 << endl;
        return 0;
    }
    cout << "Updated additional " << ivar << " variables \n" << endl;
    inp.close();

    // Write
    ofstream outp(OUTFILE);
    cout << "Writing C style code " << OUTFILE << endl;
    ivar = vlist->writeC(outp);
    cout << "Written " << ivar << " vars. \n" << endl;
    outp.close();
}
```

1997年 09月 22日 16時 45分 /users/pclund/maap/analyze/main.c ページ 2

```

    delete vlist;
}

//=====

int Var::readDef(ifstream& inp){
    char foo[UNITLEN],*vindx;
    inp >> name >> com >> unit >> foo >> defval;
    if((vindx = strchr(name,'('))) {
        indx[0] = 1;
        *vindx = '\0';
    }

    return (name[0]== ' ');
}

//=====

int VarList::readDef(ifstream& inp)
{
    int icnt=0;
    char imrk = 'A';
    ivar = 0;
    while (! inp.eof() && ivar < MAXVARS) {
        vars[ivar] = new Var();
        if (vars[ivar]->readDef(inp) == 1) return 0;
        icnt++;
    /*
        if (vars[ivar]->name[0] != imrk) {
            cout << imrk << ":" << icnt << " variables" << endl;
            imrk = vars[ivar]->name[0];
            icnt = 0;
        }
    */
        ivar++;
    }
    // cout << imrk << ":" << icnt << " variables" << endl;

    return ivar;
}

//=====

int VarList::writeC(ofstream& outp)
{
    int i,id=0;
    outp << "struct MAAPvarDef {" << endl;
    outp << "    char name[" << NAMELEN << "];" << endl;
    outp << "    char unit[" << UNITLEN << "];" << endl;
    outp << "    int vector;" << endl;
    outp << "    char desc[" << DESCLEN << "];" << endl;
    outp << "};\n" << endl;
    outp << "//-----\n" << endl;
    outp << "struct MAAPvarDef maapDefs[" << ivar << "] = {" << endl;
}

```

1997年 09月 22日 16時 45分 /users/pclund/maap/analyze/main.c ページ 3

```

for(i = 0;i<ivar;i++) {
    if (! vars[i]->writeC(outp)) return 0;
    if (i < ivar -1) outp << "," << endl;
    if (strlen(vars[i]->desc) > 0) id++;
}
outp << "\n";\n" << endl;

cout << "Found description for " << id << " of " << i << " variables (" << id*100/i
    << "%)" << endl;
return i;
}

//-----

int Var::writeC(ofstream& outp)
{
    outp << "(" << name << ",\"";
    if (strlen(name) < 5) outp << "\t";
    outp << "\t\" << unit << "\",\t";
    outp << indx[0] << ",\"";
    outp << desc << "\"";
    outp << "}";
    return 1;
}

//-----

#define MDUM 1000

int VarList::readCDEF(ifstream& inp)
{
    char dum[MDUM],*tok,*desc;
    int il = 0,id = 0,idup=0;

    while(inp.getline(dum,MDUM)) {
        strtok(dum, " \n\t");
        il++;
        tok = strtok(NULL, " \n\t=");
        if (! tok || ! isupper(tok[0])) continue;
        // Search for the variable
        for(int i=0;i<MAXVARS;i++) {
            if (strcmp(vars[i]->name,tok)==0) {
                desc = tok;
                desc+= strlen(tok)+1;
                while(desc[0] == ' ') desc++;
                idup += vars[i]->setDesc(desc);
                id++;
                break;
            }
        }
    }
    return id - idup;
}

```

1997年 09月 22日 16時 45分 /users/pclund/maap/analyze/main.c ページ 4

```
//-----
int VarList::readPara(ifstream& inp)
{
    char dum[MDUM],*tok,*desc;
    int il = 0,id = 0,idup=0;

    while(inp.getline(dum,MDUM)) {
        il++;
        if (dum[0] == ' ' || dum[0] == '*') continue;
        tok = strtok(dum, " \n\t");
        if (! tok || ! isupper(tok[0])) continue;
        // Search for the variable
        for(int i=0;i<MAXVARS;i++) {
            if (strcmp(vars[i]->name,tok)==0){
                tok = strtok(NULL, " \n\t");
                desc = tok;
                desc+= strlen(tok)+1;
                while(desc[0] == ' ') desc++;
                idup += vars[i]->setDesc(desc);
                id++;
                break;
            }
        }
        return id - idup;
    }

//-----
int Var::setDesc(char* d)
{
    if (strlen(desc))    return 1;
    strcpy(desc,d);
    return 0;
}
//-----
```

1997年 09月 22日 16時 06分 /users/pclund/maap/analyze/analyze.h ページ 1

```
-----  
#include <stdio.h>  
#include <stdlib.h>  
#include <fstream.h>  
  
#define NAMELEN 30  
#define UNITLEN 10  
#define DESCLEN 100  
#define MAXVARS 3919  
  
class Var  
{  
public:  
    Var(){indx[0]=0;};  
    ~Var(){};  
  
    int readDef(ifstream&);  
    int setDesc(char* d);  
    int writeC(ofstream&);  
    char name[NAMELEN];  
  
    char com[NAMELEN];  
    char unit[UNITLEN];  
    char desc[DESCLEN];  
  
    float defval;  
    int indx[4];  
};  
-----  
class VarList  
{  
public:  
    VarList(){  
  
        int readDef(ifstream&);  
        int readCDEF(ifstream&);  
        int readPara(ifstream&);  
        int writeC(ofstream&);  
    private:  
        Var* vars[MAXVARS];  
        int ivar;  
    };  
-----
```

1997年 09月 11日 15時 47分 /users/pclund/maap/m2p/Makefile ページ 1

```

#
# Makefile for the MAAP to PICASSO filer program "m2p" at HP UX workstations
# Written by Per Christer Lund, Feb. 1997
#

##### System configuration #####
# Programs and flags
# Use GNU C++ compiler
CC = c++
CFLAGS = -O
STRIP = strip

# Directories
SRCDIR =
BKKDIR = $(HOME)/backup/maap/convert

# Libraries and headers
LIBS = -lm -L$(HOME)/lib -lutility
INCS = -I. -I$(HOME)/include

##### End of system configuration #####
##### Program definitions #####
PROG = m2p

.c.o:
    ${CC} -c ${CFLAGS} ${INCS} $<

OBJS = m2p.o \
       convert.o \
       category.o \
       writeComp.o

INCF = maap2picasso.h \
       category.h

##### Make program #####
all: ${PROG}

${PROG}: ${OBJS}
    ${CC} ${CFLAGS} -o ${@} ${OBJS} ${LIBS}
    ${STRIP} ${@}

# Dependencies
convert.o: ${INCF}
m2p.o: ${INCF}
writeButton.o: ${INCF}
writeComp.o: ${INCF}
category.o: ${INCF}

# Cleaning and backup
clean:
    rm -f *~ core *.log

```

1997年 09月 11日 15時 47分 /users/pclund/maap/m2p/Makefile ページ 2

```
veryclean:  
        rm -f *~ core *.log *.o ${PROG}
```

```
backup:  
        cp *.c *.h Makefile $(BKKDIR)
```

#===== end of file Makefile =====

1997年 09月 11日 16時 46分 /users/pclund/maap/m2p/category.h ページ 3

```
public:
    HxCoil(Converter*);
    PErrorCode write(ofstream&);
};

//=====
// Class:      Valve
// Superclass: None
// Description: Class for valve
//=====

class Valve : public PictObject
{
public:
    Valve(Converter*);
    PErrorCode write(ofstream&);
};

//=====
// Class:      ControlRod
// Description: Class for ControlRod
//=====

class ControlRod : public PictObject
{
public:
    ControlRod(Converter*);
    PErrorCode write(ofstream&);
};

//=====
// Class:      Pump
// Superclass: PictObject
// Description: Class for pumps
//=====

class Pump : public PictObject
{
public:
    Pump(Converter*);
    PErrorCode write(ofstream&);
};

//=====
// Class:      Cooling
// Superclass: PictObject
// Description: Class for cooling system
//=====

class Cooling : public PictObject
{
public:
    Cooling(Converter*);
    PErrorCode write(ofstream&);
};
```

1997年 09月 11日 16時 46分 /users/pclund/maap/m2p/category.h ページ 4

```
//=====
// Class ContFloor
//=====

//=====
// Class:      CoreNode
// Superclass: PictObject
// Description: Class for heat exchangers
//=====

class CoreNode : public PictObject
{
public:
    CoreNode(Converter*);
    // ~CoreNode();
    PErrorCode read(ifstream&,SDTstring,int,int,int);
    PErrorCode write(ofstream&);
};

//=====
// Class:      WaterLevel
// Superclass: PictObject
// Description: Class for water levels
//=====

class WaterLevel : public PictObject
{
public:
    WaterLevel(Converter*);
    // ~WaterLevel();
    PErrorCode read(ifstream&,SDTstring,int,int,int);
    PErrorCode write(ofstream&);
};

//=====
// Class:      SprayPattern
// Superclass: PictObject
// Description: Class for spray pattern
//=====

//=====
// Class:      Shield
// Superclass: PictObject
// Description: Class for shield
//=====

//=====
// Class:      Walls
// Superclass: PictObject
// Description: Class for walls
//=====

//=====
// Class:      H2Conc
// Superclass: PictObject
```

1997年 09月 11日 18時 33分 /users/pclund/maap/m2p/writeComp.c ページ 1

```
//=====================================================================
// 
// File name:      writeComp.c
// Author:         Per Christer Lund, PNC
// Creation date: June. 1997
// Description:   MAAP-GRAAP to PICASSO translator
//                  Methods for writing the compartment picture
//=====================================================================

#include "maap2picasso.h"

//=====================================================================

PErrorCode Converter::writeCompartment(SDTstring fname)
{
    PErrorCode status;
    char text[MDUM];
    int i,j,k;
    int NWIN = 4;           // Number of windows
    int worldX = 1000, worldY = 1000;

    // Open file
    ofstream outp(fname);
    if (! outp.is_open())
        return theLogger->error((SDTstring)"Error opening output file " + fname);

    // Write header
    if ((status = writeHeader(outp,"compartment","","")) != L_OK) return status;

    // Write picture objects
    sprintf(text,"Writing %i %s pictures",nobj[COMPARTMENT],(char*)cdef[COMPARTMENT].name);
    theLogger->log(text,TO_OUTPUT);

    for(i = 0; i < NWIN/2; i++){
        for(j = 0; j < NWIN/2; j++){
            outp << "    instance maapLib.CompFrame " << endl;
            outp << "    {" << endl;
            outp << "        attribute comp = 'compData[" << i*2 + j << "]';" << endl;
            outp << "        x = " << i * (worldX/2) << ";" << endl;
            outp << "        y = " << j * (worldY/2) << ";" << endl;
            outp << "        visibility = 1;" << endl;
            outp << "        xReference = 1;" << endl;
            outp << "        yReference = 1;" << endl;
            outp << "        xScaleFactor = 1;" << endl;
            outp << "        yScaleFactor = 1;" << endl;
            outp << "        rotationAngle = 0;" << endl;
            outp << "    } // end of instance \n" << endl;
        }
    }

    for(i = 0; i < nobj[COMPARTMENT];i++){
        Compartment* comp = (Compartment*) pobj[COMPARTMENT][i];
        for(j = 0; j < 2; j++) {

```

1997年 09月 11日 18時 33分 /users/pclund/maap/m2p/writeComp.c ページ 2

```

for(k = 0; k < 2; k++) {
    comp->setCoord(0.5,0.5,j*worldX/2,k*worldY/2);
    if ((status = comp->write(outp,j*2+k)) != L_OK) return status;
}
}

// Write footer
if ((status = writeFooter(outp,worldX,worldY,0,0)) != L_OK) return status;

outp.close();
theLogger->logg((SDTstring)"Written PICASSO file      " + fname,TO_OUTPUT);

return status;
}

//=====
PErrorCode Converter::writeFooter(ofstream& of,int height,int width,int x,int y)
{
// int ypos= 43,xpos = 900,lh = 31;
// Write the color legends

of << " backgroundColour      = 0;" << endl;
of << " xSnap                  = 0;" << endl;
of << " ySnap                  = 0;" << endl;
of << " angleSnap               = 0;" << endl;
of << " radiusSnap              = 0;" << endl;
of << " worldX                 = " << x << ";" << endl;
of << " worldY                 = " << y << ";" << endl;
of << " worldWidth              = " << width << ";" << endl;
of << " worldHeight             = " << height << ";" << endl;
of << " oneToOne                = 1;" << endl;
of << " resizeMode               = 0;" << endl;
of << " updateMode              = 2;" << endl;
of << " crosshairStyle          = 1;" << endl;
of << " crosshairLength         = 0;" << endl;
of << " crosshairFollowSnap     = 'isInEditMode()';" << endl;
of << " crosshairVisibility     = 'isInEditMode()';" << endl;
of << " xReference              = " << (width -x) / 2 << ";" << endl;
of << " yReference              = " << (height - y)/2 << ";" << endl;
of << " xScaleFactor            = 1;" << endl;
of << " yScaleFactor            = 1;" << endl;
of << " rotationAngle           = 0;" << endl;
of << "}"                      // End of picture MainPict" << endl;
of << "===== end of file =====" << endl;

return L_OK;
}

//=====
// Compartment category class
//=====

Compartiment::Compartiment(Converter* cd) : PictObject(cd)

```

1997年 09月 11日 18時 33分 /users/pclund/maap/m2p/writeComp.c ページ 3

```
{
    catNr = COMPARTMENT;
    name = "Compartment";
    eventObj = 0;
    isWater = 0;
    noVis = 1;
}

//=====

PErrorCode Compartment::write(ofstream& outp, int indx)
{
    char str2[MDUM];
    strcpy(str2,desc);
    strtok(str2, " ");
    objNr = atoi(strtok(NULL, " "));

    outp << endl;
    outp << " polygon \t// " << name << objNr << " " << desc << endl;
    outp << " {" << endl;
    coords->write(outp);
    outp << "   foregroundFillColour = 'compData[" << indx << "].color[" << objNr << "]';" << endl;
    outp << "   visibility = 1;" << endl;
    writeDefault(outp);
    outp << " }" << endl;
    return writeText(outp,indx);
}

//=====

PErrorCode Compartment::writeText(ofstream& outp, int indx)
{
    xpos = coords->xRef/2 - 20;
    ypos = coords->yRef/2;

    if (indx > 1) xpos += 500;
    if (indx == 1 || indx == 3) ypos+=500;

    outp << " text" << endl;
    outp << " {" << endl;
    outp << "   format = \"% .2f\";" << endl;
    outp << "   theText = 'compData[" << indx << "].data[" << objNr << "]';" << endl;
    outp << "   foregroundColour = 'white';" << endl;
    outp << "   theFont = 'font3';" << endl;
    writeStandard(outp,2);
    outp << " }\n" << endl;

    return L_OK;
}

//===== end of file =====
```

1997年 08月 13日 11時 51分 /users/pclund/maap/m2p/writeButton.c ページ 1

```

//=====
// File name:      m2c.c
// Author:         Per Christer Lund, PNC
// Creation date: June. 1997
// Description:   MAAP-GRAAP to PICASSO translator
//
//=====

#include "maap2picasso.h"

PErrorCode Converter::writeButtons()
{
    int i, ypos = 0, xpos = 0;

    // Open output file name
    SDTString bfname = getenv(MAAPHOME); bfname+="/picasso/maapButton.Tdoc";
    ofstream bstr(bfname);
    if (! bstr.is_open()) return theLogger->error("Error opening file " + bfname);
    theLogger->log("Writing PICASSO-3 Tdoc file " + bfname);

    // Write header
    bstr << "/* MAAP-GRAPH to PICASSO configuration file " << endl;
    bstr << " This file declares the category buttons " << endl;
    bstr << " This file is automatically generated by the program m2p" << endl;
    bstr << "*/\n" << endl;
    bstr << "picture buttonPict \n {" << endl;
    bstr << endl;

    // Write the buttons
    for(i = 1; i < NCAT; i++) {
        if (cdef[i].ctyp == -1) continue;
        bstr << "\n instance maapLib.pushButton // Visibility button\n {" << endl;
        bstr << "     attribute borderColourReleased = 9;" << endl;
        bstr << "     attribute borderColourPressed = 0;" << endl;
        bstr << "     attribute DOWN = 1;" << endl;
        bstr << "     attribute UP = 0;" << endl;
        bstr << "     attribute state = " << !cdef[i].vis << ";" << endl;
        bstr << "     attribute bottomColourReleased = 17;" << endl;
        bstr << "     attribute bottomColourPressed = 17;" << endl;
        bstr << "     attribute buttText = \" " << cdef[i].name << "\";" << endl;
        bstr << "     attribute visFlag = '&:fugen.fugenPict.visFlag' << i << ';" << endl;
        bstr << "     x = " << xpos << ";" << endl;
        bstr << "     y = " << ypos << ";" << endl;
        bstr << "     visibility = 1;" << endl;
        bstr << "     xReference = " << xpos << ";" << endl;
        bstr << "     yReference = " << ypos << ";" << endl;
        bstr << "     xScaleFactor = 1;" << endl;
        bstr << "     yScaleFactor = 1;" << endl;
        bstr << "     rotationAngle = 0;" << endl;
        bstr << " } // end of instance \n" << endl;
        ypos += 35;
    }

    // Write the other picture data
    // The picture's other attributes:
}

```

1997年 08月 13日 11時 51分 /users/pclund/maap/m2p/writeButton.c ページ 2

```
bstr << "backgroundColour      = 0;" << endl;
bstr << "xSnap                = 0;" << endl;
bstr << "ySnap                = 0;" << endl;
bstr << "angleSnap             = 0;" << endl;
bstr << "radiusSnap            = 0;" << endl;
bstr << "worldX                = 0;" << endl;
bstr << "worldY                = 0;" << endl;
bstr << "worldWidth             = 250;" << endl;
bstr << "worldHeight            = 600;" << endl;
bstr << "oneToOne               = 1;" << endl;
bstr << "resizeMode              = 0;" << endl;
bstr << "updateMode              = 1;" << endl;
bstr << "crosshairStyle          = 1;" << endl;
bstr << "crosshairLength         = 0;" << endl;
bstr << "crosshairFollowSnap     = 'isEditMode()';" << endl;
bstr << "crosshairVisibility     = 'isEditMode()';" << endl;
bstr << "xReference             = 100;" << endl;
bstr << "yReference             = 100;" << endl;
bstr << "xScaleFactor            = 1;" << endl;
bstr << "yScaleFactor            = 1;" << endl;
bstr << "rotationAngle           = 0;" << endl;
bstr << "}" << endl;

bstr.close();

return L_OK;
}

//===== end of file =====
```

1997年 09月 11日 17時 23分 /users/pclund/maap/m2p/maap2picasso.h ページ 1

```
=====  

//  

// File name:      maap2picasso.h  

// Author:         Per Christer Lund, PNC  

// Creation date: June. 1997  

// Description:   MAAP-GRAAP to PICASSO translator  

//  

=====  

#ifndef MAAP2PICASSO  

#define MAAP2PICASSO  

#include <stdlib.h>  

#include <stdio.h>  

#include <fstream.h>  

#include <string.h>  

#include "logger.h"  

#define PPICTEXT "Pict.Tdoc"    // PICASSO-3 picture definiton file extension  

#define BPICTEXT "Butt.Tdoc"    // PICASSO-3 button picture def. file ext.  

#define MPICTEXT ".pic"        // MAAP picture definiton file extension  

#define PDEFEXT " .pdef"       // PICASSO definition file extension  

#define COMPARTNAME "compartmentPict.Tdoc" // Compartment picture file name  

#define INCNAME "mainPict.inc"    // Main picture include  

#define LOGFILE "./maap2picasso.log" // Logg file  

#define MAAPHOME "MAAPHOME"     // Environment variable for maap def. files  

#define MAXCOMP 13             // Maximum number of compartments  

#define NODESECT 18            // Number of node sections  

#define NLEGEND 25             // Number of color legends  

#define NCAT 24                // Number of categories  

#define MAXCAT 1000            // Maximum number of object in each category  

// Definition keywords  

#define COORDKEY   "COORD"  

#define PICASSOKEY "PICASSO.DB"  

#define CATEGORYKEY "CATEGORY"  

#define TEXTKEY     "TEXTOBJECT"  

#define EVENTKEY    "EVENTS"  

#define ENDKEY      "END"  

// Picture categories  

enum ShapeCategory {PIPE = 1, VALVE, CONTROLROD, RODOUT, PUMP, CONTFLOOR, CORENODE,  

                    WATERLEVEL, SPRAYPATTERN, SHIELD, WALLS, H2CONC, RODDRIVE, CONCRETE,  

                    CONTWATER, CORIUM, DRYWELLLEFT, DRYWELLRIGHT, WATERDRYWELLLEFT, WATERDRYWELLRIGHT, CONTFLAG,  

                    TUBE, COMPARTMENT};  

#define MAXCOORD 1000  

#define MAXVAL 10000  

#define MINVAL -10000  

#define MDUM 1000  

class Converter;  

#include "category.h"
```

1997年 09月 24日 11時 44分 /users/pclund/maap/picasso/data/maapData.h ページ 1

```
/*
FILE: maapData.h
This file contains the shared PICASSO data with MAAP
This is included in the "fugenMain.Tdoc" file
Author: Per Christer Lund, PNC/FUGEN
Date: August 1997
*/
***** */

/* -----
/* PICASSO specific parameters
/* ----- */

/* Flags */
//float globalTime = 0;      /* The MAAP trancient time */
int theLanguage = 0;        /* Language flag */
int maapIsRunning = 0;      /* Run flag */
int STATUSFLAG = 0;
int WARNINGFLAG = 1;
int ERRORFLAG = 2;
int visFlag[23] = 1;        /* Visibility flags */

/* Strings */
char warningString[100] = "Warning";
char errorString[100] = "Error";
char statusText[100] = "MAAP not started"; /* Status text */
char statusString[100] = "MAAP not started"; /* Status text */
char timeString1[100] = "00:00:00.000"; /* Time label */
char timeString2[100] = "00:00:00.000"; /* Time label */
char timeString3[100] = "00:00:00.000"; /* Time label */
char timeString4[100] = "00:00:00.000"; /* Time label */

/* Trend logger */
int timeVar = 0;

/* -----
/* MAAP specific parameters and structures
/* ----- */

float maapTime = 0;          /* The MAAP trancient time */
float realTime = 0;           /* The real simulation time */
int32 ievent[351] = 2;       /* The MAAP/GRAAPH shape events. */
float waterLevel[351] = 0.0;  /* Normalized water levels */
float nodeWidth[108] = 1.0;   /* Normalized core fuel node width */
int32 nodeColor[108] = 27;    /* Core fuel node temperatures (27 = grey) */

/* Compartment data */
struct CompartData
{
    char name[32];             /* Variable number */
    char unit[8];              /* Unit string */
    int indx[40];               /* Pointer index */
    float data[40];             /* Data */
    int color[40];              /* Color code */
}
```

1997年 09月 24日 11時 44分 /users/pclund/maap/picasso/data/maapData.h ページ 2

```
float minValue;           /* Minimum value */
float stepVal;            /* Color change increment */
float minValue;           // Min. value
float maxValue;           // Maximum compartment value
int iunit;                /* Unit flag (1 = SI, 2 = BR) */
int icf;                  /* Conversion factor */
int chFlag;               /* Change flag */
int failFlag;             /* = 1 if the variable is not found by MAAP */
char message[100];         // Error/warning etc.
};

CompartData compData[4];    /* Four simultaneous compartment data */

/* Scalar variable */
struct MaapVariable
{
    char name[32];          /* Variable number */
    char unit[8];            /* Unit string */
    int idx;                 /* Pointer index */
    float data;              /* Data */
    float minValue;          /* Min. value */
    float maxValue;          /* Maximum compartment value */
    int iunit;                /* Unit flag (1 = SI, 2 = BR) */
    int icf;                  /* Conversion factor */
    int chFlag;               /* Change flag */
    int failFlag;             /* = 1 if the variable is not found by MAAP */
    char message[100];         /* Error/warning etc. */
};

MaapVariable maapVar[50];   /* A maximum of 50 MAAP variables */

/*===== end of file =====*/
```

1997年 09月 11日 17時 23分 /users/pclund/maap/m2p/maap2picasso.h ページ 2

```

int charstrip(char*,char*); // Translate 2 byte Kanjii to two 1 byte ascii
//=====
//
// Picture category class
//
//=====

class CategoryDef {
public:
    CatDef() {ctyp = -1; lth = 1; vis = 1;}
    int      ctyp;      // Category type
    SDTstring fcol;    // fill color
    int      lth;       // line thickness
    SDTstring lcol;    // line color
    int      patt;     // Pattern
    int      vis;       // Visibility flag
    SDTstring name;   // Name
};

//=====
//
// Converter class
//
//=====

class Converter
{
public:
    Converter(int rf = 0);           // Constructor
    ~Converter();                   // Destructor

    PErrorCode convertPict(SDTstring); // Covert picture file

    int pxl,pyl,px2,py2;           // Global picasso coordinates
    float xcf, ycf;               // Coordinate conversion factors
    CategoryDef cdef[NCAT];        // Array of category definitions

private:
    PErrorCode readMaapFile(SDTstring); // Read the MAAP picture file
    PErrorCode readDefinitions(SDTstring); // Read the PICASSO definition file
    PErrorCode identifyObject(ifstream&); // Identify and write object

    PErrorCode writeMainPict(SDTstring); // Write the main picture file
    PErrorCode writeHeader(ofstream&,SDTstring,SDTstring,SDTstring); // Write the output file header
    PErrorCode writeCompartiment(SDTstring); // Write the compartment file
    PErrorCode writeFooter(ofstream&,int,int,int,int); // Write the compartment footer

    int      runPcc;                // Flag to determine whether to runpcc
    SDTstring caseName;            // The case name
    SDTstring casePath;            // The case name

    PictObject* pobj[NCAT][MAXCAT]; // Picture objects
    int      nobj[NCAT];           // Number of objects in each category
    TextPict* txtObj[MAXCAT];     // Free picture objects
    int nTxt;                     // Number of free text objects
    int lNr;                      // Line number
}

```

1997年 09月 11日 17時 23分 /users/pclund/maap/m2p/maap2picasso.h ページ 3

```
};  
#endif  
//===== end of file =====
```

1997年 09月 11日 15時 48分 /users/pclund/maap/m2p/m2p.c ページ 1

```

//=====
// File name:      m2c.c
// Author:         Per Christer Lund, PNC
// Creation date: June. 1997
// Description:   MAAP-GRAAP to PICASSO translator
// Revisions:
//=====

#include "maap2picasso.h"

// Global variables
Logger* theLogger = NULL; // A logger object
#define CDUM 100
char currDate[CDUM];           // Not used (must be included)
char currMessage[CDUM];        // Not used (must be included)

//=====
// MAIN program
//=====

int main(int argc,char** argv)
{
    PErrorCode status;
    SDTstring cname;
    int runPcc = 1,writeButt = 0;

    // Check arguments
    if (argc <2){
        cout << "Usage: " << argv[0] << " <flags> <MAAP definition name i.e. fugen>" << endl;
        cout << "\tflags: -n do no run PICASSO pcc" << endl;
        cout << "\t\t-b make button picture" << endl;
        return -1;
    }

    // Check that environment variable MAAHOME exists
    if (strlen(getenv(MAAHOME)) < 1){
        cout << argv[0] << " ERROR: no MAAHOME environment variable" << endl;
        cout << "Please set environment variable MAAHOME to the place your MAAP input files resides" << endl;
        cout << "Also remember to create a $(MAAHOME)/picasso directory where the output files will be saved\n"
        return -1;
    }

    // Get the arguments
    // Should have used the getopt() C function here, maybe later.....
    if (argv[1][0] == '-'){
        if (argv[1][1] == 'n') runPcc = 0;
        cname = argv[2];
    }
    else
        cname = argv[1];

    // Open logger file
    theLogger = new Logger(LOGFILE);
    if (! theLogger || theLogger->status != L_OK){
        cout << "Error opening message logger" << endl;
    }
}

```

1997年 09月 11日 15時 48分 /users/pclund/maap/m2p/m2p.c ページ 2

```
    return -1;
}
theLogger->logg("\n\tMAAP-GRAAPH to PICASSO-3 converter", TO_OUTPUT);

// Make a converter
Converter* conv = new Converter(runPcc);

// Convert the picture file (cname.pic)
status = conv->convertPict(cname);

// That's all
delete conv;
delete theLogger;

cout << "Logg written to file " << LOGGFILE << endl;

return (status == L_OK) ? 0 : -1;
}

//===== END OF FILE m2p.c =====
```

1997年 09月 11日 18時 32分 /users/pclund/maap/m2p/convert.c ページ 1

```

//=====
// 
// File name:      convert.c
// Author:        Per Christer Lund, PNC
// Creation date: June. 1997
// Description:   MAAP-GRAAP to PICASSO translator
// 
//=====

#include <time.h>
#include "maap2picasso.h"

//=====

Converter::Converter(int rf = 0)
{
    runPcc = rf;
    nTxt = 0;
    for(int i = 0; i < NCAT; i++) nobj[i] = 0;
}

//=====

Converter::~Converter(){}

//=====
// Method:convertPict(string caseName):
// Argument:
//     caseName - a string containing the case name, without prefix, f.ex. "fugen"
// Description:
//     The method does the following:
//     1) Creates a set of input and output file names from the case name, for example
//         "fugen.pic", "fugen.pdef" input files, and "fugenPict.Tdoc" output name. It
//         also creates a compartment name "compartmentPic.Tdoc". All these files are
//         expected to reside at the current directory.
//
//     2) Calls the method "readDefinitions" to read the Picasso definition file
//
//     3) Calls the method "readMaapFile" to read and interpret MAAP/GRAAPH picture file
//
//     4) Calls the method "writeMainPict" to write the main window Tdoc file
//
//     5) Calls the method "writeCompartment" to write the compartment Tdoc file
//
//     6) If the flag "runPcc" is set (default), the picasso-3 program "pcc" is invoked
//        to compile the two Tdoc files.
//
// Return: status
//=====

PErrorCode Converter::convertPict(SDTstring cn)
{
    PErrorCode status;
    SDTstring ifname, ofname, cname, command, dfname;

```

1997年 09月 11日 18時 32分 /users/pclund/maap/m2p/convert.c ページ 2

```

// File names
caseName = cn;
casePath = getenv(MAAPHOME); casePath += "/"; casePath += caseName;
ifname = caseName + MPICTEXT; // Input file name
ofname = caseName + PPICTEXT; // output file name
cfname = COMPARTNAME; // Compartment file name
dfname = caseName + PDEFEXT; // Picasso definition file name

// Read the definitions
if ((status = readDefinitions(dfname)) != L_OK) return status;

// Read the maap/graaph picture file
if ((status = readMaapFile(ifname)) != L_OK) return status;

// Write the main picture
if ( (status = writeMainPict(ofname)) != L_OK) return status;

// Write compartment picture file
if
(nobj[COMPARTMENT] == 0)theLogger->logg("No compartments ");
else
if ( (status = writeCompartment(cfname)) != L_OK) return status;

// Run the PICASSO preprocessor
if (! runPcc) return theLogger->logg("Not invoking the PICASSO preprocessor at this time",TO_OUTPUT);
command = "pcc " + ofname;
if (system(command) != 0)
return theLogger->error("Error running command " + command);

// command = "pcc " + cfname;
// if (system(command) != 0)
// return theLogger->error("Error running command " + command);

return L_OK;
}

//=====
// Method: readMaapFile()
// Arguments:
//   filename - name of the Graaph picture file, f.ex. "fugen.pic"
// Description:
//   The methods performs the following tasks:
//   1) Reads the graaph picture size constants, and computes the conversion
//      parameters from GRAAPH cooridnates to Picasso world coordinates.
//   2) Ignores three following lines (may be utilized later)
//   3) Calls the method "identifyObject" to read the definition of each graaph
//      picture until end of file.
//=====

PErrorCode Converter::readMaapFile(SDTstring fname)
{
    PErrorCode status;
    char dum[MDUM];
    float mx1,my1,mx2,my2; // Global MAAP coordinates
    float ip1, ip2;
}

```

1997年 09月 11日 18時 32分 /users/pclund/maap/m2p/convert.c ページ 3

```

lNr = 0;

// Open input file
ifstream inp(fname);
if (!inp.is_open()) return theLogger->error((SDTstring)"Error opening input file " + fname);

// Read the global MAAP coordinates
inp.getline(dum,MDUM);
if (! strstr(dum,"SIZE")) return theLogger->error("Keyword SIZE expected");
inp >> mx1 >> my1 >> mx2 >> my2;

// Compute the conversion factors
xcf = (px2 - px1)/(mx2 - mx1);
ycf = (py2 - py1)/(my2 - my1);

// Ignore the following lines (so far)
inp.getline(dum,MDUM);
inp.getline(dum,MDUM);
inp.getline(dum,MDUM);
lNr = 6;

// Read the pictures
while(! inp.eof() && ((status = identifyObject(inp)) == L_OK));
if (status != L_OK) return status;

// Close input files
inp.close();

theLogger->log((SDTstring)"Read MAAP picture file " + fname,TO_OUTPUT);

return status;
}

```

```

//=====
// Method: writeMainPict
// Arguments:
//   output file name
// Description:
//   This method performs the following:
//   1) Opens the PICASSO-3 Tdoc file for the main picture window
//   2) Writes a general commented header with summary of the picture objects
//   3) Writes a list of visibility flags which is used by the buttons to toggle
//      the visibility of pictures on/off. The initial value is read from the
//      CategoryDef object.
//   4) Writes the WaterLevel picture objects. These has to be written first to
//      ensure they are not obscuring other images.
//   5) Writes the other pictures in the remaining categories. The order of writing
//      can be changed, in order to let some pictures be on top or under others.
//   6) Writes the free text objects
//   7) Appends the general text file "mainPict.inc" to include the buttons, header
//      text and other case-independent images to the window.
//=====
```

```
PErrorCode Converter::writeMainPict(SDTstring fname)
```

1997年 09月 11日 18時 32分 /users/pclund/maap/m2p/convert.c ページ 4

```
{
    PErrorCode status;
    char text[MDUM];
    int i, j;

    // Open output file
    ofstream outp(fname);
    if (!outp.is_open()) return theLogger->error((SDTstring)"Error opening output file " + fname);

    // Write the header
    if ((status = writeHeader(outp, caseName, "", "")) != L_OK) return status;

    // Write the picture visibility flag
    outp << "// Constructor" << endl;
    outp << "function 'void constructor()'" << endl;
    outp << "{" << endl;

    for(i = 1; i < NCAT-1; i++)
        outp << "    visFlag[" << i << "] = " << cdef[i].vis
            << ";" // Visibility flag for " << cdef[i].name << endl;
    outp << "}" << endl;

    outp << "    int lvisFlag = 1; // Visibility flag for the water level objects" << endl;

    // Write the objects; first the water levels
    sprintf(text,"Writing %i %s pictures",nobj[WATERLEVEL],(char*)cdef[WATERLEVEL].name);
    theLogger->log(text,TO_OUTPUT);
    for(j = 0; j < nobj[WATERLEVEL];j++) {
        if ((status = pobj[WATERLEVEL][j]->write(outp)) != L_OK) return status;
    }

    // Write the objects; then the walls
    sprintf(text,"Writing %i %s pictures",nobj[WALLS],(char*)cdef[WALLS].name);
    theLogger->log(text,TO_OUTPUT);
    for(j = 0; j < nobj[WALLS];j++) {
        if ((status = pobj[WALLS][j]->write(outp)) != L_OK) return status;
    }

    // Then the other pictures in sequence
    for(i = 0; i < COMPARTMENT; i++){
        if (nobj[i] == 0 || i == WATERLEVEL || i == WALLS) continue;
        sprintf(text,"Writing %i %s pictures",nobj[i],(char*)cdef[i].name);
        theLogger->log(text,TO_OUTPUT);
        for(j = 0; j < nobj[i];j++)
            if ((status = pobj[i][j]->write(outp)) != L_OK) return status;
    }

    // Write the text objects
    sprintf(text,"Writing %i text objects",nTxt);
    theLogger->log(text,TO_OUTPUT);
    for(i = 0; i < nTxt; i++)
        if ((status = txtObj[i]->write(outp)) != L_OK) return status;
}
```

1997年 09月 11日 18時 32分 /users/pclund/maap/m2p/convert.c ページ 5

```

// Write the footer
writeFooter(outp, 730, 1000, 0, 110);

// Close output file
outp.close();

theLogger->log((SDTstring)"Written PICASSO file      " + fname, TO_OUTPUT);
return status;
}

//=====
//=====
PErrorCode Converter::writeHeader(ofstream& of, SDTstring pname, SDTstring ifname, SDTstring ofname)
{
    time_t now = time(NULL);
    of << /* MAAP-GRAAPH to PICASSO configuration file " << endl;
    of << " This file is automatically generated by the m2p program " << endl;
    of << " Creation date           : " << ctime(&now);
    of << " User                  : " << getenv("USER") << endl;
    of << " MAAP/GRAAPH picture input file name   : " << ifname << endl;
    of << " Auxillary PICASSO-3 definition file name : " << endl;
    of << " Summary:" << endl;
    of << "     Global coordinate conv. factor: " << xcf << "(x) " << ycf << "(y) " << endl;
    for(int i = 1; i < NCAT;i++){
        if (nobj[i] == 0) continue;
        of << "     " << cdef[i].name << ":\t" << nobj[i] << " picture(s)" << endl;
    }
    of << "     Text objects           : " << nTxt << endl;
    of << "*/\n" << endl;
    of << "picture " << pname << "Pict \n {" << endl;
    of << endl;

    return L_OK;
}

//=====
// Method: identifyObject()
// Arguments:
//   input file
// Description:
//   This method does the following:
//   1) Reads the picture number, the number of point coordinates and an picture
//      description from the input file
//   2) Reads the picture category number [1-23], and 5 additional parameters which
//      are not used.
//   3) Create an instance of a PictureObject subclass according to the category
//      number.
//   4) Calls the generic "read" method to the picture instance to read the remaining
//      data (i.e. point coordinates). Each instance will then build a Tdoc object
//      according to its class definitions.
//=====

PErrorCode Converter::identifyObject(ifstream& inp)
{
    PErrorCode status;

```

1997年 09月 11日 18時 32分 /users/pclund/maap/m2p/convert.c ページ 6

```

int np,ix,iy,objNr,ip1,ip2,ip3,ip4,ip5,icol,ityp, poly = 0,icmp;
int iwater = 1;
float x,y;
char cdum[MDUM],lName[MDUM],objname[10],txt[1000];
char* desc;

// Get the picture number, number of coordinates and picture description
inp >> objNr >> np;
inp.getline(lName,MDUM);
if (inp.eof()) return L_OK;
desc = strstr(lName,"/");
desc++;
desc++;

// Get picture category number (and some unused data)
inp >> ityp >> ip2 >> ip3 >> ip4 >> ip5 >> icol;
inp.getline(cdum,MDUM);
lNr += 2;

// Make the picture object accoring to the ityp
switch (ityp) {
case PIPE:
    if (np == 31)      pobj[ityp][nobj[ityp]] = new HeatEx(this);
    else if (np == 11)  pobj[ityp][nobj[ityp]] = new HxCoil(this);
    else              pobj[ityp][nobj[ityp]] = new Pipe(this);
    break;
case VALVE:
    if (strstr(lName,"VALVE") && np == 5) pobj[ityp][nobj[ityp]] = new Valve(this);
    else          pobj[ityp][nobj[ityp]] = new DefaultPict(this,ityp);
    break;
case CONTROLROD:
    pobj[ityp][nobj[ityp]] = new DefaultPict(this,ityp);
    break;
case PUMP:
    if (strstr(desc,"PUMP") && np == 31) pobj[ityp][nobj[ityp]] = new Pump(this);
    else          pobj[ityp][nobj[ityp]] = new Cooling(this);
    break;
case CORENODE:
    pobj[ityp][nobj[ityp]] = new CoreNode(this);
    break;
case WATERLEVEL:
    pobj[ityp][nobj[ityp]] = new WaterLevel(this);
    break;
case COMPARTMENT:
    pobj[ityp][nobj[ityp]] = new Compartment(this);
    break;
// Default pictures
case DRYWELLRIGHT: case DRYWELLLEFT: case RODOUT: case WATERDRYWELLLEFT: case WATERDRYWELLRIGHT:
case RODDRIVE: case TUBE: case CONFLAG: case CORIUM: case CONTWATER: case CONCRETE: case H2CONC:
case WALLS: case SHIELD: case SPRAYPATTERN: case CONTFLOOR:
    pobj[ityp][nobj[ityp]] = new DefaultPict(this,ityp);
    break;
// Illegal category
default:
    sprintf(txt,"Illegal picture category %i (%s), line nr. %i",ityp,desc,lNr);
    return theLogger->error(txt);
}

```

1997年 09月 11日 18時 32分 /users/pclund/maap/m2p/convert.c ページ 7

```

// Read the rest of the data
if (! pobj[ityp][nobj[ityp]]) return theLogger->error("Error creating picture object");
status = pobj[ityp][nobj[ityp]]->read(inp,desc,np,objNr,nobj[ityp]);
nobj[ityp]++;
lNr += np;

sprintf(txt,"Object %s: line nr %i ",(char*)pobj[ityp][nobj[ityp]]->Name(),lNr);
theLogger->log(txt);

return status;
}

//=====
// Method: readDefinitions(string filename)
// Arguments:
//   filename - name of the Picasso-3 defintion file (f.ex. "fugen.pdef")
// Description:
//   This method opens the definition file and reads the definitions. The format of
//   the definition file is given elsewhere.
//   The definitions concerning the categories are stored in a "CategoryDef" object.
//   The free text definitions are stored in a list of "TextPict" objects.
//=====

PErrorCode Converter::readDefinitions(SDTstring fname)
{
    PErrorCode status;
    char dum[MDUM],*tok;
    int i,ict;
    SDTstring picassoFName;

    // Open defintion file
    ifstream dfil(fname);
    if (! dfil.is_open()) return theLogger->error("Cannot open definition file " + fname);

    // Read global definitions
    while(dfil.getline(dum,MDUM)){
        if (dum[0] == '/' || strlen(dum) < 5) continue; // Ignore comments and empty lines
        tok = strtok(dum, "\n\t");

        // Read the coordinates
        if (strstr(tok,COORDKEY)) {
            px1 = atoi(strtok(NULL, "\n\t"));
            py1 = atoi(strtok(NULL, "\n\t"));
            px2 = atoi(strtok(NULL, "\n\t"));
            py2 = atoi(strtok(NULL, "\n\t"));
            continue;
        }

        // Read the PICASSO-3 database file name (not used yet)
        if (strstr(tok,PICASSOKEY)) {
            picassoFName = strtok(NULL, "\n\t");
            continue;
        }

        // Read the category definitions
    }
}

```

1997年 09月 11日 18時 32分 /users/pclund/maap/m2p/convert.c ページ 8

```

if (strstr(tok,CATEGORYKEY)) {
    int nCat = atoi(strtok(NULL, "\n\t"));
    if (nCat < 1 || nCat > NCAT) return theLogger->error("Illegal category number");
    for(ict = 1; ict <= nCat; ict++) {
        dfil.getline(dum,MDUM);
        tok = strchr(dum,'/');
        tok++;
        i = atoi(strtok(dum, "\n\t"));
        if (i != ict) return theLogger->error("Inconsistency");
        cdef[i].ctyp = atoi(strtok(NULL, "\n\t"));
        if (cdef[i].ctyp >= 0) {
            cdef[i].fcol = strtok(NULL, "\n\t");
            cdef[i].lth = atoi(strtok(NULL, "\n\t"));
            cdef[i].lcol = strtok(NULL, "\n\t");
            cdef[i].patt = atoi(strtok(NULL, "\n\t"));
            cdef[i].vis = atoi(strtok(NULL, "\n\t"));
            cdef[i].name = tok;
        }
    }
    dfil.getline(dum,MDUM);
    if (!strstr(dum,ENDKEY)) return theLogger->error(dum);
    continue;
}

// Read and write the text objects
if (strstr(tok,TEXTKEY)) {
    while(dfil.getline(dum,MDUM)){
        if (strstr(dum,ENDKEY)) break; // end of input
        txtObj[nTxt] = new TextPict;
        txtObj[nTxt]->read(dum,nTxt);
        nTxt++;
    }
    continue;
}

// Else illegal keyword
cout << "WARNING: Unrecognized keyword " << tok << endl;
}

dfil.close();
theLogger->log((SDTstring)"Read PICASSO definition file " + fname,TO_OUTPUT);

return L_OK;
}

//=====
// FUNCTION:  charstrip
// ARGUMENTS:  (char*) kstr:  the original character string
//             (char*) astr:  the modified character string
// RETURN:     (int):  number of characters modified
// DESCRIPTION: Makes a modified copy of the original string, where
//               the last bit in each character is set to 0.
//=====

int charstrip(char* kstr,char* astr)
{

```

1997年 09月 11日 18時 32分 /users/pclund/maap/m2p/convert.c ページ 9

```
char tmp, *indx=astr;
int nchr=0;
for(int i=0;i<strlen(kstr); i++){
    tmp = kstr[i] & 127;           // Set last bit
    if (tmp != kstr[i]) nchr++;   // Is this character modified?
    // Check for special character
    if (tmp == '\\' || tmp == '\"') {*indx = '\\'; indx++;}
    *indx = tmp;
    indx++;
}
*indx = '\0'; // Add NULL character
return nchr;
}

//----- end of file -----
```

1997年 09月 11日 16時 46分 /users/pclund/maap/m2p/category.h ページ 1

```

//=====
// 
// File name:      category.h
// Author:         Per Christer Lund, PNC
// Creation date: Aug. 1997
// Description:    MAAP-GRAAP to PICASSO translator
// 
//=====

#ifndef CATEGORY_H
#define CATEGORY_H

//=====
// Class:          Coord
// Superclass:     None
// Description:    Containing a coordinate vector and methods for IO
//=====

class Coord {
public:
    Coord(Converter*,int);           // Constructor
    ~Coord();                      // Destructor
    PErrorCode read(ifstream&);     // Read coordinates
    PErrorCode write(ofstream&);    // Write coords in Tdoc format
    float getAngle();              // Return angle (for valves)
    void setCoord(float x,float y,int ix,int iy)
    {xsize = x; ysize = y; x0 = ix; y0 = iy;} //

    float maxx,maxy,minx,miny;      // Maximum and minimum values
    float xRef,yRef;               // The reference coordinate
    int nCoord;                    // Number of coordinates
    float x[MAXCOORD];            // Vector of x coordinates
    float y[MAXCOORD];            // Vector of y coordinates
    Converter* conv;               // Pointer to the main object
    int x0;
    int y0;
    float xsize;
    float ysize;
};

//=====
// Class:          PictObject
// Superclass:     None
// Description:    Generic class for picture categories
//=====

class PictObject
{
public:
    PictObject(Converter*);
    ~PictObject();

    PErrorCode read(ifstream&,SDTstring,int,int,int); // Read the picture
    virtual PErrorCode write(ofstream&) {return L_OK;} // Virtual method
}

```

1997年 09月 11日 16時 46分 /users/pclund/maap/m2p/category.h ページ 2

```

int lineNr() {return lNr;};
SDTstring Name() {return name;};
protected:
PErrorCode writeDefault(ofstream&); // Write default object
PErrorCode writeMaapLib(ofstream&); // Write an maapLib instance
PErrorCode writeStandard(ofstream&,int); // Write standard attrivutes

SDTstring name; // PictObject (class) name
SDTstring desc; // Description extracted from MAAP .pic file

int objNr; // Internal object number
int eventNr; // Associated MAAP event number
int catNr; // PictObject number
int isWater; // Is a water or air component ?
int eventObj; // Is this an object with user intervention ?
int noVis;
float angle,xpos,ypos; // Rotation angle, intial coordinat
Converter* conv; // Link to converter
Coord* coords; // Coordinates
int lNr;
};

//=====
// Class: Pipe
// Superclass: None
// Description: Class for general PIPE images
//=====

class Pipe : public PictObject
{
public:
Pipe(Converter*);
PErrorCode write(ofstream&);
};

//=====
// Class: HeatExCategory
// Superclass: None
// Description: Class for heat exchangers
//=====

class HeatEx : public PictObject
{
public:
HeatEx(Converter*);
PErrorCode write(ofstream&);
};

//=====
// Class: HxCoil
// Superclass: None
// Description: Class for heat exchanger coils
//=====

class HxCoil : public PictObject
{

```

1997年 09月 11日 16時 46分 /users/pclund/maap/m2p/category.h ページ 5

```
// Description: Class for hydrogen concentration
//=====
// Class:      Concrete
// Superclass: PictObject
// Description: Class for concrete ablation
//=====

//=====
// Class:      ContWater
// Superclass: PictObject
// Description: Class for containment vessel water level
//=====

//=====
// Class:      Corium
// Superclass: PictObject
// Description: Class for corium level
//=====

//=====
// Class:      ContFlag
// Superclass: PictObject
// Description: Class for containment failure flag
//=====

//=====
// Class:      Tube
// Superclass: PictObject
// Description: Class for tubes
//=====

//=====
// Class:      Compartment
// Superclass: PictObject
// Description: Class for compartments
//=====

class Compartment : public PictObject
{
public:
    Compartment(Converter*);
    PErrorCode write(ofstream&,int);
    void setCoord(float x,float y,int ix,int iy) {xsize = x; ysize = y; x0 = ix; y0 = iy;
    coords->setCoord(xsize,ysize,x0,y0);}
private:
    PErrorCode writeText(ofstream&,int); // Numeric values centered in the compartment
    float xsize;
    float ysize;
    int x0;
    int y0;
};

//=====
// Class:      DefaultPict
```

1997年 09月 11日 16時 46分 /users/pclund/maap/m2p/category.h ページ 6

```
// Superclass: PictObject
// Description: Class for default objects (temporary)
//=====
class DefaultPict : public PictObject
{
public:
    DefaultPict(Converter*,int);
// ~DefaultPict();
// PErrorCode read(ifstream&,SDTstring,int,int,int);
    PErrorCode write(ofstream&);
    int poly;
};

//=====
// Class:      TextPict
// Superclass: PictObject
// Description:
//=====

class TextPict
{
public:
    TextPict();
    ~TextPict();
    PErrorCode read(char*,int);
    PErrorCode write(ofstream&);
private:
    int xpos,ypos,nTxt;
    SDTstring egtxt,japtxt;
};

#endif

//===== endo of file =====
```

1997年 09月 11日 17時 24分 /users/pclund/maap/m2p/category.c ページ 1

```

//=====
// File name: compart.c
// Author: Per Christer Lund, PNC
// Creation date: June. 1997
// Description: Implementation of the compartment classes
//=====
//=====

#include "maap2picasso.h"

const SDTstring catNames[NCAT]=
{ "NONE0", "PIPE", "VALVE", "CONTROLROD", "RODOUT", "PUMP", "CONTFLLOOR", "CORENODE",
  "WATERLEVEL", "SPRAYPATTERN", "SHIELD", "WALLS", "H2CONC", "RODDRIVE", "CONCRETE",
  "CONTWATER", "CORIUM", "DRYWELLLEFT", "DRYWELLRIGHT", "WATERDRYWELLLEFT", "WATERDRYWELLRIGHT", "CONTFLAG",
  "TUBE", "COMPARTMENT" };

//=====
// Coord class
//=====

Coord::Coord(Converter* c, int nc)
{
    conv = c;
    nCoord = nc;
    if (nCoord > MAXCOORD) theLogger->error("Maximum coordinate number 100 exceeded");
    maxx = MINVAL;
    maxy = MINVAL;
    minx = MAXVAL;
    miny = MAXVAL;
    x0 = 0;
    y0 = 0;
    xsize = 1.0;
    ysize = 1.0;
}

Coord::~Coord()
{}

//=====

PErrorCode Coord::read(ifstream& inp)
{
    float fx,fy;
    for(int ic = 0; ic < nCoord; ic++){
        inp >> fx >> fy;
        x[ic] = (int) (fx * conv->xcf);
        y[ic] = (int) (conv->py2 - (fy * conv->ycf));

        if (x[ic] <= 0 || y[ic] <= 0)
            return theLogger->error("Illegal coordinate ");
        if (x[ic] > maxx) maxx = x[ic];
        if (x[ic] < minx) minx = x[ic];
        if (y[ic] > maxy) maxy = y[ic];
        if (y[ic] < miny) miny = y[ic];
    }
}

```

1997年 09月 11日 17時 24分 /users/pclund/maap/m2p/category.c ページ 2

```

xRef = (int)(maxx + minx)/2;
yRef = (int)(maxy + miny)/2;

return L_OK;
}

//=====

PErrorCode Coord::write(ofstream& of)
{
    of << "    numberofPoints = " << nCoord << ";" << endl;
    for(int ic = 0; ic < nCoord; ic++)
        of << "    point[" << ic << "] = (" << x[ic]*xsize + x0 << ", " << y[ic]*ysize +y0 << ");" << endl;
    return L_OK;
}

//=====

float Coord::getAngle()
{
    float deg;
    // Determine angle
    float xdif = (x[1] - x[0]);
    float ydif = (y[1] - y[0]);
    if      (ydif == 0) deg = (y[2]<y[0]) ? 90 : 270;
    else if (xdif == 0) deg = (x[2]>x[0]) ? 0 : 180;
    else          deg = (ydif<0)     ? 225 : 315;

    return deg;
}

//=====
// Generic category class
//=====

PictObject::PictObject(Converter* cd)
{
    conv = cd;
    angle = 0;      // Default rotation angle
    noVis = 0;
}

PictObject::~PictObject()
{
    if (coords) delete coords;
}

//=====

PErrorCode PictObject::read(ifstream& inp, SDTstring d, int nc,int enr,int onr)
{
    eventNr = enr;
}

```

1997年 09月 11日 17時 24分 /users/pclund/maap/m2p/category.c ページ 3

```

objNr    = onr;
desc     = d;
isWater = (! strstr(desc, "AIR"));

// Read coordinates
coords = new Coord(conv, nc);

return coords->read(inp);
}

//=====

PErrorCode PictObject::writeDefault(ofstream& outp)
{
CategoryDef* def = &conv->cdef[catNr];

// Foreground color
if (eventObj)
    outp << "    foregroundColour = 'eventColor(ievent[" <<
        eventNr << "]," << isWater << ")';" << endl;
else
    outp << "    foregroundColour = '" << def->lcol << "';" << endl;

// Foreground fill color
if (catNr == COMPARTMENT){}
else if (eventObj)
    outp << "    foregroundFillColour = 'eventColor(ievent[" <<
        eventNr << "]," << isWater << ")';" << endl;
else
    outp << "    foregroundFillColour = '" << def->fcoll << "';" << endl;

// Standard attributes
return writeStandard(outp, def->lth);
}

//=====

PErrorCode PictObject::writeStandard(ofstream& outp, int lwd)
{
outp << "    x = " << xpos << ";" << endl;
outp << "    y = " << ypos << ";" << endl;
if (!noVis) outp << "    visibility = 'visFlag[" << catNr << "]';" << endl;
outp << "    backgroundColour = -1;" << endl;
outp << "    lineWidth      = " << lwd << ";" << endl;
outp << "    linePattern    = 'SolidPattern';" << endl;
outp << "    fillPattern    = 'SolidPattern';" << endl;
outp << "    lineStyle      = 'SolidPattern';" << endl;
outp << "    xReference     = " << coords->xRef << ";" << endl;
outp << "    yReference     = " << coords->yRef << ";" << endl;
outp << "    xScaleFactor   = 1;" << endl;
outp << "    yScaleFactor   = 1;" << endl;
outp << "    rotationAngle  = " << angle << ";" << endl;

return L_OK;
}

```

1997年 09月 11日 17時 24分 /users/pclund/maap/m2p/category.c ページ 4

```

//=====
PErrorCode PictObject::writeMaapLib(ofstream& outp)
{
    outp << "    x = " << xpos << ";" << endl;
    outp << "    y = " << ypos << ";" << endl;
    outp << "    visibility = 'visFlag[" << catNr << "]';" << endl;
    outp << "    xReference = " << coords->x[0] << ";" << endl;
    outp << "    yReference = " << coords->y[0] << ";" << endl;
    outp << "    xScaleFactor = 1;" << endl;
    outp << "    yScaleFactor = 1;" << endl;
    outp << "    rotationAngle = " << angle << ";" << endl;
    return L_OK;
}

//=====
// Pipe PictObject class
//=====

Pipe::Pipe(Converter* cd) : PictObject(cd)
{
    catNr = PIPE;
    name = "Pipe";
    eventObj = 1;
}

//=====

PErrorCode Pipe::write(ofstream& outp)
{
    outp << endl;
    outp << "line pipe" << objNr << " \t//\n" << desc << endl;
    outp << " {" << endl;
    coords->write(outp);
    writeDefault(outp);
    outp << " }" << endl;
    return L_OK;
}

//=====
// Heat Exchanger category class
//=====

HeatEx::HeatEx(Converter* cd) : PictObject(cd)
{
    catNr = PIPE;
    name = "HeatExchanger";
    eventObj = 1;
}

//=====

PErrorCode HeatEx::write(ofstream& outp)
{
    xpos = coords->minx;
}

```

1997年 09月 11日 17時 24分 /users/pclund/maap/m2p/category.c ページ 5

```

ypos = coords->miny;
outp << endl;
outp << " instance maapLib.HeatExchanger " << name << objNr << "\t// " << desc << endl;
outp << " {" << endl;
outp << "     attribute eventId = " << eventNr << ";" << endl;
outp << "     attribute description = \" " << desc << "\";" << endl;
writeMaapLib(outp);
outp << " }" << endl;

return L_OK;
}

//=====
// Pump category class
//=====

Pump::Pump(Converter* cd) : PictObject(cd)
{
    catNr = PUMP;
    name = "Pump";
    eventObj = 1;
}

//=====

PErrorCode Pump::write(ofstream& outp)
{
    xpos = coords->minx;
    ypos = coords->miny;
    outp << endl;
    outp << " instance maapLib.Pump " << name << objNr << "\t// " << desc << endl;
    outp << " {" << endl;
    outp << "     attribute eventId = " << eventNr << ";" << endl;
    outp << "     attribute description = \" " << desc << "\";" << endl;
    outp << "     attribute waterPump = " << isWater << ";" << endl;
    writeMaapLib(outp);
    outp << " }" << endl;

    return L_OK;
}

//=====
// Valve category class
//=====

Valve::Valve(Converter* cd) : PictObject(cd)
{
    catNr = VALVE;
    name = "Valve";
    eventObj = 1;
}

//=====

PErrorCode Valve::write(ofstream& outp)
{

```

1997年 09月 11日 17時 24分 /users/pclund/maap/m2p/category.c ページ 6

```

xpos = coords->x[0];
ypos = coords->y[0];
angle = coords->getAngle();
outp << endl;
outp << " instance maapLib.Valve " << name << objNr << "\t// " << desc << endl;
outp << " {" << endl;
outp << "     attribute eventId = " << eventNr << ";" << endl;
outp << "     attribute description = \" " << desc << "\";" << endl;
outp << "     attribute waterValve = " << isWater << ";" << endl;
writeMaapLib(outp);
outp << " }" << endl;

return L_OK;
}

//=====================================================================
// HxCoil class
//=====================================================================

HxCoil::HxCoil(Converter* cd) : PictObject(cd)
{
    catNr = PIPE;
    name = "HxCoil";
    eventObj = 0;
}

//=====================================================================

PErrorCode HxCoil::write(ofstream& outp){return L_OK; }

//=====================================================================
// CoreNode category class
//=====================================================================

CoreNode::CoreNode(Converter* cd) : PictObject(cd)
{
    catNr = CORENODE;
    name = "nodeSect";
    eventObj = 0;
}

//=====================================================================

PErrorCode CoreNode::write(ofstream& outp)
{
    int i,icol,ix,iy;
    float width = coords->maxx - coords->minx;
    float height = (coords->maxy - coords->miny)/NODESECT;
    xpos = coords->minx;
    // Write node sectors
    outp << "// Node sections for core node " << objNr << " (GRAAPH no. "
        << eventNr << ")\n" << endl;

    for(i = 0; i < NODESECT; i++){
        ypos = coords->miny + (i * height);
}

```

1997年 09月 11日 17時 24分 /users/pclund/maap/m2p/category.c ページ 7

```

    icol = objNr * NODESECT + i;
    outp << " rectangle nodeSect" << objNr << " " << i << "\n {" << endl;
//    outp << "     width = 'nodeWidth[" << icol << "] * " << width << ";" << endl;
    outp << "     width = " << width << ";" << endl;
    outp << "     height = " << height << ";" << endl;
    outp << "     foregroundColour = 'nodeColor[" << icol << "]';" << endl;
    outp << "     foregroundFillColour = 'nodeColor[" << icol << "]';" << endl;
    writeStandard(outp,0);
    outp << " }" << endl;
    outp << endl;
}

return L_OK;
}

//=====
// Water Level category class
=====

WaterLevel::WaterLevel(Converter* cd) : PictObject(cd)
{
    catNr = WATERLEVEL;
    name = "WaterLevel";
    eventObj = 1;
}

//=====

PErrorCode WaterLevel::write(ofstream& outp)
{
    // Write water level object
    outp << " polygon waterLevel" << objNr << " \t// " << desc << endl;
    outp << " {" << endl;
    coords->write(outp);
    outp << "     foregroundColour = 'white';" << endl;
    outp << "     backgroundColour = -1;" << endl;
    outp << "     foregroundFillColour = 'blue';" << endl;
    writeStandard(outp,1);
    outp << " }" << endl;
    outp << endl;

    // Write the shadow object
    xpos = coords->minx;
    ypos = coords->miny;
    float dely = coords->maxy - coords->miny;
    noVis = 1;
    outp << " rectangle shadowPict" << objNr << endl;
    outp << " {" << endl;
    outp << "     width = " << coords->maxx - coords->minx << ";" << endl;
    outp << "     height = '(1.0-waterLevel[" << eventNr << "]) * " << dely << ";" << endl;
    outp << "     visibility = 'lvisFlag';" << endl;
    outp << "     foregroundFillColour = 'black';" << endl;
    outp << "     backgroundFillColour = -1;" << endl;
    outp << "     foregroundColour = 'black';" << endl;
    writeStandard(outp,0);
    outp << " }" << endl;
}

```

1997年 09月 11日 17時 24分 /users/pclund/maap/m2p/category.c ページ 8

```

outp << endl;

// Write the border object
outp << " line borderLine" << objNr << endl;
outp << " {" << endl;
coords->write(outp);
outp << " foregroundColour = 'white';" << endl;
writeStandard(outp, 0);
outp << " }" << endl;

return L_OK;
}

//=====
// DefaultPict category class
//=====

DefaultPict::DefaultPict(Converter* cd,int cnr) : PictObject(cd)
{
    catNr = cnr;
    name = catNames[catNr];
    eventObj = 1;
    poly = cd->cdef[cnr].ctyp;
}

//=====

PErrorCode DefaultPict::write(ofstream& outp)
{
    // Write output header
    if (poly) outp << " polygon ";
    else        outp << " line ";
    outp << name << objNr << "\t//\" << desc << " (GRAAPH no. " << eventNr << ")" << endl;
    outp << " {" << endl;
    coords->write(outp);
    outp << " foregroundColour = '" << conv->cdef[catNr].lcol << "';" << endl;
    outp << " foregroundFillColour = '" << conv->cdef[catNr].fcol << "';" << endl;
    writeStandard(outp,conv->cdef[catNr].lth);
    outp << " }" << endl;

    return L_OK;
}

//=====
// Cooling system category class
//=====

Cooling::Cooling(Converter* cd) : PictObject(cd)
{
    catNr = PUMP;
    name = catNames[catNr];
    eventObj = 1;
}

```

1997年 09月 11日 17時 24分 /users/pclund/maap/m2p/category.c ページ 9

```

//=====
PErrorCode Cooling::write(ofstream& outp)
{
    outp << " polygon " << name << objNr << "\t//"
        << desc << " (GRAAPH no. " << eventNr << ")" << endl;
    outp << " (" << endl;
    coords->write(outp);
    outp << " foregroundColour = 'white';" << endl;
    outp << " foregroundFillColour = 'eventColor(ievent[" <<
        eventNr << "], " << isWater << ")';" << endl;
    writeStandard(outp, conv->cdef[catNr].lth);
    outp << " )" << endl;

    return L_OK;
}

//=====
// TextPicture category class
//=====

TextPict::TextPict(){}
TextPict::~TextPict() {}

//=====

PErrorCode TextPict::read(char* cdum, int ix)
{
    nTxt = ix;
    xpos = atoi(strtok(cdum, ","));
    ypos = atoi(strtok(NULL, ","));
    egtxt = strtok(NULL, ",");
    char* jptxt = strtok(NULL, ",");
    charstrip(jptxt, japtxt);
    return L_OK;
}

//=====

PErrorCode TextPict::write(ofstream& outp)
{
    outp << endl;
    outp << " char kanji" << nTxt << "[100] = \" " << japtxt << "\";" << endl;
    outp << " text textObject" << nTxt << "\n {" << endl;
    outp << "     x = " << xpos << ";" << endl;
    outp << "     y = " << ypos << ";" << endl;
    outp << "     format = \"%s\\";" << endl;
    outp << "     theText = '(theLanguage) ? \" " << egtxt << "\" : kanji" << nTxt << "';" << endl;
    outp << "     visibility = 1;" << endl;
    outp << "     foregroundColour = 'white';" << endl;
    outp << "     backgroundColour = -1;" << endl;
    outp << "     theFont = '(theLanguage) ? font0:jap6';" << endl;
    outp << "     xReference = " << xpos << ";" << endl;
    outp << "     yReference = " << ypos << ";" << endl;
}

```

1997年 09月 11日 17時 24分 /users/pclund/maap/m2p/category.c ページ 10

```
    outp << "    xScaleFactor = 1;" << endl;
    outp << "    yScaleFactor = 1;" << endl;
    outp << "    rotationAngle = 0;" << endl;
    outp << "\n" << endl;

    return L_OK;
}

//===== end of file =====
```

1997年 09月 24日 15時 39分 /users/pclund/maap/maapsrc/picasso.c ページ 1

```
*****
File name:      picasso.c
Author:         Per Christer Lund, PNC
Creation date: July 1997
Description:    Provides interface between the MAAP code and
                the PICASSO-3 GUI

Functions:      int init_pic()      - initializes and controls the PICASSO API
                only function necessary to call from MAAP
                void up(..)        - called when the API starts, creates variables
                void down(..)     - called if the API is lost
                int repeater()    - calls the MAAP loop and updates PICASSO variables
                int restart()
                int halt()

*****
#include "picasso.h"

#define NPICCALL 1000 /* Milliseconds between each PICASSO call */
#define NPRINTOUT 20   /* Number of iterations between each printout */
#define MAAPLOOP 1     /* Number of MAAP calls pr. PICASSO update */

/* Picasso variable IDs */
int nPicId;
int32 picassoIds[MAXPICID];

/* Compartment data */
struct CompartData compData[MAXCOMPDATA];

/* Scalar data */
struct MaapVariable maapVar[MAXVARIABLE];

/* Parameters to send to PICASSO-3 GUI */
int timeVar;           /* Time variable used for trends */
float maapTime;        /* MAAP transient time */
float levels[MAXSHAPES+1]; /* Water levels */
float nodeWidth[MAXNODECOLOR]; /* Core fuel mass fractions */

int events[MAXSHAPES+1]; /* Shape events */
int nodeColor[MAXNODECOLOR]; /* Core fuel temperature color codes */
char statusText[TEXTLEN]; /* Status text */
char timeTexts[5][TEXTLEN]; /* Time strings */

/* Other global variables */
int nVariable;          /* Number of variables to be sent to Picasso */
int nIter;               /* Overall number of iterations */
int nPrint;              /* Printout counter */
int isRunning;           /* Running flag */
float endTime;           /* End simulation time */
clock_t CPUuse;          /* Total CPU usage */
clock_t CPUiter;         /* CPU usage pr. iteration */
float CPUMaap;           /* CPU usage from MAAP */
float tLast;              /* Last timestep */

*****
int init_pic(float* tend)
```

1997年 09月 24日 15時 39分 /users/pclund/maap/maapsrc/picasso.c ページ 2

```
{
    int i;
    endTime = *tend;           /* End time for simulation */
    CPUuse = clock();          /* Total CPU usage */
    strncpy(statusText,title.title,40);

/* listParameters(); return 0; */

printf("Calling PICASSO init program \n");
/* Initialize the PICASSO API */
if (PfInitialize(PROCNAME,NULL,"rtm",NULL,100,1,connectToPicasso,disconnectFromPicasso) != OK) {
    printf("Error initializing PICASSO-3 connection \n");
    printf("Returning control to normal MAAP procedure\n");
    return 1;
}
PfSetProcessHandler(repeater, NPICCALL);
printf("Initialized PICASSO-3 API process \"%s\"\n",PROCNAME);

// Initialize variables
nIter = 0; nPrint = 0; timeVar=0;
isRunning = 1;
CPUiter = clock();
CPUmaap = 0;

PfExecute(NULL,"strcpy(trendTitle,\"MAAP variable trending\")");

/* Enter the main loop */
printf("Entering the main loop\n");
PfMainLoop();

return 0;
}

/*=====
void connectToPicasso(idum,cdum/*int32 idum,char* cdum*/)
int32 idum;
char* cdum;
{
    int i;
    nPicId = 0;
    printf("Creating PICASSO-3 API variables\n");

    /* Setting up compartment variables */
    if (! setupCompVar()) return;

    /* Setting up scalar maap variables */
    if (! setupMaapVar()) return;

    /* Creating global time variable */
    picassoIds[nPicId] = PfCreateVar("timeVar",PfCInt, NULL, 1, &timeVar);
    nPicId++;

    /* Creating the PICASSO-3 variables */
    picassoIds[nPicId]= PfCreateArray("statusText",PfCUncsignedChar,TEXTLEN,NULL,1,statusText);
    nPicId++;
}
```

1997年 09月 24日 15時 39分 /users/pclund/maap/maapsrc/picasso.c ページ 3

```

/* Time string */
picassoIds[nPicId] = PfCreateArray("statusString", PfCUndefinedChar, TEXTLEN, NULL, 1, timeTexts[0]);
nPicId++;
picassoIds[nPicId] = PfCreateArray("timeString1", PfCUndefinedChar, TEXTLEN, NULL, 1, timeTexts[1]);
nPicId++;
picassoIds[nPicId] = PfCreateArray("timeString2", PfCUndefinedChar, TEXTLEN, NULL, 1, timeTexts[2]);
nPicId++;
picassoIds[nPicId] = PfCreateArray("timeString3", PfCUndefinedChar, TEXTLEN, NULL, 1, timeTexts[3]);
nPicId++;
picassoIds[nPicId] = PfCreateArray("timeString4", PfCUndefinedChar, TEXTLEN, NULL, 1, timeTexts[4]);
nPicId++;

/* Get the MAAP internal transient time */
picassoIds[nPicId] = PfCreateVar("maapTime", PfCFloat, NULL, True, &maapTime);
nPicId++;

/* Create the event array */
picassoIds[nPicId] = PfCreateArray("ievent", PfCInt, MAXSHAPES, NULL, True, events);
nPicId++;

/* Create the water level array */
picassoIds[nPicId] = PfCreateArray("waterLevel", PfCFloat, MAXSHAPES, NULL, True, levels);
nPicId++;

/* Create the fuel node color (temperature) array */
picassoIds[nPicId] = PfCreateArray("nodeColor", PfCInt, MAXNODECOLOR, NULL, True, nodeColor);
nPicId++;

/* Create the fuel node width (mass fraction) array */
picassoIds[nPicId] = PfCreateArray("nodeWidth", PfCInt, MAXNODECOLOR, NULL, True, nodeWidth);
nPicId++;

/* Check variables */
for(i=0;i<nPicId;i++) if (picassoIds[i] == NullId) printf("Error making some Picasso variable\n");

/* Flush the created variables */
PfFlushCreateVar();
PfFlush();
}

/*
void disconnectFromPicasso(idum,cdum/*int32 idum,char* cdum*/
int32 idum;
char* cdum;
{
  printf("Lost connection to PICASSO - shutting down...\n");
  PfEndLoop();
}

/*
#define MINNTEMP 1000 /* Degrees Kelvin */
#define NTEMPSTEP 40 /* Temp step pr. color change */
#define FIRSTCOL 89 /* First color in the PICASSO resource definitions */

```

1997年 09月 24日 15時 39分 /users/pclund/maap/maapsrc/picasso.c ページ 4

```

int32 repeater(istp/*int32 istp*/
int32 istp;
{
    int i,j,k,img = 0,idison = 1,icol,istep,guiRun;
    float tdzpic=1.0,col,ttot;
    float tuse,tstp,trest;
    char* res;
    clock_t tmaap; /* CPU usage counters */

    /* Update the trancient time */
    maapTime = (float) time2.timd;
    timeVar++;
    updateTime(maapTime,timeVar);

    /* Check if the user has halted execution of MAAP */
    guiRun = atoi(PfExecute(NULL, "maapIsRunning"));
    if (isRunning && !guiRun) return halt();
    if (! isRunning && guiRun) restart();
    isRunning = guiRun;
    if (! isRunning) return OK;

    /* Call the main MAAP iteration loop */
    tmaap = clock();
    for(i = 0; i < MAAPLOOP; i++)
        picloop(); /* FORTRAN subroutine (pic_loop.f) */
    pic3cnt.intp3 = 0;
    pic_api(); /* FORTRAN subroutine (pic_api.f) */
    CPUmaap += ((float) (clock() - tmaap)) * 1e-6;

    /* Update the event and water level array */
    for(i = 1; i <= MAXSHAPES; i++) {
        events[i] = flgs.i_event[i-1];
        levels[i] = pic3lev.zlev[i-1];
    }

    /* Update and compute the core node colors and shapes */
    for(i = 0; i < MAXNODES; i++) {
        for(j = 0; j < MAXNODESECT; j++) {
            col = pic3nod.tpic[i][j][0];
            icol = FIRSTCOL + (int) ((col - MINNTEMP)/NTEMPSTEP);
            if (col < 600) icol = GREY;
            nodeColor[i*MAXNODESECT + j] = icol;
            nodeWidth[i*MAXNODESECT + j] = pic3nod.fmass[i][j][k];
            if (nodeWidth[i*MAXNODESECT + j] >.1.0){
                printf("%i %j: %.2f\n", i, j, nodeWidth[i*MAXNODESECT + j]);
                nodeWidth[i*MAXNODESECT + j] = 1.0;
            }
        }
    }

    /* Update compartment data */
    for(i = 0; i < MAXCOMPDATA; i++) {
        if (compData[i].chFlag) resetVector(&compData[i],i);
        else updateVector(&compData[i],i);
    }
}

```

1997年 09月 24日 15時 39分 /users/pclund/maap/maapsrc/picasso.c ページ 5

```

/* Update maap variables */
for(i = 0; i < nVariable; i++) {
    if (maapVar[i].chFlag)    resetMaapVar(&maapVar[i],i);
    else                      updateMaapVar(&maapVar[i],i);
}

/* Inform PICASSO about the updates */
for(i = 0; i < nPicId; i++)    PfSend(picassoIds[i]);
PfFlush();

/* Check the time for terminating execution */
if (time2.timd > endTime) {
    printf("Terminating execution at trancient time %.2f\n",time2.timd);
    PfEndLoop();
}

/* Print some status line for each 10 iteration */
if (nIter == 0 || nPrint >= NPRINTOUT) printOutput();
nIter++; nPrint++;

return OK;
}

/***********************/

void printOutput()
{
    float ttot,tuse,tstp,trest;

    ttot = ((float) (clock() - CPUiter)) * 1e-6;
    tuse = ((float) (clock() - CPUuse)) * 1e-6;
    printf("\n*** Iteration %i\n",nIter+1);
    printf("      Real time: %.2f - MAAP time %.2f [sec] (%.2f%%). MAAP CPU usage %.2f%%\n",
           tuse,maapTime,maapTime*100/endTime,CPUmaap*100/ttot);
    tstp = tuse/maapTime;
    trest = tstp * (endTime - maapTime);
    printf("      MAAP/Realtime: %.2f [sec]; estimated time left: ",1.0/tstp);
    printTime(trest);

    nPrint = 0;
    CPUiter = clock();
    CPUmaap = 0;
}

/***********************/

void printTime(float sec)
{
    int hour,min;
    float rsec;
    hour = (int) (sec/3600);
    rsec = sec - (hour*3600);
    min = (int) (rsec/60);
    rsec = rsec - (min*60);
}

```

1997年 09月 24日 15時 39分 /users/pclund/maap/maapsrc/picasso.c ページ 6

```

    printf("%2i:%2i:%5.2f\n",hour,min,rsec);
}

/****************************************/
int restart()
{
    int i;
    printf("Restarting...\n");

    /* Update the event and water level array */
    for(i = 1; i <= MAXSHAPES; i++) {
        if (events[i] != flgs.i_event[i-1]) {
            flgs.i_event_old[i-1] = -2;
            printf("Event no. %i changed from %i to %i \n",i,flgs.i_event[i-1],events[i]);
            flgs.i_event[i-1] = events[i];
        }
    }

    pic3cnt.intp3 = 1;
    pic_api();      /* This is a FORTRAN subroutine (pic_api.f) */

    sprintf(statusText,"MAAP restarted by user at %.2f sec",maapTime);

    for(i = 0; i < nPicId; i++)    PfSend(picassoids[i]);
    PfFlush();

    return OK;
}

/****************************************/

int halt()
{
    int i;
    printf("Halting...\n");
    sprintf(statusText,"MAAP halted by user at %.2f sec",maapTime);

    isRunning = 0;
    for(i = 0; i < nPicId; i++)    PfSend(picassoids[i]);
    PfFlush();

    return OK;
}

/****************************************/
/* Setting up the compartment structures and PICASSO structs */
/****************************************/

int setupCompVar()
{
    int i;

    // Declaring the PICASSO struct */
    PfBeginRecord("CompartData");

```

1997年 09月 24日 15時 39分 /users/pclund/maap/maapsrc/picasso.c ページ 7

```

PfAddField("CompartData", "name", NAMELEN, PfCChar, NULL, NULL);
PfAddField("CompartData", "unit", UNITLEN, PfCChar, NULL, NULL);
PfAddField("CompartData", "indx", NNODX, PfCInt, NULL, NULL);
PfAddField("CompartData", "data", NNODX, PfCFloat, NULL, NULL);
PfAddField("CompartData", "color", NNODX, PfCInt, NULL, NULL);
PfAddField("CompartData", "minVal", 1, PfCFloat, NULL, NULL);
PfAddField("CompartData", "stepVal", 1, PfCFloat, NULL, NULL);
PfAddField("CompartData", " minValue", 1, PfCFloat, NULL, NULL);
PfAddField("CompartData", " maxValue", 1, PfCFloat, NULL, NULL);
PfAddField("CompartData", "iunit", 1, PfCInt, NULL, NULL);
PfAddField("CompartData", "icf", 1, PfCInt, NULL, NULL);
PfAddField("CompartData", "chFlag", 1, PfCInt, NULL, NULL);
PfAddField("CompartData", "failFlag", 1, PfCInt, NULL, NULL);
PfAddField("CompartData", "message", 100, PfCChar, NULL, NULL);
PfEndRecord("CompartData");
if ( (picassoIds[nPicId]
      = PfCreateArray("compData", PfCRecord, MAXCOMPDATA, "CompartData", TRUE, compData)) == NullId) {
    printf("Error making Picasso variable compData[%i]\n", MAXCOMPDATA);
    return 0;
}
nPicId++;

// Register the structs with MAAP data. Default data.
if (getVector(&compData[0], "TGRB", 40, 10.0, 0) != 0)      return 0; /* Temperatures */
if (getVector(&compData[1], "MH2RB", 0, 1.0, 1) != 0)      return 0;
if (getVector(&compData[2], "PPSTRB", 0, 1000, 2) != 0)     return 0;
if (getVector(&compData[3], "MFSTRB", 0, 0.01, 3) != 0)    return 0;

return 1;
}

/*****************************************/
/* Setting up the compartment structures and PICASSO structs */
/*****************************************/

int setupMaapVar()
{
    int i;

    // Declaring the PICASSO struct */
    PfBeginRecord("MaapVariable");
    PfAddField("MaapVariable", "name", NAMELEN, PfCChar, NULL, NULL);
    PfAddField("MaapVariable", "unit", UNITLEN, PfCChar, NULL, NULL);
    PfAddField("MaapVariable", "indx", 1, PfCInt, NULL, NULL);
    PfAddField("MaapVariable", "data", 1, PfCFloat, NULL, NULL);
    PfAddField("MaapVariable", " minValue", 1, PfCFloat, NULL, NULL);
    PfAddField("MaapVariable", " maxValue", 1, PfCFloat, NULL, NULL);
    PfAddField("MaapVariable", "iunit", 1, PfCInt, NULL, NULL);
    PfAddField("MaapVariable", "icf", 1, PfCInt, NULL, NULL);
    PfAddField("MaapVariable", "chFlag", 1, PfCInt, NULL, NULL);
    PfAddField("MaapVariable", "failFlag", 1, PfCInt, NULL, NULL);
    PfAddField("MaapVariable", "message", 100, PfCChar, NULL, NULL);
    PfEndRecord("MaapVariable");
    if ( (picassoIds[nPicId]
          = PfCreateArray("maapVar", PfCRecord, MAXVARIABLE, "MaapVariable", TRUE, maapVar)) == NullId) {
        printf("Error making Picasso variable maapVar[%i]\n", MAXVARIABLE);
    }
}

```

1997年 09月 24日 15時 39分 /users/pclund/maap/maapsrc/picasso.c ページ 8

```

    return 0;
}
nPicId++;

// Register the structs with MAAP data. Default data.
if (getVariable(&maapVar[0], "TGRB(13)", 0, 300, 350) != 0)      return 0;
if (getVariable(&maapVar[1], "MH2RB(13)", 1, 0, 0.1) != 0)      return 0;
if (getVariable(&maapVar[2], "PPSTRB(13)", 2, 0, 10000) != 0)     return 0;
if (getVariable(&maapVar[3], "MFSTRB(13)", 3, 0, 0.1) != 0)      return 0;
nVariable = 4;

return 1;
}

/*****************/
void updateTime(float tm,int tnr)
{
    int i,indx2,nStep = 62;
    float nTime, dT = tm - tLast;
    int ts = atoi(PfExecute(NULL,"::maap.varTrend.theTrend.timeSpan"));

    nStep = (int) (ts/4);
    dT = (tnr>0)? tm / tnr : 0;

    /* Make time labels*/
    for(i = 0; i < 5; i++){
        nTime = tm - (dT * i * nStep);
        if (nTime > 0) makeTimeString(timeTexts[i],nTime);
        else timeTexts[i][0] = '\0';
    }
}

/*****************/
void makeTimeString(char* tstr,float sec)
{
    int hour,min;
    float rsec;
    hour = (int) (sec/3600);
    rsec = sec - (hour*3600);
    min = (int) (rsec/60);
    rsec = rsec - (min*60);
    sprintf(tstr,"%2i:%2i:%6.3f",hour,min,rsec);
}

/*-----*/

```

1997年 10月 30日 09時 27分 /users/pclund/maap/maapsrc/picasso.h ページ 1

```
*****
File name:      picasso.h
Author:         Per Christer Lund, PNC
Creation date: August 1997
Description:    Constants, external structs, external functions for the MAAP/PICASSO interface
*****  

#ifndef PICASSO_H
#define PICASSO_H  

/* Includes */
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <api/api.h>      /* Including PICASSO API */  

/* Declaration of common data structures */
#include "comdata.h"  

/* Constants declarations */
#define PROCNAME "maap"
#define MAXCOMPDATA 4      /* Maximum number of compartment data simultaneously */
#define MAXVARIABLE 50     /* Maximum number of simultaneous scalar variables */
#define MAXPICID 100       /* Maximum number of PICASSO variables (structs etc) */
#define NAMELEN 32
#define UNITLEN 8  

#define ICST    8
#define MAXNODES 6
#define MAXNODESECT 18
#define MAXNODECOLOR MAXNODES*MAXNODESECT
#define NNODX  40          /* Number of compartment nodes */
#define NWX   120
#define NCMX  10
#define NCOMPS 13  

#define LASTCOLOR 25  

/* PICASSO-3 color codes */
#define GREY   27
#define WHITE  26
#define RED    1
#define BLUE   18
#define TEXTLEN 100  

/* Constants etc. */
#define K2C 273.14          /* Kelvin to Celcius */  

/* Declare FORTRAN COMMON blocks as external structs */
#define MAXSHAPES 350      /* Maximum number of shapes */
#define CMAXRAD 10
#define CMAXAX  20
```

1997年 10月 30日 09時 27分 /users/pclund/maap/maapsrc/picasso.h ページ 2

```

extern struct {      /* Time data (matr.f) */
    double timd,timfpd;
} time2;

extern struct {      /* The FLGS common block (apicl.f) */
    int i_event[MAXSHAPES];
    int i_event_old[MAXSHAPES];
    int i_event_auto[MAXSHAPES];
} flgs;

extern struct {      /* Core node temp. and mass fractions (apicl.f) */
    int cmaxrad,cmaxax,icst;
    float tpic[ICST][CMAXAX][CMAXRAD];
    float fmass[ICST][CMAXAX][CMAXRAD];
} pic3nod;

extern struct {      /* Water and corium levels (apicl.f)*/
    float zlev[MAXSHAPES];
} pic3lev;

extern struct {      /* Execution controls (apicl.f) */
    int intp3;
} pic3cnt;

extern struct {      /* Titles */
    char titl[80];
    char tit2[80];
    char tit3[80];
} title;

//=====
// PICASSO structs
//=====

// Declaration of functions
// FORTRAN Subroutines
extern void picloop();          /* The main MAAP cycle (one cycle) */
extern void pic_api();           /* The modified APICL (apicl.f) */
extern void comvar(char*,char*,int*,int*,int*,int*,int*,int*,int,int); // Get variable pointer etc.
extern void comval(int*,int*,float*,int*,char*,int);                  // Get variable value

// C Functions
int setupCompVar();
int getVector(struct CompartData*,char*,float,float,int);
int updateVector(struct CompartData*,int);
int resetVector(struct CompartData*,int);

int setupMaapVar();
int getVariable(struct MaapVariable* var,char* name,int indx,float min, float max);
int updateMaapVar(struct MaapVariable*,int);
int resetMaapVar(struct MaapVariable*,int);

int restart();
int halt();

```

1997年 10月 30日 09時 27分 /users/pclund/maap/maapsrc/picasso.h ページ 3

```
void printTime(float);
void printOutput();
void updateTime(float,int);
void makeTimeString(char*,float);

/* Picasso functions */
void connectToPicasso(/*int32,char**/);
void disconnectFromPicasso(/*int32,char**/);
int32 repeater(/*int32*/);

#endif

***** end of file *****
```

1997年 09月 22日 14時 13分 /users/pclund/maap/maapsrc/parameter.c ページ 1

```
*****
File name: parameter.c
Author: Per Christer Lund, PNC
Creation date: July 1997
Description: Provides interface between the MAAP code and
the PICASSO-3 GUIProvides methods for extracting parameters from MAAP

Functions:
*****
#include "picasso.h"

*****
#define MAXCBLK 158      // Number of common blocks
#define MAXPAR 3920
#define NLEN 32

int listCommonBlocks()
{
    int ip=1,icom,invar,inloc,iopc=2,totvar = 0,totloc = 0;
    char cname[NLEN];

    printf("\n*** Common blocks:\n");

    for(icom = 1; icom < MAXCBLK; icom++) {
        com(cname,&icom,&invar,&inloc,&iopc,NLEN);
        printf("%3i: %s: %4i var, %6i floats \n",icom,cname,invar,inloc);
        totvar += invar;
        totloc += inloc;
    }

    listCompVector();

    printf("Total number of variables %i\n",totvar);
    printf("Total number of floats %i\n",totloc);

    return 1;
}

*****
int listParameters()
{
    int i,ip=1,indx,icom,ivar,iloc,icf,iopc=2,ierr,iunit=1;
    char vname[NAMELEN],cname[NAMELEN],unit[UNITLEN];
    float data;

    FILE* outp = fopen("var.lst", "w");

    fprintf(outp, "\n*** Variables:\n");

    for(i = 1; i <= MAXPAR; i++){
        varnml(&i,vname,NAMELEN);
        comvar(cname,vname,&indx,&icom,&ivar,&iloc,&icf,&iopc,&ierr,NAMELEN,NAMELEN);
        if (ierr != 0) printf("Error \n");
    }
}
```

1997年 09月 22日 14時 13分 /users/pclund/maap/maapsrc/parameter.c ページ 2

```

comval(&iunit,&indx,&data,&icf,unit,UNITLEN);
if (unit[0] == ' ') unit[0]='-';
fprintf(outp,"%s %s %.2f\n",vname,unit,data);
}
fprintf(outp,"\\n%i variables\\n",MAXPAR);
fclose(outp);
return 1;
}

/*****
#define VARLEN 15
int listScalar()
{
    int i,ip=1;
    char vname[VARLEN];

    printf("\\n*** Scalars:\\n");

    for(i = 1; i <= MAXPAR; i++){
        varnm1(&i,vname,VARLEN);
        if (! strchr(vname,'(')) {
            printf("%s ",vname); ip++;
        }
        if (ip == 5) { ip = 0; printf("\\n");}
    }
    return 1;
}

/*****
int listCompVector()
{
    int i,ip=0,npar=0;
    char vname[VARLEN];

    printf("\\n*** Compartment vectors:\\n");

    for(i = 1; i <= MAXPAR; i++){
        varnm1(&i,vname,VARLEN);
        if (strstr(vname, "(40)")){printf("%s ",vname);ip++;npar++;}
        if (ip == 5) { ip = 0; printf("\\n");}
    }

    printf("\\nTotal number of compartment variables %i (%i floats)\\n",npar,npar*40);

    return 1;
}

*****
char scalarNames[MAXVARIABLE][NAMELEN];

int getVariable(struct MaapVariable* var,char* name,int indx,float min, float max)
{
    char cname[NAMELEN], unit[UNITLEN], *cnl;
    int icom,ivar,iloc,iopt = 2,ierr;

```

1997年 09月 22日 14時 13分 /users/pclund/maap/maapsrc/parameter.c ページ 3

```

var->iunit = 1;

/* Get the variable's data */
comvar(cname, name, &var->indx, &icom, &ivar, &iloc, &var->icf, &iopc, &ierr, NAMELEN, NAMELEN);
if (ierr != 0){
    printf("Error code %i (Illegal variable name %s)", ierr, name);
    return ierr;
}

/*Get the variable's value */
comval(&var->iunit, &var->indx, &var->data, &var->icf, unit, UNITLEN);

/* Insert a null character */
if ((cnl = strchr(unit, ' '))) *cnl = '\0';

/* Update the variable */
strcpy(var->name, name);
strcpy(var->unit, unit);
var->maxValue = max;
var->minValue = min;

printf("Defined variable %s value %.2f [%s]\n",
       var->name, var->data, var->unit);

strcpy(scalarNames[indx], var->name);
return ierr;
}

/***********************/

int updateMaapVar(struct MaapVariable* var,int indx)
{
    char unit[UNITLEN];
    /*Get the variable's value */
    comval(&var->iunit, &var->indx, &var->data, &var->icf, unit, UNITLEN);
    return 1;
};

int resetMaapVar(struct MaapVariable* var,int indx)
{
    char vname[NAMELEN], cname[NAMELEN];
    int icom, ivar, iloc, iopc, ierr, iptr, icf;
    iopc = 2;

    sprintf(vname, "%s", var->name);
    comvar(cname, vname, &iptr, &icom, &ivar, &iloc, &icf, &iopc, &ierr, NAMELEN, NAMELEN);
    if (ierr != 0){
        printf("Error code %i (Illegal variable name %s)", ierr, vname);
        strcpy(var->name, scalarNames[indx]);
        printf("Restoring old variable name %s\n", var->name);
        sprintf(var->message, "Undefined MAAP variable %s, restoring %s", vname, var->name);
        var->failFlag = 1;
        return ierr;
    }
    var->indx = iptr;
}

```

1997年 09月 22日 14時 13分 /users/pclund/maap/maapsrc/parameter.c ページ 4

```
var->icf      = icf;
updateMaapVar(var,indx);
var->chFlag = 0;

strcpy(scalarNames[indx],var->name);
printf("Reset vector %s [%s] (%.2f - %.2f) \n",
       var->name,var->unit,var->minValue,var->maxValue);

return ierr;
};

/*********************************************************************:
```

1997年 09月 18日 10時 24分 /users/pclund/maap/maapsrc/compartment.c ページ 1

```
*****
File name: compartment.c
Author: Per Christer Lund, PNC
Creation date: July 1997
Description: Functions for setting up compartment display at the GUI

Functions: getVector() - gets a vector of compartment parameters
           resetVector() - resets the vector to use another variable
           updateVector() - updates the values of the vector elements
           setLimits() - computes the best upper and lower plot limits
*****
```

```
#include "picasso.h"

*****
char varNames[MAXCOMPDATA][NAMELEN]; /* Backup of variable names */

int getVector(struct CompartData* compData, char* name, float val, float step, int indx)
{
    char vname[NAMELEN], cname[NAMELEN];
    int icom, ivar, iloc, iopt, ierr, i;

    iopt = 2;
    strcpy(compData->name, name);
    compData->iunit = 1;
    compData->minVal = val;
    compData->stepVal = step;
    compData->minValue = 1e10;
    compData->maxValue = -1e10;

    /* Get the variable's data */
    for(i = 1; i <= NCOMPS; i++) {
        sprintf(vname, "%s(%i)", compData->name, i);
        comvar(cname, vname, &compData->indx[i], &icom, &ivar, &iloc, &compData->icf, &iopt, &ierr, NAMELEN, NAMELEN);
        if (ierr != 0) {
            printf("Error code %i (Illegal variable name %s)", ierr, vname);
            compData->failFlag = 1;
            return ierr;
        }
    }

    compData->chFlag = 1;
    updateVector(compData, indx);
    compData->chFlag = 0;

    compData->failFlag = 0;

// setLimits(compData);

    /* Make backup of the names */
    strcpy(varNames[indx], compData->name);
    printf("Defined vector %s [%s] (%.2f -> %.2f) \n",
           varNames[indx], compData->unit, compData->minVal, compData->minVal + (compData->stepVal*LASTCOLOR));
}
```

1997年 09月 18日 10時 24分 /users/pclund/maap/maapsrc/compartmant.c ページ 2

```

return ierr;
}

/********************************************/

int resetVector(struct CompartData* compData,int indx)
{
    char vname[NAMELEN],cname[NAMELEN];
    int icom,ivar,iloc,iopt,ierr,i,iptr,icf;
    iopt = 2;

/* Get the variable's data */
for(i = 1; i <= NCOMPS; i++) {
    sprintf(vname,"%s(%i)",compData->name,i);
    comvar(cname,vname,&iptr,&icom,&ivar,&iloc,&icf,&ierr,NAMELEN,NAMELEN);
    if (ierr != 0){
        printf("Error code %i (Illegal variable name %s)",ierr,vname);
        strcpy(compData->name,varNames[indx]);
        printf("Restoring old variable name %s\n",compData->name);
        sprintf(compData->message,"Undefined MAAP variable %s, restoring %s",vname,compData->name);
        compData->failFlag = 1;
        return ierr;
    }
    compData->indx[i] = iptr;
    compData->icf = icf;
}
updateVector(compData,indx);
compData->chFlag = 0;

strcpy(varNames[indx],compData->name);
printf("Reset vector %s [%s] (%.2f -> %.2f) \n",
       varNames[indx],compData->unit,compData->minVal,compData->minVal + (compData->stepVal*LASTCOLOR));

return ierr;
}

/********************************************/

int updateVector(struct CompartData* compData,int indx)
{
    char unit[UNITLEN],*cnl;
    int i,icol;
    float data;

/* Let the fail flag remain for two seconds */
if (compData->failFlag > 2) compData->failFlag = 0;
if (compData->failFlag) compData->failFlag++;

/* Get the variable's value */
for(i = 1; i <= NCOMPS; i++) {
    comval(&compData->iunit,&compData->indx[i],&data,&compData->icf,unit,UNITLEN);
}

```

1997年 09月 18日 10時 24分 /users/pclund/maap/maapsrc/compartment.c ページ 3

```

compData->data[i] = data;
if (compData->name[0] == 'T') compData->data[i] -= K2C;      // Kelvin to Celcius for temp variables
// Compute the color code
if(compData->stepVal <= 0) compData->stepVal = 1;
icol = (int) ((compData->data[i] - compData->minVal)/compData->stepVal);
compData->color[i] = (icol < 1) ? 1 : icol;
if (compData->color[i] > LASTCOLOR) compData->color[i] = LASTCOLOR;

/* Get max and min values */
if (compData->data[i] > compData->maxValue) compData->maxValue = compData->data[i];
if (compData->data[i] < compData->minValue) compData->minValue = compData->data[i];
}

// Add a unit string (only after change)
if (compData->chFlag) {
    if ( (cnl = strchr(unit, ' ')) *cnl = '\0';
        strcpy(compData->unit,unit);
}

// Check the legend limit settings
if (compData->minValue < compData->minVal)
    sprintf(compData->message,"Suggest setting min. value <= %.2f",compData->minValue);
else if (compData->maxValue > (compData->minVal + (compData->stepVal*LASTCOLOR)) )
    sprintf(compData->message,"Suggest setting step. value >= %.2f",
            (compData->maxValue - compData->minValue)/(LASTCOLOR-2));
else
    sprintf(compData->message,"%s: [%.2f -> %.2f] [%s]",
            compData->name,compData->minValue,compData->maxValue,compData->unit);

return 1;
}

/*
int setLimits(struct CompartData* compData)
{
    compData->minVal = (compData->minValue < 1) ? 0 : 1;
    printf("%.2f %.2f ",compData->minVal,compData->minValue);
    while(compData->minValue > compData->minVal) compData->minVal*=10;
    printf("Min. value %i \n",compData->minVal);

    return 1;
}

/*

```

1997年 09月 05日 15時 20分 /users/pclund/maap/maapsrc/matr.f ページ 1

```

PROGRAM M4ATR
IMPLICIT REAL (A-H, K-Z)
SAVE
EXTERNAL ADATA, COMPT1, COMZR0, EVLABL
EXTERNAL POOLD, METOXD
CNOMG START
    EXTERNAL MBDATA, FONTS
CNOMG END
C*****
C*****
C*
C* MAAP-FUGEN DEGRADED CORE AND CONTAINMENT ANALYSIS PROGRAM *
C* DEVELOPED BY: *
C*
C* FAUSKE AND ASSOCIATES, INC *
C* 16W070 WEST 83RD STREET *
C* BURR RIDGE, ILLINOIS 60521 *
C* (708) 323-8750 *
C*
C* CODE      RELEASED      MANUAL DATE *
C* -----
C* MAAP 4.00    MAY 1994    MAY 1, 1994 *
C* MAAP 4.00a   JUNE 1994   MAY 1, 1994 *
C*
C*****
C*****
C
C     USER NOTES: *
C
C     THE ARCHITECTURE OF THE FISSION PRODUCT MODELS REQUIRES THAT *
C     THE COMPUTER STORE ARRAYS SUCH THAT THE FIRST INDEX VARIES *
C     MOST RAPIDLY; THIS IS STANDARD PRACTICE FOR ALL FORTRAN *
C     COMPILERS KNOWN TO THE CODE AUTHORS; IF YOUR MACHINE STORES *
C     ARRAYS IN A DIFFERENT ORDER, WE SUGGEST YOU CONTACT US *
C     IE, FOR ARRAY "A" DIMENSIONED (3,2) THE FOLLOWING STORAGE *
C     ORDER IS ASSUMED: A(1,1),A(2,1),A(3,1),A(1,2),ETC. *
C
C     MAAP WAS DEVELOPED ON DEC VAX COMPUTERS WHICH FORCE THE CORE *
C     TO BE ZEROED WHEN THE CODE IS STARTED. RUNNING THE CODE ON *
C     COMPUTERS WHICH EITHER DO NOT ALLOW THIS OPTION (IBM) OR *
C     WHICH ALLOW THE USER TO CHOOSE (CDC) OCCASIONALLY PRESENTS *
C     PROBLEMS WITH UNDEFINED VARIABLES. *
C
C     WE ARE GRATEFUL FOR ANY FEEDBACK FROM USERS WHO ENCOUNTER *
C     PROBLEMS WITH UNDEFINED VARIABLES AND CAN USUALLY FIX THE *
C     PROBLEM FAIRLY RAPIDLY ONCE IT HAS BEEN IDENTIFIED. IN THE *
C     VAST MAJORITY OF CASES, THE PROBLEM DOESN'T AFFECT THE RESULTS*
C     (EG THE PROBLEM AROSE BECAUSE A VARIABLE BEING SOLVED FOR *
C     ITERATIVELY DID NOT HAVE AN INITIAL GUESS ON THE FIRST TIME *
C     STEP). IF YOU ARE USING THE CODE ON A COMPUTER *
C     WHERE YOU HAVE THE OPTION TO ZERO THE CORE, YOU MAY CHOOSE TO *
C     DO SO TO MINIMIZE HEADACHES. IF YOU CHOOSE NOT TO, OR IF YOU *
C     DON'T HAVE THE OPTION, PLEASE LET US KNOW IF ANY PROBLEMS *
C     ARISE. IT IS VERY HELPFUL IF YOU CAN IDENTIFY: *
C         A. WHICH VARIABLE WAS UNDEFINED *
C         B. IN WHAT SUBROUTINE WAS IT BEING USED WHEN THE ERROR *

```

1997年 09月 05日 15時 20分 /users/pclund/maap/maapsrc/matr.f ページ 2

C OCCURED
C C. APPROXIMATELY WHEN IN THE RUN THE ERROR OCCURRED (EG, *
C AT T=0 IN A RESTART RUN, OR AFTER AN OPERATOR ACTION, *
C ETC.). *

C ON SOME COMPUTERS, GETTING THIS INFORMATION MAY NECESSITATE *
C COMPILE THE CODE WITH A "DEBUG" OPTION SELECTED. *

C WE ALSO OCCASIONALLY SEE PROBLEMS WITH THE FORTRAN COMPILER *
C OPTIMIZERS ON VARIOUS COMPUTERS. THESE MANIFEST THEMSELVES *
C WHEN THE CODE DIES (SAY, ON THE SAMPLE PROBLEM) FOR NO *
C APPARENT REASON. IF YOU ENCOUNTER SUCH PROBLEMS, YOU MAY WISH *
C TO TRY SELECTING NO-OPTIMIZATION OR A LOW LEVEL OF *
C OPTIMIZATION WHEN YOU COMPILE MAAP. AGAIN, WE APPRECIATE *
C FEEDBACK FROM USERS ENCOUNTERING THESE KINDS OF PROBLEMS. *

C*****

COMMON/CONSTS / MOLWW, MOLWH2, MOLWO2, MOLWCO, MOLWC2, MOLWN2, MOLWCA,
1 CVH2, CVO2, CVCO, CVC2, CVN2, CVHE, CPH2, CPO2, CPCO, CPC2, CPN2, CPHE,
2 GAMMA, RGAS, GRAV, PI, SIGMA, TCRITW, PCRITW, LHICE, CPICE,
3 MOLWU2, MOLWZR, MOLWZO, MOLWCS, TU2MP, TZRMP, TZOMP, TCSMP,
4 QMH202, QMCAC2, QMZRW, QMCAW, QMZR2, QMC02, MWMIN, QMCWS, QMCSCO,
5 QMCRW, MOLWSO, MOLWCI, TCAMP, IDIAG, MOLWU, MOLWO, MOLWHE,
6 TSSMP, TSOMP, TBCMP, TAGSOL, TAGLIQ, TBSMP, MOLWSS, MOLWBC, TWMP,
7 CBPC, MOLWB, MOLWC, FDU2, MWMIC, MOLWHW, TINCMPP
COMMON/CONTRL/IPSPPR, ICPR, IRUNG, IRSTW, IRSTR, IPLMAP, IPOUT,
1 ICOUT, IPLT1, ISUM, IPTSAV, IPTSPK, IPTSMX, IRECIR, IDISCH,
2 IAUXW, IAUXR, INODRB, ISTW, ISTR, ISENS, IPAINT,
3 INPGRP, JNTGRT, ITDLIM, ISORT, JSAVE, ISAVE, IRMAX, JCHMAX,
4 INNODE, ICWRI, INSMGR, INLOOP,
6 JBB, ICWALL(4,12), ILCM, ILEST, IU02, IZR, IZO, ICS, ISO, INFPMX, ITMLPS,
9 IEMBAL, ICRLB, IENVIR, IMXNCM, IMXJCM, INHSAX, INEQAX, ITLNX,
. IDBGRF, IDBGWF, IDBGTD, IRFRSH, IPC,
1 ISIDRL, ICANDL, IDISSV, IEUTEC, IDISP, ITTOP, IHTHA, IHTFC, IOXIDE,
2 IPSET1(40), IPSET2(40), IUPLIS(25), IRS, IRUNGK, IMVUNT,
3 IAOPX, IBALON, IFLOOD, IUSETD, ILPBAL, IDISSB, IXPSL, IBCCR,
4 IMPAIR, ILOG, ISSCL, IPOWER, IOPTVF, IRSBAD, IDBLVL, IQDPB, IGCHF, IDPQNC,
5 IHTGPL, IUXOLD, IHGRAD, IXPSE, ISTEP3, IFUGEN, JRMAX(5,2), ILOOPS,
6 IPTREL, ITDSUB, ICAEVT, IJETHT, IJETAT(5,2)
COMMON/TIMING/ TTPRNT, TILAST, TDMAX, TDMIN, MFCHMX, MDFPMN, FTGCHX,
1 MFCHFP, TTSTAT, TIOP, TIM, TD, NCRTEQ, TDOLD, TDAUX, TIMRAT, TDNEW,
2 FPPSHL, TIUX, TDFQMN, TDFQMX, TDCT, TDSK,
3 TDHCL(2), TDHUTR(2), TDHUU2(2), TDHUPT(2), TDHUKT(2), TDHU(2),
4 FMCHX, FUCHX, FTCHX, FPCHX, FVCHX, FWCHX, TDAUXM, TIPRNT, TDINV,
3 TDMVIE, TISCRM, TIVFCT, TTIMPR, TIIMPR, TIDENT, TDXUCT(2),
4 TDXECT(2), TDXLCT(6,2), TDXUSK, TDXLSK(6), TIVFSK,
5 TIFWTP(2), TIMSIV(2), TICPMP(2), TIAPCI(2), TICSP,
6 TDAVG(25)
DOUBLE PRECISION TIMD, TIMFPD
COMMON/TIME2/TIMD, TIMFPD
CHARACTER*80 ITITL
COMMON/TITLE/ITITL(3)
COMMON/UNITS/ IUNITS

1997年 09月 05日 15時 20分 /users/pclund/maap/maapsrc/matr.f ページ 3

```

COMMON/CODERV/ VRSION
C
CHARACTER*80 LINE
DATA INFILE/0/,IFIRST/0/

C****. PICASSO interface implementation
C**** DECLARATION OF SUBROUTINES AND RETURN VALUES
C**** Per Christer Lund, PNC/Fugen, July 1997

c**** Data transferred from the main program file ****
c common/pic3dat/ IMG,IFILOP,IPRM,IERR

EXTERNAL INIT_PIC
EXTERNAL UPDATE_PIC
INTEGER*4 INIT_PIC, UPDATE_PIC
INTEGER*4 PSTATUS

C**** END Picasso

CUNIX-HP UNCOMMENT START
C for hp, ignore underflows, ie., do not trap underflows
c      ON REAL UNDERFLOW IGNORE
c      ON REAL*8 UNDERFLOW IGNORE
c      ON DOUBLE PRECISION UNDERFLOW IGNORE
CUNIX-HP UNCOMMENT END
C
VRSION = 'MAAP-FUGEN 0.01 FEB 14, 1997'
C
C INITIALIZE THE TIMER
CALL TIMIT(0,0)

C
C WRITE BANNER TO SCREEN
C
WRITE(*,101) VRSION
101 FORMAT(///1X,' ',A,
          /1X,'MAAP PICASSO'///)
C
C SET UP DEFAULT MODEL PARAMETER VALUES
C
CALL MAAPC(2,0)

C
C DEFINE FILE OPENING MODE
C
C SET IFILOP = 0, FOR BATCH MODE JCL ASSIGNMENT (DEFAULT)
C      = 1, USE FORTRAN OPEN & FILES IN UPPER CASE
C      = 2, USE FORTRAN OPEN & FILES IN LOWER CASE
C
IFILOP=2
C
ILOG=-6
IF (IFILOP.EQ.0) THEN
C JCL ASSIGNMENTS

```

1997年 09月 05日 15時 20分 /users/pclund/maap/maapsrc/matr.f ページ 4

```

ISTR= 5
ISTR= 6
IDIAG= 6
ELSE
C FORTRAN OPEN STATEMENT;
C UNITS 5 & 6 RESERVED FOR KEYBOARD & SCREEN EXCLUSIVELY.
ISTR= 7
ISTR= 8
IDIAG= 8
CALL GETFIL (ISTR,ISTW,IDIAG,ILOG,IERR,IPRM,
CREV 4.0.2 MAM 4/26/95 START      PASS IPARG PARAMETER
C      &      IFILOP, IPLT1, IUXOLD)
&      IFILOP, IPLT1, IUXOLD, IPARG)
CREV 4.0.2 MAM 4/26/95 END
ENDIF
C
C CODE BANNER FOR DIAGNOSTIC LOG FILE
C
      WRITE (IDIAG,701)
      WRITE (IDIAG,703)
      WRITE (IDIAG,705) VRSION
      WRITE (IDIAG,701)
C
C READ & PROCESS THE INPUT FILE IN ENTIRITY
C
      CALL INPUT4 (ISTR,ISTW,IDIAG,ILOG,IERR,IFILOP,0,IPRM,
      ITMLPS,ISENS,ITITL,TIM,TILAST,TTPRNT,TTSTAT,IRSTR,IUNITS,
CREV 4.0.2 MAM 4/26/95 START      PASS IPARG PARAMETER
C      IMG, IPLT1, IUXOLD)
      IMG, IPLT1, IUXOLD, IPARG)
CREV 4.0.2 MAM 4/26/95 END
C
C PRINT MAAPC SUMMARY
C
      CALL MAAPC(1,0)
C
C STAND-ALONE BENCHMARK MODE INITIALIZATION
C
      CALL BENCH(0,0)
C
C INITIALIZE THE CODE
C
      CALL FPGRP(1)
      CALL DIFFUN(0)
      CALL FPGRP(2)
      CALL DIFFP(0)
C
      IF (IRSTR.EQ.0) THEN
C INITIAL RUN
      TIM=0.E0
      TD=TDMIN
      TDNEW=TDMIN
      CALL DIFFUN(2)
      CALL DIFFUN(4)
      CALL DIFFUN(3)
      ELSE

```

1997年 09月 05日 15時 20分 /users/pclund/maap/maapsrc/matr.f ページ 5

```

C MAAP RESTART
    CALL RESREK(1)
    ENDIF
C
C CODE BANNER FOR TABULAR OUTPUT
C
    WRITE (IPOUT,702)
    WRITE (IPOUT,703)
    WRITE (IPOUT,705) VRSION
    WRITE (IPOUT,702)
C
C INITIALIZE DIFFUN (THERMAL HYDRAULIC) & DIFFP (FISSION PRODUCT)
C
    CALL DIFFP(2)
    CALL DIFFUN(2)
    IF (IRSTR.EQ.0) CALL DIFFUN(4)
C
C INITIALIZE DOUBLE PRECISION TIME VARIABLES
C
    TIMD=TIM
    TIMFPD=TIM
C
C**** Initiate PICASSO-3 connection, and let the PICASSO timer
C     control the looping. Jump over the main loop.
C
    PTEND=-TILAST
    PSTATUS = INIT_PIC(TILAST)
    IF-(PSTATUS.EQ.0) THEN
        GOTO 125
    ENDIF
C**** End
C ====== START OF MAAP MAIN LOOP ======
C
111  CONTINUE
C
C SET "EVENT CODES" WHICH CONTROL THE MODELS
C IF THIS IS A RESTART RUN & FIRST TIME STEP, SKIP EVENTS
C IE., USE EVENT FLAGS FROM THE RESTART FILE
C
    IF (IRSTR.EQ.0 .OR. IFIRST.EQ.1) THEN
        CALL EVTCOD (ICAEVN,IMG,IFILOP,IPRM)
    ELSE
        CALL EVTCOD (0,IMG,IFILOP,IPRM)
    ENDIF
    IFIRST=1
C
C SAVE THE DEBUG RESTART FILE TO UNIT IDBGWF EVERY IDBGTD Timesteps
C
    IF (IDBGTD.GT.0) CALL RESREK(4)
C
C TABULAR OUTPUT
C
    CALL OUTPT1(0,INFILE)
C

```

1997年 09月 05日 15時 20分 /users/pclund/maap/maapsrc/matr.f ページ 6

```

C SUMMARY OF KEY EVENTS
C
C       IF (ISUM.NE.0) CALL SUMMARY
C
C PLOT DATA
C
C       CALL PLTMAP (ISTR,ISTW,IDIAG,0,0,IERR,
C                      IFILOP,1,IPLT1,LINE)
C
C CHECK LIMITING Timestep VARIABLES
C
C       CALL TIMCHK
C
C IF END OF RUN, STOP
C
C       IF (TIM.GE.TILAST) GO TO 125
C
C UPDATE THERMAL HYDRAULIC VARIABLES
C
C       CALL INTGRT
C
C UPDATE FISSION PRODUCT VARIABLES
C
C       CALL INTGFP (TDMAX,TDMIN,MFCHFP,MDFPMN)
C
C LOOP BACK TO 111
C
C       GO TO 111
C
C ===== END OF MAAP MAIN LOOP =====
C
125 CONTINUE

CNOMG START
CREV 4.0.2 CEH 7/31/95 START
C       IF (IMG.GT.0 .OR. IMVUNT.GT.0) CALL MGPI(0,IMG)
C       IF (IMG.GT.0) THEN
C           CALL MGPI(0,IMG)
C       ELSE IF (IMVUNT.GT.0) THEN
C           CALL MGPI(-1,IMG)
C       END IF
CREV 4.0.2 CEH 7/31/95 END
CNOMG END

C
701  FORMAT(/1X,78('=')/)
702  FORMAT(2(1X,130(' '))/)
703  FORMAT(1X,/,1X,'M A A P 4 - F U G E N S E V E R E',
1' A C C I D E N T A N A L Y S I S C O D E')
705  FORMAT(24X,A)
C
C RETURN THE ELAPSED (AND/OR CPU) TIME
    CALL TIMIT(1,IDIAG)
    STOP
    END

```

1997年 08月 26日 11時 41分 /users/pclund/maap/maapsrc/picloop.f ページ 1

```

SUBROUTINE PICLOOP(istat)
IMPLICIT REAL (A-H, K-Z)
SAVE

c COMMON/CONSTS/ MOLWW, MOLWH2, MOLW02, MOLWCO, MOLWC2, MOLWN2, MOLWCA,
c 1 CVH2, CVO2, CVCO, CVC2, CVN2, CVHE, CPH2, CPO2, CPCO, CPC2, CPN2, CPHE,
c 2 GAMMA, RGAS, GRAV, PI, SIGMA, TCRITW, PCRITW, LHICE, CPICE,
c 3 MOLWU2, MOLWZR, MOLWZO, MOLWCS, TU2MP, TZRMP, TZOMP, TCSMP,
c 4 QMH202, QMCAC2, QMZRW, QMCAW, QMZRC2, QMC002, MWMIN, QMCsw, QMCSCO,
c 5 QMCRW, MOLWSO, MOLWCI, TCAMP, IDIAG, MOLWU, MOLWO, MOLWHE,
c 6 TSSMP, TSOMP, TBCMP, TAGSOL, TAGLIQ, TB SMP, MOLWSS, MOLWBC, TWMP,
c 7 CPBC, MOLWB, MOLWC, FDU2, MMIC, MOLWHW, TINCM P
COMMON/CONTRL/IPSPPR, ICPR, IRUNG, IRSTW, ISTR, IPLMAP, IPUT,
1 ICOUT, IPLT1, ISUM, IPTSAV, IPTSPK, IPTSMX, IRECIR, IDISCH,
2 IAUXW, IAUXR, INODRB, ISTW, ISTR, ISENS, IPAIN T,
3 INPGRP, JNTGRT, ITDLIM, ISORT, JSAVE, ISAVE, IRMAX, JCHMAX,
4 INNODE, ICWRI, INSMGR, INLOOP,
6 JBB, ICWALL(4, 12), ILCM, ILEST, IU02, IZR, IZO, ICS, ISO, INFPMX, ITMLPS,
9 IEMBAL, ICRBAL, IENVIR, IMXNCM, IMXJCM, INHSAX, INEQAX, ITLNX,
. IDBGRE, IDBGWF, IDBGTD, IRFRSH, IPC,
1 ISIDRL, ICANDL, IDISSV, IEUTEC, IDISP, ITTOP, IHTHA, IHTEC, IOXIDE,
2 IPSET1(40), IPSET2(40), IUPLIS(25), IRS, IRUNGK, IMVUNT,
3 IAOXP, IBALON, IFLOOD, IUSETD, ILPBAL, IDISSB, IXPSL, IBCCR,
4 IMPAIR, ILOG, ISSCL, IPOWER, IOPTVF, IRSBAD, IDBLVL, IQDPB, IGCHF, IDPQNC,
5 IHGPL, IUXOLD, IHGRAD, IXPSE, ISTEP3, IFUGEN, JRMAX(5, 2), ILOOPS,
6 IPTREL, ITDSUB, ICAEV T, IJETHT, IJETAT(5, 2)

COMMON/TIMING/ TTPRNT, TILAST, TDMAX, TDMIN, MFCHMX, MDFPMN, FTGCHX,
1 MFCHFP, TTSTAT, TIOP, TIM, TD, NCRTEQ, TDOLD, TDAUX, TIMRAT, TDNEW,
2 FPPSHL, TIUX, TDFQMN, TDFQMX, TDCT, TDSK,
3 TDHUCL(2), TDHUTR(2), TDHUU2(2), TDHUPT(2), TDHUKT(2), TDHU(2),
4 FMCHX, FUCHX, FTCHX, FPCHX, FWCHX, TDAUXM, TIPRNT, TDINV,
3 TDMVIE, TISCRM, TIVFCT, TTIMPR, TIIMPR, TIDENT, TDXUCT(2),
4 TDXECT(2), TDXLCT(6, 2), TDXUSK, TDXLSK(6), TIVFSK,
5 TIFWTP(2), TIMSIV(2), TICPMP(2), TIAPCI(2), TICSP,
6 TDAVG(25)

DOUBLE PRECISION TIMD, TIMFPD
COMMON/TIME2/TIMD, TIMFPD
c CHARACTER*80 ITITL
c COMMON/TITLE/ITITL(3)
c COMMON/UNITS/ IUNITS
c
c CHARACTER*30 VRSION
c COMMON/CODERV/ VRSION
c

CHARACTER*80 LINE
DATA INFILE/0/, IFIRST/0/

***** Data transferred from the main program file *****
common/pic3dat/ IMG, IFILOP, IPRM, IERR

C ===== START OF MAAP MAIN LOOP =====
C
C SET "EVENT CODES" WHICH CONTROL THE MODELS
C IF THIS IS A RESTART RUN & FIRST TIME STEP, SKIP EVENTS

```

1997年 08月 26日 11時 41分 /users/pclund/maap/maapsrc/picloop.f ページ 2

```
C IE., USE EVENT FLAGS FROM THE RESTART FILE
C
IF (IRSTR.EQ.0 .OR. IFIRST.EQ.1) THEN
  CALL EVTCOD (ICAEVT,IMG,IFILOP,IPRM)
ELSE
  CALL EVTCOD (0,IMG,IFILOP,IPRM)
ENDIF
IFIrst=1
C
C SAVE THE DEBUG RESTART FILE TO UNIT IDBGWF EVERY IDBGTD TIMESTEPS
C
IF (IDBGTD.GT.0) CALL RESREK(4)
C
C TABULAR OUTPUT
C
CALL OUTPT1(0,INFILE)
C
C SUMMARY OF KEY EVENTS
C
IF (ISUM.NE.0) CALL SUMMRY
C
C PLOT DATA
C
CALL PLTMAP (ISTR,ISTW,IDIAG,0,0,IERR,
              IFILOP,1,IPLT1,LINE)
C
C CHECK LIMITING TIMESTEP VARIABLES
C
CALL TIMCHK
C
C IF END OF RUN, STOP
C
IF (TIM.GE.TILAST) GO TO 125
C
C UPDATE THERMAL HYDRAULIC VARIABLES
C
CALL INTGRT
C
C UPDATE FISSION PRODUCT VARIABLES
C
CALL INTGFP (TDMAX,TDMIN,MFCHFP,MDFPMN)
C
=====      END OF MAAP MAIN LOOP      =====
```

125 RETURN
END

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 1

```

CREV-MG START
C      SUBROUTINE APIC1(TTSCRN, ISCRN, TDZPIC, IMG, IDISON)
C      SUBROUTINE PIC_API()
C      IMPLICIT REAL    (A-H, K-Z)
C      SAVE
C*****
C APIC1
C      Subroutine which provides the interface between
C      ATR MAAP 4.0 and MAAP-GRAAPH display routines
C VARIABLES
C
C passed:
C TTSCRN - (input) screen update interval
C ISCRN  - (input) flag to indicate if graphic display is
C           activated (0-yes, 1-no)
C TDZPIC - time interval for saving movie data
C IMG     - flag indicating maap4-graaph display status:
C           IMG = 0 - display off
C           IMG = 1 - display on
C           IMG = 2 - movie playback mode
C IDISON - flag indicating if display should be updated immediately
C           or wait for screen update time:
C           IDISON = 1 - wait for screen update time
C           IDISON = 0 - update immediately
C common:
C INTRPT - (input,output) flag indicating if the users wants to
C           do an operator intervention (0-yes, 1-no)
C TTPRNT - (input) MAAP print interval for restart files
C IMVER  - (input,output) flag indicating which MAAP code is running
C           (0-BWR, 1-PWR)
C DXNOD  - (output) delta x value for placing core node temperatures
C           on screen
C DYNOD  - (output) delta y value for placing core node temperatures
C           on screen
C CHNOD  - (output) height of text to be placed on screen
C IPAINT - (input,output) flag indicating refreshing of the MG
C           screen (0-no,1-yes)
C KEY    - (input) contain the character corresponding to the key
C           hit on the keyboard
C ITMLPS - (input) flag indicating if time lapse restart is active
C ICORE   - (input) flag indicating if the core needs to be updated
C IMOVIE  - (input,output) flag indicating which mide is running for
C           movie playback (0-Interactive w/graphics, -2-Interactive
C           w/o graphics, -1-Batch)
C I_EVENT - (input,output) array containing the event flags for the
C           corresponding shapes
C I_EVENT_OLD - (input,output) array containing the old event flags
C           for the corresponding shapes
C TIM     - (input) current accident time
C TIOP    - (input,output) time for an operator intervention
C IFLAGP  - (input,output) flag indicating intervention set for time
C           condition only for an operator intervention
C local:
C FMASS  - (output) array containing mass fraction of core nodes
C ZLEV   - (output) array containing water and corium levels

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 2

```

C TCRM - (output) array containing temperatures of corium
C TPIC - (output) array containing core node temperatures
C IMAP - mapping of the core nodes
C MFH2C - array containing mass fraction of H2
C INDX - array containing index into MFH2C by the corresponding
        shape numbers
C TIZPIC - calculates TIM + TDZPIC to find time to write movie data
C ICAL - flag indicating if node masses have been set to 0 (0=yes,1-no)
C JLPCI - index indicating which LPCI loop is active
C JSRV - index indicating which SRV is active
C TTSCRN - calculates TIM + TTSCRN to see when to update screen
C TNOD - temporary array for core node temperatures
C MNOD - temporary array for core node masses
C MO - arrary containing initial core node masses
*****
CREV-MAM START ADD PARAMETER CONSTANTS
C*CORPAR
    PARAMETER (IRX=25)
C*CORPAR
C*PARMS
    INTEGER NNODX, NCMX, NHSX, NEQX, NICMX
    PARAMETER (NNODX=40, NCMX=10, NHSX=200, NEQX=200, NICMX=2)
    INTEGER NWX, NACTX, NAMATX
    PARAMETER (NWX=120, NACTX=120, NAMATX=NNODX+NACTX)
    INTEGER NCMWX, NWNX, NGX, NNODIC
    PARAMETER (NCMWX=10, NWNX=10, NGX=6, NNODIC=5)
C*PARMS
CREV-MAM END
C*RCSPAR
    PARAMETER (ICH5=5, ILS=2, ITAB=10, IHSF=16, INODS=19, IWGX=14)
    PARAMETER (ITAB1=ITAB*2, JUNS=INODS+9)
    PARAMETER (JMPO=5, JMPN=5, JMSD=5, JMDC=5, JMWD=5, JMPI=5)
    PARAMETER (IMPO=2, IMPN=2, IMSD=5, IMDC=5, IMWD=5, IMPI=2)
C*RCSPAR
    LOGICAL evtst, evtst2
C*MAXCONTS
    INTEGER MAXSHAPES, MAXPTS, MAXNODES, CMAXAX, CMAXRAD, CMAXPTS
    PARAMETER (MAXSHAPES = 350)
    PARAMETER (MAXPTS = 200)
    PARAMETER (MAXNODES = 200)
    PARAMETER (CMAXAX = 20)
    PARAMETER (CMAXRAD = 10)
    PARAMETER (CMAXPTS = 4)
    PARAMETER (ICST = 8)
C*MAXCONTS
    COMMON/CONSTS/ MOLWW, MOLWH2, MOLW02, MOLWCO, MOLWC2, MOLWN2, MOLWCA,
1 CVH2, CVO2, CVCO, CVC2, CVN2, CVHE, CPH2, CPO2, CPCO, CPC2, CPN2, CPHE,
2 GAMMA, RGAS, GRAV, PI, SIGMA, TCRITW, PCRITW, LHICE, CPICE,
3 MOLWU2, MOLWZR, MOLWZO, MOLWCS, TU2MP, TZRMP, TZOMP, TCSMP,
4 QMH2O2, QMCAC2, QMZRW, QMCAW, QM2RC2, QMCOO2, MWMIN, QMCSSW, QMCSCO,
5 QMCRW, MOLWSO, MOLWCI, TCAMP, IDIAG, MOLWU, MOLWO, MOLWHE,
6 TSSMP, TSOMP, TBCMP, TAGSOL, TAGLIO, TBSMP, MOLWSS, MOLWBC, TWMP,
7 CPBC, MOLWB, MOLWC, FDU2, MMWIC, MOLWHW, TINCMP
    COMMON/TPOL11/ IVOL(NWX, 2), IJRB(NNODX, NNODX), IJMAX(NNODX),
1 IHRB(NWX), IFRB(NWX), IMAXJ, IJUMAX, IJACT(NWX),
2 IRBIG(NNODX), IRBSP(NNODX), IRBC2(NNODX), IBRRB(NNODX),

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 3

```

3 INHSRB(NNODX), INCM(NNODX), IJNOMX(NNODX), IDONRB(NWX), INJDF(NWX)
COMMON/PSCPTR/ JNRVCT, JNSOCT, JNBOCT, JNDR, JNPS(INODS,2), JNRVPS,
1 ICMAX, JNSPL, JNSK, JNRVSK, JNSFP,
2 JNCM(NCMX), JCMW(20), JDGF(4),
3 JSCTS(NWX), JNERT(NNODX), JFRB(NWX),
4 JJVSK(10), JJRVCT(10), JJRVSK(10), JNFCS, JNFCD,
5 JNCSP(4), JNCSPA, JNCHS, JNCHD, JNDRSK,
6 JJBB(10,2), JJRV(10,2), JNDUM(4)
COMMON/CONTRL/IPSPL, ICPR, IRUNG, IRSTW, IRSTR, IPLMAP, IPUT,
1 ICOUT, IPLT1, ISUM, IPTSAV, IPTSPK, IPTSMX, IRECIR, IDISCH,
2 IAUXW, IAUXR, INODRB, ISTW, ISTR, ISENS, IPAINT,
3 INPGRP, JNTGRT, ITDLIM, ISORT, JSAVE, ISAVE, IRMAX, JCHMAX,
4 INNODE, ICWRI, INSMGR, INLOOP,
6 JBB, ICMALL(4,12), ILCM, ILEST, IUO2, IZR, IZO, ICS, ISO, INFPMX, ITMLPS,
9 IEMBAL, ICRBAL, IENVIR, IMXNCM, IMXJCM, INHSAX, INEQAX, ITLNX,
. IDBGRF, IDBGWF, IDBGTD, IFRSH, IPC,
1 ISIDRL, ICANDL, IDISSV, IEUTE, IDISP, ITTOP, IHTHA, IHTFC, IOXIDE,
2 IPSET1(40), IPSET2(40), IUPLIS(25), IRS, IRUNGK, IMVUNT,
3 IAOXP, IBALON, IFLOOD, IUSETD, ILPBAL, IDISSB, IXPSL, IBCCR,
4 IMPAIR, ILOG, ISSCL, IPOWLR, IOPTVF, IRSBAD, IDBLVL, IQDPB, IGCHF, IDPQNC,
5 IHGPL, IUXOLD, IHGRAD, IXPSE, ISTEP3, IFUGEN, JRMAX(5,2), ILOOPS,
6 IPTREL, ITDSUB, ICAEVT, IJETH, IJETAT(5,2)
COMMON/EVENTC/ IAEVMX, IEVMAX, IEVNT(1600), IOULD(1600),
. JCOUNT, JEVNT(280), IBCEVT, IECEVT, INCEVT,
. ILIST(1600),
. ISTUCK(10,2), IFAILC(10,2), ILOCK(10,2), IMANON(10,2), IOPEN(10,2)
COMMON/TIMING/ TTPRNT, TILAST, TDMAX, TDMIN, MFCHMX, MDFPMN, FTGCHX,
1 MFCHFP, TTSTAT, TIOP, TIM, TD, NCRTEQ, TDOLD, TDAUX, TIMRAT, TDNEW,
2 FPPSHL, TIUX, TDFQMN, TDFQMX, TDCT, TDSK,
3 TDHUCI(2), TDHUTR(2), TDHUU2(2), TDHUPT(2), TDHUKT(2), TDHU(2),
4 FMCHX, FUCHX, FTCHX, FPCHX, FVCHX, FWCHX, TDAUXM, TIPRNT, TDINV,
3 TDMVIE, TISCRM, TIVFCT, TTIMPR, TIIMPR, TIDENT, TDXUCT(2),
4 TDXECT(2), TDXLCT(6,2), TDXUSK, TDXLSK(6), TIVFSK,
5 TIFWTP(2), TIMSIV(2), TICPMP(2), TIAPCI(2), TICSP,
6 TDAVG(25)
COMMON/HEAT2/
1 QWCH(ICHS,2), WWCH(ICHS,2), WSTCH(ICHS,2), WH2CH(ICHS,2),
2 WN2CH(ICHS,2), WO2CH(ICHS,2), WC2CH(ICHS,2), WCOCH(ICHS,2),
3 QGCH(ICHS,2), WHECH(ICHS,2), MCR(2), TCRHOT(2), TCRAVG(2),
4 MLTCH(ICHS,2), FZRRRC(2), FZRRP(2), FZRRK(2), FZRRB(2),
5 TNOD(IRX, ICHS, 2),
6 TU2N(3, IRX, ICHS, 2), TCLN(3, IRX, ICHS, 2), TPTN(IRX, ICHS, 2),
7 TTRN(IRX, ICHS, 2), TTKTN(IRX, ICHS, 2), TCBN(IRX, ICHS, 2),
8 MNOD(IRX, ICHS, 2), MCLN(3, IRX, ICHS, 2), MTRN(IRX, ICHS, 2),
9 MPTN(IRX, ICHS, 2), MKTN(IRX, ICHS, 2), MCRN(IRX, ICHS, 2),
. MZRUZ(3, IRX, ICHS, 2), FECLPT(IRX, ICHS, 2), MZRUN(IRX, ICHS, 2),
1 MLNOD(IRX, ICHS, 2), VLCMCH(ICHS, 2), MSSOP, MSSOK,
2 WSTMCH(ICHS, 2), WWOXCH(ICHS, 2), QWCR(ICHS, 2),
3 WSTCNT(ICHS, 2), WU2ND(IRX, ICHS, 2), WZRND(IRX, ICHS, 2),
4 WZOND(IRX, ICHS, 2), WSSND(IRX, ICHS, 2), WSOND(IRX, ICHS, 2),
5 WUND(IRX, ICHS, 2), WBCND(IRX, ICHS, 2), QCMND(IRX, ICHS, 2),
6 WFQCMD(IRX, ICHS, 2), WSTLVL(ICHS, 2), VFDNCR(ICHS, 2),
7 XDCLDN(IRX), XDPRTN(IRX), XTPTN(IRX), XTKTN(IRX), VFDGO(ICHS, 2),
8 XDHYCT(IRX), XHEAT2(5)
COMMON/HEAT4/
1 MU2N0(IRX), MZRC0(IRX), MZOC0(IRX), MSSC0(IRX),

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 4

```

2 MSOC0(IRX), MZRR0(IRX), MZRP0(IRX), MSSP0(IRX),
3 MZRK0(IRX), MSSK0(IRX), MZRB0(IRX), MSSB0(IRX), MBCB0(IRX),
4 MNOD0(IRX), AGCH0, TGCH(ICHS, 2), ZD(IRX), AHT0(IRX),
4 VLCMN(IRX, ICHS, 2), XRPN(3, IRX, ICHS, 2), XRU2(3, IRX, ICHS, 2),
5 AGCH(IRX, ICHS, 2), XDHY(IRX, ICHS, 2), EPS(3, IRX, ICHS, 2),
6 ACHJ(3, IRX, ICHS, 2), FAA(3, ICHS, 2), FPRR(3, ICHS, 2),
7 WSTNOD(IRX, ICHS, 2), WH2NOD(IRX, ICHS, 2),
8 WNCNOD(IRX, ICHS, 2), QGNOD(IRX, ICHS, 2), TGNOD(IRX, ICHS, 2),
9 VGNOD(IRX, ICHS, 2), MOLNCN(IRX, ICHS, 2), AHTMX(3, IRX, ICHS, 2),
. AHTOX(3, IRX, ICHS, 2), VCLN0(IRX), VU2N0(IRX), VTRN0(IRX),
1 AHT(3, IRX, ICHS, 2), AHTPT(IRX, ICHS, 2), AHTKT(IRX, ICHS, 2),
2 AHTTR(IRX, ICHS, 2), AHTCB(IRX, ICHS, 2), XPWET(3, IRX, ICHS, 2),
3 AHTTR0(IRX, ICHS, 2), MATRN0(IRX), MINC60(IRX),
4 MATRN(3, IRX, ICHS, 2), MINC6C(3, IRX, ICHS, 2), XRCRST(IRX, ICHS, 2),
5 XCRUST(IRX, ICHS, 2), TCRUST(IRX, ICHS, 2), FAFLOW(3, ICHS, 2),
6 MINCTR(IRX), MINC6R(IRX, ICHS, 2)

```

C\$PS2

C\$P22

COMMON /ES1/

```

1 NACUM, MACUM0, VACUM, PACUM0, TDAPCI, TACUM, XDACUM, XLACM1,
2 XLACM2, XLACM3, KA1, KA2, KA3, MACUML, MWCST0, TWCST, XDHPEC,
3 XLHP1, XLHP2, XLHP3, KHP1, KHP2, KHP3, NPHPIP, DPHPI(10), WVHPI(10),
4 XDLPEC, XLLP1, XLLP2, XLLP3, KLP1, KLP2, KLP3, NPLPIP, DPLPI(10),
5 WVLPI(10), NSPHXP, DPSPHX(10), WVSPHX(10), TWHXEC, NTUECC,
6 KRC1, KRC2, KRC3, WWRCI(2), NTURCI, FHRHR, NTURHR, WWSRHR,
7 TSRHRI, WWRHR, ZHCSP(4), ZHCSPA, FWCS(4), XDCSP,
9 NFN, WVFN0, NTFC, ATFC, AFINFC, FFINFC, RGFLHX, XDFNFC, XTTFC,
. KTF, AFLMN, XIDTFC, NREGFC, TCWHD, WCWFC,
1 NCHILL, WVCH0, NTCH, ATCH, AFINCH, FFINCH, RGFLCH, XDFNCH, XTTCH,
2 KTCH, AFMNCH, XIDTCH, NREGCH, TCWCH, WCWCH, XDSP, PSP0, TDCSP,
3 PRBHIG, ZWLSP, ZWSDNM, ZSDL2, ZSDL3, ZSDL4, ZDSDH2, PPSL1, PPSL3,
4 NPSPIP, DPSPI(10), WVSPI(10), WVECC, TWSPCL, WSPLMX, PRCICL
COMMON/ES2/
1 WACUM(2), WWHPI(2), WWLPI(2), WESFSP, WHESSP, QESFSP,
2 QHXRCI(2), QRHRPS(2), WHEFS(2), WHESF(2), WWBRKA(2),
3 DPDMPS(2), DPDMAC, RBRK(2), TWSDSP(2), HWESF, WSTTUR(2),
4 QSTTUR(2), PACUM, TIFAN1, TIFAN2, QHXSPC, QHXCSP,
5 WWSPA, WWCSPA, WSCSPA, QWCSPA, QGCSPA, TWSPIN, HWSPIN, WWSPP

```

C\$SG2

```

COMMON/CM2/ WFPCMR(14, NCMX), TCMB(NCMX), TCMMPB(NCMX), AUWB(NCMX),
1 TSUWB(NCMX), QCMRAD(NCMX), TCMIB(NCMX), MCMIB(NCMX), VOLCMB(NCMX),
2 HCMB(NCMX), VCMB(NCMX), ZCMB(NCMX), CPCMB(NCMX), KCMB(NCMX),
3 DCMB(NCMX), STCMB(NCMX),
4 ZNCMIB, XDCMIB, WCMIB(50, NCMX), QCMIB(NCMX), FFQCIB(NCMX),
4 TCNDI(NCMX), TCNSI(NCMX), TCNUI(NCMX),
5 FSOLC(NCMX), VISCOMB(NCMX), LHCMB(NCMX), TDENT, FFPENT, FDCH, RELF,
6 QGcmb(NCMX), QCMBB(NCMX), WSTCMA, WH2JET(NCMX), WH2PLX,
7 TIMQNH(NCMX), TIMQNB
COMMON/XRCSHS/TG(26, ILS), MFH2(26, ILS), MFAIR(26, ILS),
1 TPIO(JMPO, IMPO, ICHS, ILS), TPNL(JMPN, IMPN, ILS),
1 TSTD(JMSD, IMSD, ILS), TDCR(JMDC, IMD, 2, ILS),
2 TWTD(JMWD, IMWD, ILS), TPII(JMPI, IMPI, ICHS, ILS),
3 XOXN(16, 16, ILS)
COMMON/UNITS/ IUNITS
PARAMETER (IFMAX=25, IPMAX=99)
INTEGER UFILE(IFMAX), NPTS(IFMAX), NPFIL, PLTPTR(IPMAX, IFMAX)

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 5

```

COMMON/SETPLT/ UFIL, NPTS, NPFL,
  IUNTIM(IFMAX), TIMOFF(IFMAX), PLTPTR
COMMON/FLGS/I_EVENT(MAXSHAPES), I_EVENT_OLD(MAXSHAPES),
  I_EVENT_AUTO(MAXSHAPES)
CHARACTER*1 KEY
COMMON/YNOT/KEY
CHARACTER*8 UNITTS
PARAMETER (IMOV=100)
COMMON/MOVIE/VALS(IMOV), UNITTS(IMOV), ICOUNT, IMOVIE, INOPLT
CHARACTER*32 VNAME, LABL
CHARACTER*20 TOLD
REAL MOUT
PARAMETER (IMPV1=100, IMPV2=2, IMPV3=20)
COMMON/MPVAR/MOUT(IMPV1, IMPV2), TOLD(IMPV1, IMPV3), VNAME(IMPV1),
  ICORE, LABL(IMPV1), CHSIZ, SCALE, IREPT, IMVER, DXNOD, DYNOD, CHNOD
CHARACTER*80 FNAMES
INTEGER FNUM
PARAMETER (IFNM=12)
COMMON/IOFNME/FNAMES(IFNM), FNUM
COMMON/FLAGS/IFLAGP

C$REPLAY
COMMON/DSTH2/ TWDS, PSATDS, VSTTDS, HSTTDS,
1 PDS, TSATDS, VWPDPS, VSTPDS, HWPDPS, HSTPDS,
2 MUWDS, KWDS, CVWDS, HGDS, HSTDPS, TGDS,
3 MFSTDPS, MFH2DS, MFN2DS, MFO2DS, MFC2DS, MFCODS, MFHEDS,
4 VOLWDS, VOLGDS, ZWDS, VWDS, HWDS,
5 PPSTDPS, VSTDPS, USTDPS, CVSTDPS, HTCPDPS, GAMMDS, VEXDS,
6 VGDS, VFDS, CFDS, MUGDS, CPGDS, KGDS, DIFDS, MGDS,
7 MFHWDS, MFHSTDPS, XWDS, WSTDPS, QSTDPS,
8 WWOVER, QWOVER, WWDOWN, QWDOWN, WSTDOW, QGDOWN, WST2
COMMON/HSRB1/HSTYP(NHSX), NIWALL(NHSX), AHSRB(NHSX), XTHSRB(NHSX),
1XHSRB(NHSX), ZHSRB(NHSX), N1HSRB(NHSX), N2HSRB(NHSX),
2XLN1(NHSX), XLN2(NHSX), RG1(NHSX), RG2(NHSX), HTEXT,
3WWSP1(NHSX), WWSP2(NHSX), TWSP1(NHSX), TWSP2(NHSX), XPERHS(NHSX),
4AHSRB1(NHSX), THS10(NHSX), THS20(NHSX), THETHS(NHSX)
COMMON/HSRB2/KHSRB(NHSX), CPHSRB(NHSX), DHSRB(NHSX), CPEQRB(NHSX),
1 TS1(NHSX), TS2(NHSX), WCDTRB(NNODX), THSFP(NNODX),
2 TF1(NHSX), TF2(NHSX), TFEQP(NHSX), NHSFP(NNODX), ALHS(NNODX),
3 WWF1(NHSX), QWF1(NHSX), WWF2(NHSX), QWF2(NHSX), QHSTRB(NNODX),
4 QWIX(NHSX), QWOX(NHSX), QGEQX(NHSX),
5 QCDWI(NHSX), QCDWO(NHSX), QCDEQP(NHSX), QWEQX(NHSX),
6 QFPHS(2, NHSX), SHLRB(2, NHSX), QRAHS1(NHSX), QRAHS2(NHSX)
COMMON /RCS6/
1  WSTCR(ICHS, ILS), WH2CR(ICHS, ILS), WN2CR(ICHS, ILS), FPPS1(2),
2  WC2CR(ICHS, ILS), WO2CR(ICHS, ILS), WCOCR(ICHS, ILS), PHIGRV(2),
3  WHECR(ICHS, ILS), WHECR1(ICHS, ILS), WSTCRI(ICHS, ILS), ARVPS(2),
4  WH2CRI(ICHS, ILS), WN2CRI(ICHS, ILS), WC2CRI(ICHS, ILS), ASVPS(2),
5  WO2CRI(ICHS, ILS), WCOCR1(ICHS, ILS), QCCR(ICHS, ILS), MAIRT(ILS),
6  QGCRI(ICHS, ILS), VPS, PPS(ILS), MFSTPS(ILS), MFH2PS(ILS),
7  MFN2PS(ILS), MFO2PS(ILS), MFC2PS(ILS), MFCCPS(ILS), MFHEPS(ILS),
8  QGHSPS(ILS), TGPS(ILS), PPSTPS(ILS), PSATPS(ILS), HSTW(ILS),
  . HWSFPS(ILS), HSTSTP(ILS), TWSTPS(ILS), HSTPS(ILS), USTPS(ILS),
1  HTCPPS(ILS), VSTPS(ILS), VGPS(ILS), VOLGPS(ILS), MSWG(ILS),
2  VFPS(ILS), CFPS(ILS), MUGPS(ILS), CPGPS(ILS), KGPS(ILS),
3  DIFPS(ILS), MGPS(ILS), CVST(ILS), DMDTG(ILS), GAMMPS(ILS),
4  GMSTPS(ILS), TSATPS(ILS), CVWPS(ILS), MFAIRT(ILS),

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 6

```

5      SURT(ICH5,ILS), VOLWTU(ICH5,ILS), VOLWSD(ILS), VOLWDC(2,ILS),
6      VOLWWD(ILS), VOLWPI(5,ILS), GBKMX(ILS), WRPSNT(ILS), PLOWRV(2),
7      QRPSNT(2), WSWG(2), WSTMSL(2), WGRV(2), NCYCLE(10,2), WWFED(2),
8      WWRECI(ILS), MFHWP5(ILS), MFHSTPS(ILS), MFHWSD(ILS)
COMMON/SKTH2/ TWSK, PSATSK, VSTTSK, HSTTSK,
1 PSK, TSATSK, VWPSK, VSTPSK, HWPSK, HSTPSK,
2 MUWSK, KWSK, CVWSK, HGSK, HSTSK, TGSK,
3 MFSTSK, MFH2SK, MFN2SK, MFO2SK, MFC2SK, MFCOSK, MFHESK,
4 VOLWSK, VOLGSK, ZWSK, VWSK, HWSK,
5 PPSTSK, VSTSK, USTSK, CVSTSK, HTCPSK, GAMMSK, VEXSK,
6 VGSK, VFSK, CFSK, MUGSK, CPGSK, KGSK, DIFSK, MGSK,
7 MFHWSK, MFHSTSK, KSTSK, MUSTSK, CPSTSK,
8 UHSMLTSK, UHSMLXSK, TGSKOL
COMMON/CTTH2/ TWCT, PSATCT, VSTTCT, HSTTCT,
1 PCT, TSATCT, VWPC, VSTPCT, HWPC, HSTPCT,
2 MUWC, KWCT, CVWCT, HGCT, HSTCT, TGCT,
3 MFSTCT, MFH2CT, MFN2CT, MFO2CT, MFC2CT, MFCOCT, MFHECT,
4 VOLWC, VOLGCT, ZWCT, VWC, HWCT,
5 PPSTCT, VSTCT, USTCT, CVSTCT, HTCPCT, GAMMCT, VEXCT,
6 VGCT, VFCT, CFCT, MUGCT, CPGCT, KGCT, DIFCT, MGCT,
7 MFHWCT, MFHSTCT, KSTCT, MUSTCT, CPSTCT, UHSMLTCT, UHSMLXCT,
8 TGCTOL, TGDSOL, WSTCTO, MUWPCT, KWPC, CVWPCT
COMMON /CTDB4/ TCMCT(ILS), TSOLCMCT(ILS), TLIQCMCT(ILS), FSCMCT(ILS),
1 MCMCT(ILS), VLCMCT(ILS), CPCMCT(ILS), DCMCT(ILS), KCMCT(ILS),
2 VSCMCT(ILS), HCMCT(ILS), LHCMCT(ILS), TCMMPC(ILS), STCMCT(ILS),
3 QCMHS(ILS), QCMCT(ILS), ASSCT(ILS), WGOUTC,
4 QPLHIJ(6,ILS), QSLHIJ(6,ILS), QVSLL(6,ILS), MSSTCT(ILS),
5 QSTCMCT(ILS), WSTCMCT(ILS), AXLCT(6,ILS), XTWALL(ILS),
6 HTSLVW(ILS), QASLVW(ILS), TSSAC01, TSSAC02, FVFAIL, QVSLT(ILS)
COMMON /SKDB4/ TCMK, TSOLCMK, TLIQCMK, FSCMSK,
1 MCMTSK, VLCMSK, CPCMSK, DCMSK, KCMMSK,
2 VSCMSK, HCMSK, LHCMMSK, TCMMPSK, STCMMSK,
3 QCMHSK, QCMSK, ASSSK, WGOUTS,
4 QPLHSK(6), QSLHSK(6), QVSLSK(6), MSSTSK,
5 QSTCMSK, WSTCMSK, AXLSK(6), QASWSK, TSSASKW(15),
6 QGASSK, WH2SSK, WSTSSK
DIMENSION M0(5,10)
DATA M0/50*0.E0/
DIMENSION FMASS(CMAXRAD, CMAXAX, ICST), ZLEV(MAXSHAPES),
     . TCRM(CMAXRAD), TPIC(CMAXRAD, CMAXAX, ICST),
     . MFH2C(15), INDX(15)
DATA INDX/58, 124, 256, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295,
     . 2*0/
CREV-MAM START USE PASSED VALUE OF TDZPIC
     DATA TIZPIC/0.E0/, TISCRN/0.E0/
CREV-MAM END
     DATA ICAL, INTRPT/2*0/, IBATCH/0/, JSRV/1/
CPicasso-3 code inserted
COMMON /PIC3LEV/ ZLEV
COMMON /PIC3NOD/ IP3MRAD, IP3MAX, IP3ICST,
     . TPIC,
     . FMASS
     common /PIC3CNT/ INTP3
c If interupted, jump to event updates

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 7

```

if (intp3.eq.1) goto 9999

IP3MRAD = CMAXRAD
IP3MAX  = CMAXAX
IP3ICST = ICST
CEND Picasso-3 inserted code

IF (ICAL.EQ.0) THEN
  IMVER=1
  DXNOD=.007
  DYNOD=.02
  CHNOD=.01
  IPAINT=1
CREV-MAM START PASS IRMAX, JCHMAX, TIM, IMVUNT
c      IF (IMVUNT.GT.0)
c      CALL WMOVIE(ICAL,FMASS,TCRM,TPIC,ZLEV,
c                  JCHMAX,IRMAX,TIM,IMVUNT)
CREV-MAM END
ENDIF
C Test for interrupt, if so set update time to present
100  IF(KEY.EQ.'I'.OR.KEY.EQ.'i') TISCRN=TIM-TTSCRN*IDISON
     IF(KEY.EQ.'I'.AND.ITMLPS.GT.0) ICORE=1
C Is graphics display active?
  IF (ISCRN.EQ.1) THEN
    IMOVIE=-2
  ELSEIF (IBATCH.EQ.0.AND.ISCRN.EQ.0) THEN
    IMOVIE=0
  ENDIF
C Set movie flag to generate playback data in batch mode
  IF (IBATCH.NE.0) IMOVIE=-1
C If not time to update screen or write playback, return
  IF (IBATCH.EQ.0 .AND. TIM.LT.TISCRN+TTSCRN*IDISON .AND.
      . TIM.LT.TIZPIC+TDZPIC) RETURN
  IF (IBATCH.NE.0 .AND. TIM.LT.TIZPIC+TDZPIC) RETURN
*****
C Normalized water levels
*****
CALL LEVELP1(ZLEV)
*****
C Control rods
*****
TDSH=MAX(TIM-TISCRM, 0.)
IF(TDSH.EQ.0) THEN
  I_EVENT(237) = 1
  I_EVENT(240) = 1
  I_EVENT(243) = 1
ELSE
  I_EVENT(237) = 0
  I_EVENT(240) = 0
  I_EVENT(243) = 0
ENDIF
C Set up logic for pumps, valves and lines.
C EVENT CODES...
C   0 = OFF (GREEN,VALVE CLOSED)
C   1 = ON (BLUE,VALVE OPEN)
C  -1 = FAILED (RED,VALVE IN EITHER POSITION)

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 8

```

C SRVs
C (Loop 1)
  I_EVENT(251)=IOPEN(JSRV, 1)
  I_EVENT(250)=I_EVENT(251)
  I_EVENT(252)=I_EVENT(250)
C (Loop 2)
  I_EVENT(254)=IOPEN(JSRV, 2)
  I_EVENT(253)=I_EVENT(254)
  I_EVENT(255)=I_EVENT(253)
C MSIVs
C (Loop 1)
  I_EVENT(185)=1-IEVNT(3)
  I_EVENT(184)=I_EVENT(185)
  I_EVENT(186)=I_EVENT(185)
  I_EVENT(187)=I_EVENT(186)
  I_EVENT(188)=I_EVENT(187)
  IF (IEVNT(405).EQ.1.OR.IEVNT(407).EQ.1) I_EVENT_AUTO(185) = 0
C (Loop 2)
  I_EVENT(82)= 1-IEVNT(4)
  I_EVENT(81)=I_EVENT(82)
  I_EVENT(83)=I_EVENT(82)
  I_EVENT(84)=I_EVENT(83)
  I_EVENT(85)=I_EVENT(84)
  IF (IEVNT(406).EQ.1.OR.IEVNT(408).EQ.1) I_EVENT_AUTO(82) = 0
C Containment sprays (external)
  I_EVENT(77)=0
  IF (WWSP2(35).GT.0.E0.OR.WWF2(35).GT.0.0) I_EVENT(77)=1
  I_EVENT(108) = I_EVENT(77)
  I_EVENT(183) = I_EVENT(108)
  I_EVENT(279) = I_EVENT(183)
C Containment sprays (internal)
  I_EVENT(141)= IEVNT(213)
  IF (IEVNT(402).EQ.1) I_EVENT(141)=-1
  I_EVENT(149)= I_EVENT(141)
  I_EVENT(134)= I_EVENT(149)
  I_EVENT(169)= I_EVENT(134)
  I_EVENT(170)= I_EVENT(169)
  I_EVENT(171)= I_EVENT(170)
  I_EVENT(182)= I_EVENT(171)
  I_EVENT(278)= I_EVENT(182)
  IF (IEVNT(444).EQ.1.OR.IEVNT(445).EQ.1) I_EVENT_AUTO(141) = 0
C Suction from CST
  I_EVENT(153)= MIN(IEVNT(213),1-IEVNT(316))
  IF (IEVNT(402).EQ.1) I_EVENT(153)=-1
  I_EVENT(154)= I_EVENT(153)
  I_EVENT(155)= I_EVENT(154)
C Recirc Mode
  I_EVENT(162)= MIN(IEVNT(213),1-I_EVENT(153))
  IF (IEVNT(402).EQ.1) I_EVENT(162)=-1
  I_EVENT(161)= I_EVENT(162)
  I_EVENT(163)= I_EVENT(161)
  I_EVENT(138)= I_EVENT(163)
  I_EVENT(178)= I_EVENT(138)
  I_EVENT(180)= I_EVENT(178)
  I_EVENT(18)= I_EVENT(180)
  I_EVENT(19)= I_EVENT(18)

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 9

```

C Main feedwater
C (Loop 1)
I_EVENT(211)= 1-IEVNT(5)
IF (IEVNT(402).EQ.1) I_EVENT(211)=-1
I_EVENT(210)=I_EVENT(211)
I_EVENT(209)=I_EVENT(210)
I_EVENT(208)=I_EVENT(209)
I_EVENT(207)=I_EVENT(208)
I_EVENT(206)=I_EVENT(207)
I_EVENT(205)=I_EVENT(206)
I_EVENT(204)=I_EVENT(205)
I_EVENT(203)=I_EVENT(204)
I_EVENT(202)=I_EVENT(203)
I_EVENT_AUTO(211) = -1
C (Loop 2)
I_EVENT(310)= 1-IEVNT(6)
IF (IEVNT(402).EQ.1.AND.IEVNT(410).EQ.0) I_EVENT(310)=-1
I_EVENT(105)=I_EVENT(310)
I_EVENT(104)=I_EVENT(105)
I_EVENT(103)=I_EVENT(104)
I_EVENT(102)=I_EVENT(103)
I_EVENT(101)=I_EVENT(102)
I_EVENT(100)=I_EVENT(101)
I_EVENT(99)=I_EVENT(100)
I_EVENT(61)=I_EVENT(99)
I_EVENT(98)=I_EVENT(61)
I_EVENT(97)=I_EVENT(98)
I_EVENT(92)=I_EVENT(97)
I_EVENT_AUTO(310) = -1
C LPCI
C (Loop 1)
I_EVENT(220)= IEVNT(355)
IF (IEVNT(402).EQ.1) I_EVENT(220)=-1
IF (IEVNT(427).EQ.1.OR.IEVNT(428).EQ.1) I_EVENT_AUTO(220) = 0
C Inject into lower header
I_EVENT(219)= I_EVENT(220)
I_EVENT(218)= I_EVENT(219)
I_EVENT(217)= I_EVENT(218)
I_EVENT(216)= I_EVENT(217)
I_EVENT(215)= I_EVENT(216)
C (Loop 2)
I_EVENT(139)= IEVNT(356)
IF (IEVNT(402).EQ.1) I_EVENT(139)=-1
I_EVENT(144)= I_EVENT(139)
I_EVENT(24)= I_EVENT(144)
I_EVENT(127)= I_EVENT(24)
I_EVENT(128)= I_EVENT(127)
I_EVENT(129)= I_EVENT(128)
I_EVENT(130)= I_EVENT(129)
I_EVENT(167)= I_EVENT(130)
IF (IEVNT(429).EQ.1.OR.IEVNT(430).EQ.1) I_EVENT_AUTO(139) = 0
C Inject into lower header
I_EVENT(112)= I_EVENT(167)
I_EVENT(111)= I_EVENT(112)
I_EVENT(110)= I_EVENT(111)
I_EVENT(109)= I_EVENT(110)

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 10

```

I_EVENT(107) = I_EVENT(109)
I_EVENT(106) = I_EVENT(107)
C Suction from CST
I_EVENT(152) = MIN(I_EVENT(139),1-IEVNT(315))
I_EVENT(151) = I_EVENT(152)
I_EVENT(146) = I_EVENT(151)
C Recirc Mode
I_EVENT(138) = MIN(I_EVENT(139),IEVNT(315))
I_EVENT(151) = MAX(I_EVENT(138),I_EVENT(152))
I_EVENT(160) = I_EVENT(138)
I_EVENT(178) = I_EVENT(160)
I_EVENT(179) = I_EVENT(178)
I_EVENT(180) = I_EVENT(179)
I_EVENT(18) = I_EVENT(180)
I_EVENT(19) = I_EVENT(18)
C HPCI
C (Loop 1)
I_EVENT(195) = IEVNT(353)
IF (IEVNT(402).EQ.1) I_EVENT(195)=-1
I_EVENT(194) = I_EVENT(195)
I_EVENT(193) = I_EVENT(194)
I_EVENT(191) = I_EVENT(193)
I_EVENT(190) = I_EVENT(191)
I_EVENT(189) = I_EVENT(190)
IF (IEVNT(423).EQ.1.OR.IEVNT(424).EQ.1) I_EVENT_AUTO(195) = 0
C (Loop 2)
I_EVENT(140) = IEVNT(354)
IF (IEVNT(402).EQ.1) I_EVENT(140)=-1
I_EVENT(148) = I_EVENT(140)
I_EVENT(147) = I_EVENT(148)
I_EVENT(136) = I_EVENT(147)
I_EVENT(165) = I_EVENT(136)
I_EVENT(131) = I_EVENT(165)
I_EVENT(166) = I_EVENT(131)
I_EVENT(168) = I_EVENT(166)
I_EVENT(90) = I_EVENT(168)
I_EVENT(192) = I_EVENT(90)
I_EVENT(87) = I_EVENT(192)
I_EVENT(88) = I_EVENT(87)
I_EVENT(86) = I_EVENT(88)
IF (IEVNT(425).EQ.1.OR.IEVNT(426).EQ.1) I_EVENT_AUTO(140) = 0
C RHR
C (Loop 1)
I_EVENT(214) = IEVNT(433)
I_EVENT(213) = I_EVENT(214)
I_EVENT(212) = I_EVENT(213)
I_EVENT(224) = I_EVENT(214)
I_EVENT(223) = I_EVENT(224)
I_EVENT(222) = I_EVENT(223)
I_EVENT(221) = I_EVENT(222)
I_EVENT_AUTO(214) = -1
C (Loop 2)
I_EVENT(142) = IEVNT(434)
I_EVENT(174) = I_EVENT(142)
I_EVENT(173) = I_EVENT(174)
I_EVENT(172) = I_EVENT(173)

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 11

```

I_EVENT(135)= I_EVENT(172)
I_EVENT(150)= I_EVENT(135)
I_EVENT(158)= I_EVENT(142)
I_EVENT(156)= I_EVENT(158)
I_EVENT(157)= I_EVENT(156)
I_EVENT(159)= I_EVENT(157)
I_EVENT(137)= I_EVENT(159)
I_EVENT(175)= I_EVENT(137)
I_EVENT(176)= I_EVENT(175)
I_EVENT(177)= I_EVENT(176)
I_EVENT(116)= I_EVENT(177)
I_EVENT(115)= I_EVENT(116)
I_EVENT(114)= I_EVENT(115)
I_EVENT_AUTO(142) = -1

C Accumulator
C (Loop 1)
  IF (I_EVENT(220).NE.-1)THEN
    I_EVENT(220)= MAX(IEVNT(355),IEVNT(351))
  ENDIF
  I_EVENT(219)= I_EVENT(220)
  I_EVENT(218)= I_EVENT(219)
  I_EVENT(217)= I_EVENT(218)
  I_EVENT(216)= I_EVENT(217)
  I_EVENT(215)= I_EVENT(216)

C (Loop 2)
  I_EVENT(126)= IEVNT(352)
  IF (IEVNT(402).EQ.1) I_EVENT(126)=-1
  I_EVENT(125)=I_EVENT(126)
  I_EVENT(24)=I_EVENT(126)
  IF (I_EVENT(112).NE.-1)THEN
    I_EVENT(112)= MAX(I_EVENT(167),I_EVENT(125))
  ENDIF
  I_EVENT(111)= I_EVENT(112)
  I_EVENT(110)= I_EVENT(111)
  I_EVENT(109)= I_EVENT(110)
  I_EVENT(107)= I_EVENT(109)
  I_EVENT(106)= I_EVENT(107)
  I_EVENT_AUTO(126) = -1

C RCIC
C (Loop 1)
  I_EVENT(225)= IEVNT(357)
  IF (IEVNT(402).EQ.1) I_EVENT(225)=-1
  I_EVENT(249)= MAX(I_EVENT(225),I_EVENT(205))
  I_EVENT(204)= I_EVENT(249)
  I_EVENT(203)= I_EVENT(204)
  I_EVENT(202)= I_EVENT(203)
  I_EVENT_AUTO(225) = -1

C (Loop 2)
  I_EVENT(143)= IEVNT(358)
  IF (IEVNT(402).EQ.1) I_EVENT(143)=-1
  I_EVENT(164)= I_EVENT(143)
  I_EVENT(50)= I_EVENT(143)
  I_EVENT(54)= I_EVENT(50)
  I_EVENT(61)= MAX(I_EVENT(54),I_EVENT(99))
  I_EVENT(98)= I_EVENT(61)
  I_EVENT(97)= I_EVENT(98)

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 12

```

I_EVENT(92)= I_EVENT(97)
I_EVENT_AUTO(143) = -1
C RCIC HX
I_EVENT(34)= MAX(IEVNT(437),I_EVENT(143))
I_EVENT(44)= I_EVENT(34)
I_EVENT_AUTO(34) = -1
C Suppression Pool Cooling
I_EVENT(9)= IEVNT(215)
IF (IEVNT(402).EQ.1) I_EVENT(9)=-1
I_EVENT(15)= I_EVENT(9)
I_EVENT(8)= I_EVENT(15)
I_EVENT(181)= I_EVENT(8)
I_EVENT(6)= I_EVENT(181)
I_EVENT(17)= I_EVENT(8)
I_EVENT(18)= MAX(I_EVENT(17),I_EVENT(180))
I_EVENT(19)= I_EVENT(18)
I_EVENT_AUTO(9) = -1
C Heavy Water Cooling System
I_EVENT(26)= 0
IF (WWDOWN.GT.0.0) I_EVENT(26)= 1
I_EVENT(27)= IEVNT(111)
IF (IEVNT(402).EQ.1) I_EVENT(27)=-1
I_EVENT(28)= I_EVENT(27)
I_EVENT(29)= I_EVENT(28)
I_EVENT(31)= I_EVENT(29)
I_EVENT(35)= I_EVENT(31)
I_EVENT(36)= I_EVENT(35)
I_EVENT(247)= I_EVENT(36)
I_EVENT(246)= I_EVENT(247)
I_EVENT(245)= I_EVENT(246)
I_EVENT_AUTO(27) = -1
C Thermal Shield Cooling
I_EVENT(41)= IEVNT(159)
IF (IEVNT(402).EQ.1) I_EVENT(41)=-1
I_EVENT(45)= I_EVENT(41)
I_EVENT(42)= I_EVENT(45)
I_EVENT(46)= I_EVENT(42)
I_EVENT(48)= I_EVENT(46)
I_EVENT(47)= I_EVENT(48)
I_EVENT(55)= I_EVENT(47)
I_EVENT(53)= I_EVENT(55)
I_EVENT(56)= I_EVENT(53)
I_EVENT(57)= I_EVENT(56)
I_EVENT(38)= I_EVENT(57)
I_EVENT(37)= I_EVENT(38)
I_EVENT(39)= I_EVENT(37)
I_EVENT(40)= I_EVENT(39)
I_EVENT(43)= I_EVENT(40)
I_EVENT_AUTO(41) = -1
C Moderator Dump Valve
I_EVENT(21)= IEVNT(134)
I_EVENT(23)= I_EVENT(21)
I_EVENT(22)= I_EVENT(23)
IF (IEVNT(446).EQ.1.OR.IEVNT(447).EQ.1) I_EVENT_AUTO(21) = 0
C Annulus Exhaust System
I_EVENT(60)= 1

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 13

```

I_EVENT(64)= I_EVENT(60)
I_EVENT(65)= I_EVENT(64)
I_EVENT(263)= I_EVENT(65)
C Containment Air Cooling
I_EVENT(67)= IEVNT(209)
IF (IEVNT(402).EQ.1) I_EVENT(67)=-1
I_EVENT(68)= I_EVENT(67)
I_EVENT(66)= I_EVENT(68)
I_EVENT(72)= I_EVENT(66)
I_EVENT(70)= I_EVENT(72)
IF (IEVNT(439).EQ.1.OR.IEVNT(440).EQ.1) I_EVENT_AUTO(67) = 0
C HX
I_EVENT(71)= IEVNT(438)
I_EVENT(73)= I_EVENT(71)
C Recirc Pump
C (Loop 1)
C      I_EVENT(49)= (1-IEVNT(7).OR.WWRECI(1).GT.0.0)

IF ( 1-IEVNT(7).eq.1 .OR. WWRECI(1).GT.0.0) then
  I_EVENT(49) = 1
else
  I_EVENT(49) = 0
endif

IF (IEVNT(402).EQ.1) I_EVENT(49)=-1
I_EVENT(305)= I_EVENT(49)
I_EVENT(52)= I_EVENT(305)
I_EVENT(51)= I_EVENT(52)
I_EVENT(7)= I_EVENT(51)
I_EVENT(5)= I_EVENT(7)
I_EVENT_AUTO(49) = -1
C (Loop 2)
C      I_EVENT(122)= (1-IEVNT(8).OR.WWRECI(2).GT.0.0)

if (1-IEVNT(8).eq.1 .OR. WWRECI(2).GT.0.0) then
  I_EVENT(122) = 1
else
  I_EVENT(122) = 0
endif

IF (IEVNT(402).EQ.1) I_EVENT(122)=-1
I_EVENT(229)= I_EVENT(122)
I_EVENT(123)= I_EVENT(229)
I_EVENT(120)= I_EVENT(123)
I_EVENT(119)= I_EVENT(120)
I_EVENT(117)= I_EVENT(119)
I_EVENT_AUTO(122) = -1
C Outlet Riser Tubes
I_EVENT(12)= 1
I_EVENT(257)= 1
I_EVENT(258)= 1
I_EVENT(259)= 1
I_EVENT(260)= 1
I_EVENT(76)= 1
C Inlet Feeder Tubes
I_EVENT(248)= 1

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic api.f ページ 14

```

I_EVENT(230)= 1
I_EVENT(231)= 1
I_EVENT(232)= 1
I_EVENT(233)= 1
I_EVENT(234)= 1
I_EVENT(235)= 1
I_EVENT(79)= 1
C Containment
CREV-MAM START
  JNX= JNSK
  IF (JNX.GT.0) THEN
    IF (INCM(JNX).GT.0) THEN
      TCMX= TCMB(INCM(JNX))
    ELSE
      TCMX=0.
    ENDIF
  ELSE
    TCMX= 0.E0
  ENDIF
  TCRM(4)=SCT(TCMX, IUNITS)
C   TCRM(1)=SCT(TCMC, IUNITS)
CREV-MAM END
C Calandria tank
  TCRM(3)=SCT(MAX(TCMCT(1), TCMCT(2)), IUNITS)
C Shield Tank
  TCRM(1)=SCT(TCMISK, IUNITS)
  TCRM(2)=SCT(TCMISK, IUNITS)
C Clean up conversion of core node temperatures and masses
C from MAAP to MAAP-GRAAPH (all MG variables are 2 dimensional)
C Also allow for core representation size to be defined by the parameter
C file. This means do not hard code sizes but use JCHMAX and IRMAX.
C Core temps in a 10 x 3 representation
C Loop through core nodes assigning temperatures
C and normalizing mass of fuel.
C PWR MAAP stores the core node temps and masses in single subscripted
C arrays starting the lower left at 1 and increasing up each column.
C MAAP-GRAAPH stores the core node temps and normalized masses in
C double subscripted arrays starting the lower left at (1,1) (column, row).
C
C      MAAP                      MAAP-GRAAPH
C -----
C      | IRMAX                   | | 1,IRMAX |
C      | 9 .                     | | 1,9 .   |
C      | 8 .                     | | 1,8 .   |
C      | 7 .                     | | 1,7 .   |
C      | 6 16                    | | 1,6 2,6 |
C      | 5 15                    | | 1,5 2,5 |
C      | 4 14                    | | 1,4 2,4 |
C      | 3 13                    | | 1,3 2,3 |
C      | 2 12                    | | 1,2 2,2 |
C      | 1 11 ...                | | 1,1 2,1 ... |
C -----
C Calculate total number of core nodes
  ITNODS = IRMAX
C Calculate Loop 1
  ILOOP = 1

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 15

```

ICH = 4
C ICOL = column index (axial) bottom to top
C IROW = row index (radial) left to right
DO K = 1, 3
ICH = ICH - 1
DO I = 1, ITNODS
ICOL = ( (I - 1) / IRMAX ) + 1
IROW = I - (ICOL-1) * IRMAX
C Temperature
TPIC(ICOL,IROW,K) = SCT(TNOD(IROW,ICH,ILOOP),IUNITS)
C Calculate normalized core node mass
C Using:
C Mass of U2 per node +
C Mass of ZR alloy per node +
C Mass of ZR oxide per node
C Divided by
C Initial Mass per node
FMASS(ICOL,IROW,K)= MAX(0.E0, MIN(1.E0,MNOD(IROW,ICH,ILOOP)
/ MNOD0(I)))
CREV-MAM END
ENDDO
ENDDO
C Calculate Loop 2
ILOOP = 2
ICH = 0
C ICOL = column index (axial) bottom to top
C IROW = row index (radial) left to right
DO K = 4, 6
ICH = ICH + 1
DO I = 1, ITNODS
ICOL = ( (I - 1) / IRMAX ) + 1
IROW = I - (ICOL-1) * IRMAX
C Temperature
TPIC(ICOL,IROW,K) = SCT(TNOD(IROW,ICH,ILOOP),IUNITS)
C Calculate normalized core node mass
C Using:
C Mass of U2 per node +
C Mass of ZR alloy per node +
C Mass of ZR oxide per node
C Divided by
C Initial Mass per node
FMASS(ICOL,IROW,K)= MAX(0.E0, MIN(1.E0,MNOD(IROW,ICH,ILOOP)
/ MNOD0(I)))
CREV-MAM END
ENDDO
ENDDO
ICAL = 1
C Containment Failed Flag
I_EVENT(77) = IEVNT(203)
*****
C HYDROGEN CONCENTRATIONS
*****
MFH2C(1)=MFH2CT
MFH2C(2)=MFH2SK
MFH2C(3)=MFH2SK
MFH2C(4)=MFH2(11,1)

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 16

```

MFH2C(5)=MFH2(11, 2)
MFH2C(6)=MFH2(14, 1)
MFH2C(7)=MFH2(14, 2)
MFH2C(8)=MFH2(3, 1)
MFH2C(9)=MFH2(2, 1)
MFH2C(10)=MFH2(1, 1)
MFH2C(11)=MFH2(1, 2)
MFH2C(12)=MFH2(2, 2)
MFH2C(13)=MFH2(3, 2)
DO 117 J=1,15
  IF (INDX(J).EQ.0) GO TO 117
  I_EVENT(INDX(J))=-2
  IF (MFH2C(J).GE.1.0E-5.AND.MFH2C(J).LT.1.0E-3) I_EVENT(INDX(J))=0
  IF (MFH2C(J).GE.1.0E-3.AND.MFH2C(J).LT.1.0E-2) I_EVENT(INDX(J))=1
  IF (MFH2C(J).GE.1.0E-2) I_EVENT(INDX(J))=-1
C If vessel concentration changes, update innards
  IF (I_EVENT(INDX(1)).NE.I_EVENT_OLD(INDX(1))) THEN
    DO 132 JKL=189,198
      I_EVENT_OLD(JKL)=-2
132      CONTINUE
    ENDIF
117  CONTINUE
  IF (TIM.GE.TIZPIC+TDZPIC) THEN
    TIZPIC=TIM
CREV-MAM START PASS IRMAX, JCHMAX, TIM, IMVUNT
C       IF (IMVUNT.GT.0)
C       CALL WMOVIE(ICAL,FMASS,TCRM,TPIC,ZLEV,
C                   JCHMAX,IRMAX,TIM,IMVUNT)
CREV-MAM END
  ENDIF
5   FORMAT(10A8)
999  CONTINUE
C If maap-graaph is not active return
  IF (IMG.EQ.0) RETURN
  IF (TIM.LT.TISCRN+TTSCRN*IDISON .AND. IDISON.EQ.1) RETURN
C Write screen for interactive
  TISCRN=TIM
CREV-MAM START PASS IRMAX, JCHMAX, IUNITS, TIM, TILAST
C       CALL DISPLA1(ZLEV,TPIC,FMASS,TCRM,TTSCRN,INTRPT,
C                   IRMAX,JCHMAX,IUNITS,TIM,TILAST)
CREV-MAM END
  IDISON=1
CREV-MAM END
  IF (IMOVIE.LT.0) RETURN

9999 continue

C Set event flags associated with ESFS if changed by maap-graaph
  IF (IEVNT(402).EQ.1) GO TO 219
C ESF suction alignment
  IF (I_EVENT_OLD(19).EQ.-2) THEN
    I_EVENT_OLD(19)=I_EVENT(19)
    IF (I_EVENT(19).EQ.1) THEN
      I_EVENT(23)=1
      I_EVENT(43)=0
      I_EVENT(70)=0
    ENDIF
  ENDIF

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 17

```

IEVNT(220)=0
ELSE
  I_EVENT(23)=0
  I_EVENT(43)=1
  I_EVENT(70)=1
  IEVNT(220)=1
ENDIF
  GO TO 997
ENDIF
IF(I_EVENT_OLD(23).EQ.-2) THEN
  I_EVENT_OLD(23)=I_EVENT(23)
    IF(I_EVENT(23).EQ.1) THEN
      I_EVENT(19)=1
      I_EVENT(43)=0
      I_EVENT(70)=0
      IEVNT(220)=0
    ELSE
      I_EVENT(19)=0
      I_EVENT(43)=1
      I_EVENT(70)=1
      IEVNT(220)=1
    ENDIF
    GO TO 997
  ENDIF
IF(I_EVENT_OLD(43).EQ.-2) THEN
  I_EVENT_OLD(43)=I_EVENT(43)
    IF(I_EVENT(43).EQ.1) THEN
      I_EVENT(70)=1
      I_EVENT(19)=0
      I_EVENT(23)=0
      IEVNT(220)=1
    ELSE
      I_EVENT(70)=0
      I_EVENT(19)=1
      I_EVENT(23)=1
      IEVNT(220)=0
    ENDIF
    GO TO 997
  ENDIF
IF(I_EVENT_OLD(70).EQ.-2) THEN
  I_EVENT_OLD(70)=I_EVENT(70)
    IF(I_EVENT(70).EQ.1) THEN
      I_EVENT(43)=1
      I_EVENT(19)=0
      I_EVENT(23)=0
      IEVNT(220)=1
    ELSE
      I_EVENT(43)=0
      I_EVENT(19)=1
      I_EVENT(23)=1
      IEVNT(220)=0
    ENDIF
  ENDIF
 997  CONTINUE
C Containment sprays (internal)
IF(I_EVENT_OLD(141).EQ.-2) THEN

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 18

```

I_EVENT_OLD(141)=I_EVENT(141)
IF(I_EVENT(141).EQ.1) THEN
  IEVNT(444)=1
  IEVNT(445)=0
ELSEIF(I_EVENT(141).EQ.0) THEN
  IEVNT(444)=0
  IEVNT(445)=1
ELSE
  IEVNT(444)=0
  IEVNT(445)=0
ENDIF
ENDIF

C Containment sprays Suction from CST (loop 2)
IF(I_EVENT_OLD(154).EQ.-2) THEN
  I_EVENT_OLD(154)=I_EVENT(154)
  IF(I_EVENT(154).EQ.1) THEN
    IEVNT(419)=0
  ELSE
    IEVNT(419)=1
  ENDIF
ENDIF

C Containment sprays Suction from Suppression Pool (loop 2)
IF(I_EVENT_OLD(161).EQ.-2) THEN
  I_EVENT_OLD(161)=I_EVENT(161)
  IF(I_EVENT(161).EQ.1) THEN
    IEVNT(419)=1
  ELSE
    IEVNT(419)=0
  ENDIF
ENDIF

C LPCI (loop 1)
IF(I_EVENT_OLD(220).EQ.-2) THEN
  I_EVENT_OLD(220)=I_EVENT(220)
  IF(I_EVENT(220).EQ.1) THEN
    IEVNT(427)=1
    IEVNT(428)=0
  ELSEIF(I_EVENT(220).EQ.0) THEN
    IEVNT(427)=0
    IEVNT(428)=1
  ELSE
    IEVNT(427)=0
    IEVNT(428)=0
  ENDIF
ENDIF

C LPCI (loop 2)
IF(I_EVENT_OLD(139).EQ.-2) THEN
  I_EVENT_OLD(139)=I_EVENT(139)
  IF(I_EVENT(139).EQ.1) THEN
    IEVNT(429)=1
    IEVNT(430)=0
  ELSEIF(I_EVENT(139).EQ.0) THEN
    IEVNT(429)=0
    IEVNT(430)=1
  ELSE
    IEVNT(429)=0
    IEVNT(430)=0
  ENDIF
ENDIF

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 19

```

        ENDIF
    ENDIF
C LPCI Suction from CST (loop 2)
    IF(I_EVENT_OLD(152).EQ.-2) THEN
        I_EVENT_OLD(152)=I_EVENT(152)
        IF(I_EVENT(152).EQ.1) THEN
            IEVNT(419)=0
        ELSE
            IEVNT(419)=1
        ENDIF
    ENDIF
C LPCI Suction from Suppression Pool (loop 2)
    IF(I_EVENT_OLD(138).EQ.-2) THEN
        I_EVENT_OLD(138)=I_EVENT(138)
        IF(I_EVENT(138).EQ.1) THEN
            IEVNT(419)=1
        ELSE
            IEVNT(419)=0
        ENDIF
    ENDIF
C HPCI (loop 1)
    IF(I_EVENT_OLD(195).EQ.-2) THEN
        I_EVENT_OLD(195)=I_EVENT(195)
        IF(I_EVENT(195).EQ.1) THEN
            IEVNT(423)=1
            IEVNT(424)=0
        ELSEIF(I_EVENT(195).EQ.0) THEN
            IEVNT(423)=0
            IEVNT(424)=1
        ELSE
            IEVNT(423)=0
            IEVNT(424)=0
        ENDIF
    ENDIF
C HPCI (loop 2)
    IF(I_EVENT_OLD(140).EQ.-2) THEN
        I_EVENT_OLD(140)=I_EVENT(140)
        IF(I_EVENT(140).EQ.1) THEN
            IEVNT(425)=1
            IEVNT(426)=0
        ELSEIF(I_EVENT(140).EQ.0) THEN
            IEVNT(425)=0
            IEVNT(426)=1
        ELSE
            IEVNT(425)=0
            IEVNT(426)=0
        ENDIF
    ENDIF
C RHR (loop 1)
    IF(I_EVENT_OLD(214).EQ.-2) THEN
        I_EVENT_OLD(214)=I_EVENT(214)
        IF(I_EVENT(214).EQ.1) THEN
            IEVNT(433)=1
        ELSE
            IEVNT(433)=0
        ENDIF

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 20

```

ENDIF
C RHR (loop 2)
  IF(I_EVENT_OLD(142).EQ.-2) THEN
    I_EVENT_OLD(142)=I_EVENT(142)
    IF(I_EVENT(142).EQ.1) THEN
      IEVNT(434)=1
    ELSE
      IEVNT(434)=0
    ENDIF
  ENDIF
C RCIC (loop 1)
  IF(I_EVENT_OLD(225).EQ.-2) THEN
    I_EVENT_OLD(225)=I_EVENT(225)
    IF(I_EVENT(225).EQ.1) THEN
      IEVNT(435)=0
    ELSE
      IEVNT(435)=1
    ENDIF
  ENDIF
C RCIC (loop 2)
  IF(I_EVENT_OLD(143).EQ.-2) THEN
    I_EVENT_OLD(143)=I_EVENT(143)
    IF(I_EVENT(143).EQ.1) THEN
      IEVNT(436)=0
    ELSE
      IEVNT(436)=1
    ENDIF
  ENDIF
C RCIC HX
  IF(I_EVENT_OLD(34).EQ.-2) THEN
    I_EVENT_OLD(34)=I_EVENT(34)
    IF(I_EVENT(34).EQ.1) THEN
      IEVNT(437)=0
    ELSE
      IEVNT(437)=1
    ENDIF
  ENDIF
C Suppression Pool Cooling
  IF(I_EVENT_OLD(9).EQ.-2) THEN
    I_EVENT_OLD(9)=I_EVENT(9)
    IF(I_EVENT(9).EQ.1) THEN
      IEVNT(417)=1
    ELSE
      IEVNT(417)=0
    ENDIF
  ENDIF
C Heavy Water Cooling
  IF(I_EVENT_OLD(27).EQ.-2) THEN
    I_EVENT_OLD(27)=I_EVENT(27)
    IF(I_EVENT(27).EQ.1) THEN
      IEVNT(448)=0
    ELSE
      IEVNT(448)=1
    ENDIF
  ENDIF
C Thermal Shield Cooling

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 21

```

IF(I_EVENT_OLD(41).EQ.-2) THEN
  I_EVENT_OLD(41)=I_EVENT(41)
  IF(I_EVENT(41).EQ.1) THEN
    IEVNT(449)=0
  ELSE
    IEVNT(449)=1
  ENDIF
ENDIF
C Containment Air Cooling
IF(I_EVENT_OLD(67).EQ.-2) THEN
  I_EVENT_OLD(67)=I_EVENT(67)
  IF(I_EVENT(67).EQ.1) THEN
    IEVNT(439)=1
    IEVNT(440)=0
  ELSEIF(I_EVENT(67).EQ.0) THEN
    IEVNT(439)=0
    IEVNT(440)=1
  ELSE
    IEVNT(439)=0
    IEVNT(440)=0
  ENDIF
ENDIF
C Recirculation (loop 1)
IF(I_EVENT_OLD(49).EQ.-2) THEN
  I_EVENT_OLD(49)=I_EVENT(49)
  IF(I_EVENT(49).EQ.1) THEN
    IEVNT(413)=0
  ELSE
    IEVNT(413)=1
  ENDIF
ENDIF
C Recirculation (loop 2)
IF(I_EVENT_OLD(122).EQ.-2) THEN
  I_EVENT_OLD(122)=I_EVENT(122)
  IF(I_EVENT(122).EQ.1) THEN
    IEVNT(414)=0
  ELSE
    IEVNT(414)=1
  ENDIF
ENDIF
C If an SBO is in progress, allow for the accumulator block valve, MSIVs,
C turbine driven feedwater, and moderator dump valve to be turned on and off.
219  CONTINUE
C Accumulator block valve
IF(I_EVENT_OLD(126).EQ.-2) THEN
  I_EVENT_OLD(126)=I_EVENT(126)
  IF(I_EVENT(126).EQ.1) THEN
    IEVNT(421)=0
  ELSE
    IEVNT(421)=1
  ENDIF
ENDIF
C MSIVs
C  (Loop 1)
IF(I_EVENT_OLD(185).EQ.-2) THEN
  I_EVENT_OLD(185)=I_EVENT(185)

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 22

```

IF(I_EVENT(185).EQ.1) THEN
  IEVNT(407)=1
  IEVNT(405)=0
ELSEIF(I_EVENT(185).EQ.0) THEN
  IEVNT(407)=0
  IEVNT(405)=1
ELSE
  IEVNT(407)=0
  IEVNT(405)=0
ENDIF
ENDIF
C (Loop 2)
IF(I_EVENT_OLD(82).EQ.-2) THEN
  I_EVENT_OLD(82)=I_EVENT(82)
  IF(I_EVENT(82).EQ.1) THEN
    IEVNT(408)=1
    IEVNT(406)=0
  ELSEIF(I_EVENT(82).EQ.0) THEN
    IEVNT(408)=0
    IEVNT(406)=1
  ELSE
    IEVNT(408)=0
    IEVNT(406)=0
  ENDIF
ENDIF
C Main feedwater (turbine driven) (loop 1)
IF(I_EVENT_OLD(211).EQ.-2) THEN
  I_EVENT_OLD(211)=I_EVENT(211)
  IF(I_EVENT(211).EQ.1) THEN
    IEVNT(411)=0
  ELSE
    IEVNT(411)=1
  ENDIF
ENDIF
C Main feedwater (turbine driven) (loop 2)
IF(I_EVENT_OLD(310).EQ.-2) THEN
  I_EVENT_OLD(310)=I_EVENT(310)
  IF(I_EVENT(310).EQ.1) THEN
    IEVNT(412)=0
  ELSE
    IEVNT(412)=1
  ENDIF
ENDIF
C Moderator dump valve
IF(I_EVENT_OLD(21).EQ.-2) THEN
  I_EVENT_OLD(21)=I_EVENT(21)
  IF(I_EVENT(21).EQ.1) THEN
    IEVNT(446)=1
    IEVNT(447)=0
  ELSEIF(I_EVENT(21).EQ.0)THEN
    IEVNT(446)=0
    IEVNT(447)=1
  ELSE
    IEVNT(446)=0
    IEVNT(447)=0
  ENDIF

```

1997年 09月 24日 09時 25分 /users/pclund/maap/maapsrc/pic_api.f ページ 23

```
ENDIF
C Check intervention flag
220 IF(INTRPT.EQ.1) THEN
      TIOP=TIM
      INTRPT=0
      IFLAGP=99
      KEY=CHAR(0)
ENDIF
RETURN
END
CREV-MG END
```