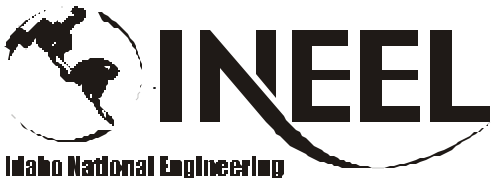
INEEL/CON-01-01145
PREPRINT



Pygmalion Users Manual

J. E. Fisher

09/05/2001 – 09/07/2001

2001 RELAP5 Users Seminar

Pygmalion Users Manual

J. E. Fisher

August 2001

Idaho National Engineering and Environmental Laboratory
Nuclear Systems Analysis Technologies Department
Idaho Falls, Idaho 83415

Prepared for the
U.S. Department of Energy
Under DOE Idaho Operations Office
Contract DE-AC07-99ID13727

PYGMALION Users Guide

SUMMARY

PYGMALION (or Pygi for short) is a RELAP5 utility program that updates the initial condition information within a RELAP5 input file. These initial conditions are obtained by performing steady-state calculations, the final results of which are written to a restart/plot (rstplt) file for subsequent use. Pygi accesses the rstplt file, obtains the final conditions for each component of the system model, and replaces the appropriate cards in the original input file with cards containing the new conditions. The new input file then accurately represents the hydrodynamic state of the problem as it was at the end of the steady-state initialization run.

INTRODUCTION

Accurate and self-consistent initial conditions are an essential part of simulating the transient response of a light water reactor coolant system or other thermal-hydraulic system. For a complex system, these initial conditions require more precision than can be easily obtained using a hand calculation. In the process of system model development, rough estimates of nominal conditions will initially be used in the input file. RELAP5 calculations will then be run with this input file to achieve an equilibrium condition representative of the desired initial state of the system for performing the transient simulations.

One method of proceeding to the transient simulations from this point is to restart directly from the rstplt file. The disadvantage of this method is that the steady-state rstplt file, which could be large, must be retained for the duration of the analyses. Also, separate transient input files, used to restart the problem from the equilibrium condition point, must be maintained. It is far more convenient to maintain a single input file, which includes initial conditions representative of the equilibrium state, and which can be used to simulate either continued steady-state operation or a transient simulation, as desired. In addition, updates to the RELAP5 code may introduce incompatibilities that prevent reading a rstplt file generated by a previous code version. These potential incompatibilities make the maintenance of historical records more difficult and create concerns if reproducibility of results is required for the project.

Therefore, a better method for proceeding to the transient analysis is to transfer the final conditions to the input file itself. Performing this step manually, however, is very labor intensive. The Pygi program was developed to automate this process.

DESCRIPTION

Pygi automates the process of specifying accurate initial conditions for a RELAP5 input file for the system being modeled. The analyst begins by supplying a rough estimate of the desired initial conditions for each component in the input file. The analyst must also include appropriate steady-state control systems in the input file that will cause the system to achieve an equilibrium condition. The analyst runs the calculation until equilibrium conditions are achieved, and saves the results on the rstplt file. The analyst then uses Pygi to replace the initial condition cards in the input file with the new data. Intermediate Pygi card replacement steps may be used as necessary to improve the steady-state conditions and accelerate convergence toward the equilibrium condition.

Specifically, for each control volume, Pygi writes pressure, liquid specific internal energy, vapor specific internal energy, and void fraction. If noncondensible is present it also writes noncondensible quality, and if boron is present it also writes boron concentration. For each junction, Pygi writes liquid and vapor velocities as the initial condition, and the mass flow rate is included as a comment on the same line. For pump and turbine components, Pygi writes the rotational speed, and for control variables it writes the calculated value.

The example below demonstrates the card replacement performed by Pygi. Component 100 is a RELAP5

pipe and consists of two volumes and one internal junction. Prior to card replacement, the initial conditions for volumes 100-01 and 100-02 are specified (on the 1001201 and 1001202 cards, respectively) as type 3 (pressure and temperature in equilibrium condition) and the approximate pressure and temperatures were entered by the analyst. Following card replacement, the initial conditions for the two volumes are specified as type 0. Pressure, liquid specific internal energy, vapor specific internal energy, and volume void fraction values from the appropriate plot record on the rstplt file have been entered into the fields of the 1001201 and 1001202 cards.

The junction initial conditions before card replacement are specified as type 1 on the 1001300 card and the liquid and vapor mass flow rates are entered on the 1001301 card. After card replacement, the initial condition type on card 1001300 has been changed to 0 and the liquid and vapor velocities from the plot record are entered into the 1001301 card. The mass flow rate from the plot record is also entered for user information following the comment character ('*').

Pipe Component Before Card Replacement	**Pipe Component After Card Replacement**
1000000 ihl pipe 1000001 2 1000101 15.420 1 1000102 13.761 2 1000301 3.339 1 1000302 4.362 2 1000601 0.0 2 1000801 0.0 2.558 1 1000802 0.0 2.417 2 1001001 00 2 1001101 0000 1 1001201 3 2245.0 589.0 0.0 0 0 1 1001202 3 2245.0 589.0 0.0 0 0 2 1001300 1 1001301 30464.55 0.0 0.0 1	1000000 ihl pipe 1000001 2 1000101 15.420 1 1000102 13.761 2 1000301 3.339 1 1000302 4.362 2 1000601 0.0 2 1000801 0.0 2.558 1 1000802 0.0 2.417 2 1001001 00 2 1001101 0000 1 1001201 0 2300.92 594.302 1048.703 0. 0. 1 1001202 0 2298.38 594.302 1048.878 0. 0. 2 1001300 0 1001301 50.2772 50.2772 0. 1 * 30282.35

Note that no action is taken for accumulator components. It is assumed that the initial state established for the accumulator does not change during the course of the steady-state initialization.

Also note that heat structure mesh point temperatures are not updated by Pygi. These mesh point temperatures are not written to the plot records of the rstplt file, and therefore are not available to Pygi. However, specifying these mesh point temperatures is not usually necessary, because they are normally generated by RELAP5 during the process of problem initialization. If for some reason the user needs or desires to input these mesh point temperatures, they can be easily obtained from the output data (outdta) file and appended onto the input deck.

Continuation Cards

Except for the multid component, continuation cards should not be used for the cards that specify the initial conditions for the components. This is because, with the noted exception, Pygi processes only cards that have sequence numbers. Cards beginning with '=', '*', '$' and '+' are copied directly to the new input file without processing. Therefore, if a continuation card is used in the card series that specifies the initial conditions for the component, Pygi will not perform the card replacement.

The following example illustrates the consequence of using a continuation card in a pipe component. This is the same RELAP5 pipe component as in the previous example, except that card 1001201 has been replaced by a continuation card. This is legal input for RELAP5 and it produces the same result as the previous example. However, Pygi does not recognize the continuation card as specifying an initial condition and copies it to the new input file without processing. It therefore does not make a card replacement for the

initial condition for volume 100-02. Also, Pygi has no mechanism for determining the whether a card replacement was made or was necessary for volume 100-02, so it does not generate an error message. This type of mistake can lead to transient simulations that do not start from the correct initial condition.

Pipe Component Before Card Replacement	**Pipe Component After Card Replacement**
1000000 ihl pipe 1000001 2 1000101 15.420 1 1000102 13.761 2 1000301 3.339 1 1000302 4.362 2 1000601 0.0 2 1000801 0.0 2.558 1 1000802 0.0 2.417 2 1001001 00 2 1001101 0000 1 1001201 3 2245.0 589.0 0.0 0 0 1 + 3 2245.0 589.0 0.0 0 0 2 1001300 1 1001301 30464.55 0.0 0.0 1	1000000 ihl pipe 1000001 2 1000101 15.420 1 1000102 13.761 2 1000301 3.339 1 1000302 4.362 2 1000601 0.0 2 1000801 0.0 2.558 1 1000802 0.0 2.417 2 1001001 00 2 1001101 0000 1 1001201 0 2300.92 594.302 1048.703 0. 0. 1 + 3 2245.0 589.0 0.0 0 0 2 1001300 0 1001301 50.2772 50.2772 0. 1 * 30282.35

This restriction should not present difficulties for the user, because continuation cards should not be necessary in the cards containing the initial conditions. As noted, coding is present in the logic that writes the new conditions for the multid component that allows the use of continuation cards.

Pygi also requires information from certain other input cards to perform its operations. For example, for pipe, branch, mtpljun, or multid components, the important card is the CCC0001 card, which contains the number of volumes for a pipe, the number of junctions for branch and mtpljun components, and the x, y, and z dimensions and the geometry flag for a multid component. Also important for a mtpljun component is the CCC0NNM card, which contains the Junction Limit for a set. Therefore, cards that are important to Pygi must not be continued using a continuation card (i.e., a '+' in column 1), but instead should be continued using the next card number in the series. In these examples, if Pygi misses information it needs it should generate an error message so the user can correct the original input deck.

Replacement Cards

Pygi checks for duplicated Component Identifier Cards (CCC0000). If one is found, an error message is placed in the new input file,

```
'$pygmsg:  component number duplicated '
'$pygmsg:  old i c card retained in deck'
```

Pygi will nevertheless attempt to process the component data and insert the initial conditions according to the latest CCC0000 card that was encountered. To avoid the occurrence of this message, the user should comment the redundant or replaced CCC0000 card.

Syntax

The command syntax for Pygi is as follows:

```
pygi [ -r rstplt ] [ -t time ] [ -e frctn ] [ -mBCIO ]
```

The default name of the restart/plot file is rstplt. Pygi reads the old input file from standard input and writes

the new input file to standard output. Pygi will select the first plot record with a time greater than or equal to the value specified by the '-t' parameter. If the '-t' parameter is not specified, Pygi will select the last plot record found on the rstplt file. Unless otherwise specified, the old and new input decks conform to the RELAP5/MOD2 or later format. Optionally, Pygi will read and write in the RELAP5/MOD1 format.

Optional keyletter arguments to Pygi are:

```
-r rstplt  The pathname to the rstplt file which was written by  RELAP5
           ( default is "rstplt" ).

-t time    The time of the plot record from which to extract the steady-
           state data ( default is the last plot record on the rstplt
           file ).

-e frctn   The maximum normalized truncation error for real numbers
           written by Pygi ( default is 1.e-6).  Smaller values will
           result in more significant digits in the real numbers written
           into the new deck.

-B         By default, Pygi ignores records which begin with a blank in
           column 1.  The –B option causes Pygi to read such records.

-C         Do not change control variable initial conditions.

-I         Old input file is to be read in RELAP5/MOD1 format.

-O         New input file is to be written in RELAP5/MOD1 format.

-m         Provide a brief description of Pygi.
```

Output

Pygi writes a short narrative to the screen of the workstation, reporting the status of its operation. The first section reports the numbers of each component type found on the old input file, and the second section reports the numbers of each component type contained on the plot records of the rstplt file. Normally, the numbers for control volumes and junctions differ between the two sections. This is because time dependent volumes and junctions are included in the count of the rstplt file plot records, whereas these components are not included in the old input file component count. The numbers for control variables may also differ, because control variables of type 'constant' are not included in the old input file component count.

The third section reports the plot record used as the source of the new initial conditions, the number of card replacements attempted and completed, and the number of error messages written to the new input file. If Pygi reports that error messages have been written, the new input file should be scanned for the occurrence of '$pygmsg', which indicates the location of the error message.

Example

The following command would cause Pygi to create a RELAP5/MOD2 input file named 'NewDeck' from the file named 'OldDeck' using initial condition data from the plot record at 237 seconds on the rstplt file named 'SSrstplt2'.

```
pygi -r SSrstplt2 -t 237. < OldDeck > NewDeck
```