PROYECTO INTEGRADOR
CARRERA DE INGENIERÍA NUCLEAR

ACOPLE NEUTRÓNICO TERMOHIDRÁULICO CITVAP – RELAP

RUBERT MARTÍN PARDO

Director
Dr. Eduardo Villarino

Co-director
Ing. Pablo Camusso

Miembros del jurado

Eduardo Villarino
Pablo Camusso
Alexis Weir
Sergio Papadakis

Departamento de Ingeniería Nuclear, INVAP S.E.

Instituto Balseiro
Comisión Nacional de Energía Atómica
Universidad Nacional de Cuyo

Junio, 2017

Rubert Martín Pardo: *Acople Neutrónico Termohidráulico* `CITVAP-RELAP`

LUGAR:
División de Ingeniería Nuclear, INVAP S.E.
San Carlos de Bariloche, Río Negro, Argentina
PERÍODO DE TIEMPO:
Agosto 2016– Junio 2017

Ohana significa Familia.
Familia significa "Nadie queda atrás, ni olvidado".

— Lilo & Stitch

Dedicado a mi Familia,
los que están y los que ya no.

RESUMEN

Para extender las capacidades de diseño de reactores nucleares, se desarrolla un acople entre el código de cálculo de núcleo CITVAP y el código termohidráulico de planta RELAP . Esto permite el cálculo de coeficientes de potencia y márgenes termohidráulicos a fenómenos críticos y el estudio de crecimiento de la capa de óxido en las vainas de los elementos combustibles.

Se desarrolla una metodología de acople entre los dos códigos para estados estacionarios del núcleo. Para ello se desarrollan los programas relap2citvap y citvap2relap que permiten el flujo de información entre los archivos auxiliares de los códigos. Para validar el correcto funcionamiento del acople, se calcula el coeficiente de realimentación por potencia del OPAL y se compara con correspondiente valor experimental, obteniéndose resultados satisfactorios.

Se inicia un estudio de factibilidad de cálculo de transitorios con CITVAP como primera etapa para el desarrollo de un acople dinámico entre RELAP y CITVAP . Para ello se proponen mecanismos de insertar los términos temporales de la ecuación de difusión a través de una fuente externa equivalente. Se desarrolla el programa transient que permite el cálculo de transitorios junto con CITVAP dentro de un ciclo de cálculo. Se verifica el correcto funcionamiento del ciclo con modelos nucleares sencillos, comparando los resultados numéricos con las correspondientes soluciones analíticas.

Finalmente, se proponen mejoras y trabajos futuros para dar continuidad al desarrollo del acople neutrónico termohidráulico entre CITVAP y RELAP , buscando incorporar la cinética neutrónica en tres dimensiones al diseño de reactores nucleares.

ÍNDICE GENERAL

Acrónimos 1

1 INTRODUCCIÓN 1
- 1.1 Motivación . . . 1
- 1.2 Dos enfoques para realizar el acople . . . 2
- 1.3 Acople estacionario y acople dinámico . . . 2
- 1.4 Objetivos . . . 2
- 1.5 Práctica Profesional Supervisada . . . 3

I ACOPLE ESTACIONARIO 5

2 ACOPLE NEUTRÓNICO TERMOHIDRÁULICO 7
- 2.1 Diagrama de flujo del acople estacionario . . . 7
- 2.2 Interpolación de secciones eficaces macroscópicas con `CITVAP` . . . 9

3 DESARROLLO DE PROGRAMAS PARA EL ACOPLE ESTACIONARIO 13
- 3.1 Objetivo de los programas `relap2citvap` y `citvap2relap` . . . 13
- 3.2 Requisitos y limitaciones del acople neutrónico termohidráulico con `relap2citvap` y `citvap2relap` . . . 13
  - 3.2.1 Modelos requeridos de `RELAP` y `CITVAP` . . . 13
  - 3.2.2 Limitaciones en el cálculo del acople . . . 14
- 3.3 Archivos externos y programas que intervienen en el acople . . . 14
- 3.4 Flujo del ciclo de cálculo . . . 16

4 VERIFICACIÓN Y VALIDACIÓN DEL ACOPLE NEUTRÓNICO TERMOHIDRÁULICO 19
- 4.1 El reactor OPAL . . . 19
  - 4.1.1 Descripción general del reactor . . . 19
  - 4.1.2 Modelo del núcleo de OPAL para `CITVAP` . . . 20
  - 4.1.3 Modelo del núcleo de OPAL para `RELAP` . . . 21
- 4.2 Verificaciones iniciales . . . 22
- 4.3 Variación de condiciones iniciales del acople . . . 22
- 4.4 Determinación del coeficiente de realimentación por potencia del OPAL . . . 24
  - 4.4.1 Determinación experimental del coeficiente de potencia . . . 25
  - 4.4.2 Cálculo de α_P a partir del acople neutrónico termohidráulico . . . 26

II INCORPORACIÓN DE TRANSITORIOS A CITVAP 29

5 DESCRIPCIÓN DEL MÉTODO DE CÁLCULO EMPLEADO POR CITVAP-CITATION 31
- 5.1 Naturaleza del problema resuelto por CITVAP-CITATION . . . 31
- 5.2 Método de solución . . . 31
- 5.3 Fuente externa . . . 32
- 5.4 Limitaciones. Estados cuasiestacionarios y estados transitorios . . . 33

6 INCORPORACIÓN DE CÁLCULO DE TRANSITORIOS A CITVAP 35
- 6.1 Ecuación de difusión no estacionaria . . . 35

6.2 Condiciones iniciales y discretización temporal 36
6.3 Mecanismos para incorporar transitorios 36
7 DESARROLLO DE UN PROGRAMA PARA EL CÁLCULO DE TRANSITORIOS 39
7.1 Objetivo del programa Transient 39
7.2 Requisitos y limitaciones del cálculo de transitorios con Transient 39
7.3 Archivos externos y programas que intervienen en el ciclo de cálculo 39
7.4 Flujo del ciclo de cálculo 40
8 VALIDACIÓN DEL CICLO DE CÁLCULO DE TRANSITORIOS 45
8.1 Modelo de núcleo empleado 45
8.2 Solución analítica para un cambio instantáneo de $\nu\Sigma_f$ 46
8.3 Escalón positivo de reactividad 47
8.3.1 Determinación del paso de tiempo Δt óptimo para el cálculo 47
8.3.2 Cálculo de flujo neutrónico y la concentración de precursores para $\Delta t = 0{,}02s$. Prompt Jump. 48
8.3.3 SCRAM 49
8.4 Conclusiones de la Validación 49

Trabajo Futuro y Conclusiones 51
9 MEJORAS Y PROPUESTAS FUTURAS 53
9.1 Propuestas para el desarrollo del Acople Estacionario (Parte I) . 53
9.2 Propuestas para el desarrollo del Cálculo de Transitorios (PARTE II) 53
9.2.1 Modificación del programa CITVAP-CITATION y su preprocesador precit 54
9.2.2 Mejoras y extensiones al método de solución de transitorios 54
9.3 Desarrollo de un acople dinámico entre RELAP y CITVAP 55
10 CONCLUSIONES 57
10.1 Acople Estacionario 57
10.2 Estudio de transitorios 57
10.3 Etapas posteriores 58

Apéndices 59
A LÍNEA DE CÁLCULO NEUTRÓNICO DE INVAP S.E. 61
A.1 Cálculo de Celda y Cálculo de Núcleo 61
A.2 Descripción General de la línea de Cálculo 61
A.3 Código de Celda CONDOR 62
A.4 Código de Núcleo CITVAP 63
B ACTIVIDADES RELACIONADAS CON LA PRÁCTICA PROFESIONAL SUPERVISADA 65
C DESCRIPCIÓN DE LOS CÓDIGOS FUENTES DE relap2citvap Y citvap2relap 67
C.1 Código fuente de citvap2relap 67
C.1.1 Definición de variables e inclusión de bibliotecas 67
C.1.2 Lectura de la línea de comandos 67

C.1.3 Llamada de rutinas . . . 68
C.1.4 Rutina `readRelations(*)` . . . 68
C.1.5 Rutina `readCDB(*)` . . . 68
C.1.6 Rutina `writeInput(*)` . . . 69
C.2 Código fuente de `relap2citvap` . . . 70
C.2.1 Definición de variables e inclusión de bibliotecas . . . 70
C.2.2 Lectura de la línea de comandos . . . 71
C.2.3 Llamada de rutinas . . . 71
C.2.4 Rutina `readRelations(*)` . . . 71
C.2.5 Rutina `readRestart (*)` . . . 71
C.2.6 Rutina `writeCDB (*)` . . . 73
C.2.7 Rutina eraseData() . . . 74
D DESCRIPCIÓN DE ARCHIVOS PARA EL ACOPLE NEUTRÓNICO TERMOHIDRÁULICO 85
D.1 Descripción del archivo `Nucleo.CII` . . . 85
D.2 Descripción del archivo `LPRR.I` . . . 85
D.3 Descripción del archivo `relaciones` . . . 85
E RESOLUCIÓN DE PROBLEMAS CON FUENTE EXTERNA CON `citvap-citation` 89
E.1 Inclusión de la sección 026 en el archivo de entrada a `CITVAP` . . . 89
E.2 Especificación de la fuente externa en el archivo `source.dat` . . . 90
E.2.1 Tarjeta 1: 026. Formato `(I3)` . . . 90
E.2.2 Tarjeta 2: Opciones. Formato `(2I3)` . . . 90
E.2.3 Tarjeta 3: Espectro energético de la fuente externa. Formato `(6E12.0)` . . . 91
E.2.4 Tarjeta 6: Intensidad de fuente externa por zona.Formato `6(I3,E9.0)` . . . 91
E.2.5 Final de la entrada de datos . . . 92
F DESCRIPCIÓN DEL CÓDIGO FUENTE DE `transient` 93
F.1 Declaración de variables . . . 93
F.2 Lectura de la línea de comandos . . . 93
F.3 Llamada de rutinas auxiliares . . . 94
F.3.1 rutina `readNuclearData()` . . . 94
F.3.2 Rutina `readCDB()` . . . 95
F.3.3 Rutina `readPrecursors()` . . . 95
F.3.4 Rutina `CalculateC()` . . . 96
F.3.5 Rutina `calculateQ()` . . . 96
F.3.6 Rutina `outputResults()` . . . 96
F.4 Observaciones finales . . . 97

BIBLIOGRAFÍA 103

ÍNDICE DE FIGURAS

Figura 1 Diagrama de flujo para el ciclo de acople estacionario. A la derecha de cada proceso se muestran ejemplos de modelos empleados para cada etapa de cálculo (Celda,Núcleo y Termohidráulico). 8

Figura 2 Obtención de la sección eficaz Σ_x en (T_{cx},ρ_{rx}, T_{rx}) por interpolación lineal. 10

Figura 3 Diagrama de flujo para el ciclo de acople estacionario con los programas `relap2citvap` y `citvap2relap`. También aparecen las relaciones de escritura/lectura con los archivos auxiliares. 16

Figura 4 El reactor nuclear OPAL 19

Figura 5 Modelos de RELAP y CITVAP para OPAL 21

Figura 6 Reactividad introducida vs. Potencia del reactor para las *Mediciones Experimentales* y para el *Cálculo Numérico con el Acople Neutrónico Termohidráulico* en el OPAL. 26

Figura 7 Diagrama de flujo para el ciclo de cálculo de transitorios. 43

Figura 8 Comparación del cálculo analítico de $\phi(t)$ con cálculos numéricos de distintos Δt 48

Figura 9 Resultados numérico y analítico de ϕ y C para una inserción positiva de reactividad. $\Delta t = 0{,}02s$. 49

Figura 10 Resultados numérico y analítico de ϕ y C para SCRAM. $\Delta t = 0{,}05s$. 50

Figura 11 Diagrama de flujo para el ciclo de cálculo de transitorios. 62

Figura 12 Ejemplo de archivo `Nucleo.CII` con las líneas de control necesarias para llevar a cabo el acople *desde* `CITVAP` 87

Figura 13 Parte del archivo `LPRR.I` 88

Figura 14 Parte del archivo `relaciones`. 88

Figura 15 Ejemplo de archivo `Nucleo.cii` preparado para resolver un problema de Fuente externa. 89

Figura 16 Ejemplo de archivo `source.dat` con información necesaria para un problema de Fuente externa con `CITVAP-CITATION`. 90

Figura 17 Ejemplo de archivo `NuclearData.dat` con para el caso de un problema de una zona, a tres grupos de energía y seis grupos de precursores. 94

ÍNDICE DE TABLAS

Cuadro 1 Archivos externos empleados en el ciclo de cálculo del acople neutrónico-termohidráulico con `relap2citvap` y `citvap2relap` 15

Cuadro 2 Casos iniciales de Modelo de Núcleo de OPAL para verificación del acople. 23

Cuadro 3 Valores finales de reactividad luego acople neutrónico termohidráulico para distintos Casos de parámetros iniciales. 23

Cuadro 4 Valores medios de los distintos parámetros termohidráulicos para los casos III y IV luego de realizado el acople. También se muestran los errores cuadráticos absolutos y relativos al restar las distribuciones de ambos casos. . . . 24

Cuadro 5 Configuraciones críticas a distintas potencias para la determinación del coeficiente de potencia del OPAL 25

Cuadro 6 Valores de coeficiente de realimentación por potencia para Cálculo experimental ($\alpha_P^{exp.}$) y Cálculo numérico con acople con `RELAP` ($\alpha_P^{núm.}$). También se muestra el resultado de un acople `CITVAP` – `Termic` ($\alpha_P^{Termic.}$) 26

Cuadro 7 Programas que intervienen en el ciclo de cálculo de transitorios. 40

Cuadro 8 Archivos externos empleados en el ciclo de cálculo de transitorios con `Transient` 41

Cuadro 9 Errores absoluto ($\varepsilon = |\phi_n - \phi(t_n)|$) y relativo ($\varepsilon_n = \varepsilon/\phi(t_n)$) entre solución analítica $\phi(t_n)$ y cálculos numéricos ϕ_n para distintos Δt a tiempo $t_n = 150s$. 48

ÍNDICE DE CÓDIGOS

Código 1 Ejemplo de llamada de `citvap2relap` desde la línea de comandos de Windows 68
Código 2 Ejemplo de llamada de `citvap2relap` desde la línea de comandos de Windows 71
Código 3 Listado del archivo `citvap2relap.f95` 75
Código 4 Listado del archivo `relap2citvap.f95` 76
Código 5 Listado del archivo `utilities.f95` 77
Código 6 Ejemplo de llamada del programa `Transient` desde la línea de comandos de Windows 93
Código 7 Listado del archivo `Transient.f95` 98

1 INTRODUCCIÓN

Los avances de los últimas décadas en sistemas computacionales han permitido el desarrollo de métodos innovativos para el cálculo de reactores. Ahora es posible incluir información de ingeniería adicional a la empleada tradicionalmente.

La línea de cálculo de INVAP es usada actualmente para un amplio rango de aplicaciones en paralelo con cálculos termohidráulicos de distintos códigos o modelos. De esta manera se obtiene una evaluación más precisa del comportamiento del núcleo en distintas circunstancias. Una descripción de dicha línea de cálculo se encuentra en el Apéndice A.

El cálculo de núcleo con CITVAP [1] (perteneciente a la línea de cálculo de INVAP) incorpora el uso de distintos códigos termohidráulicos para obtener un correcto acople entre las variables termohidráulicas de distintos modelos de reactores y las variables neutrónicas que intervienen en los mismos.

Además de realizar la correcta realimentación termohidráulica este acople permite el estudio del crecimiento de la capa de óxido en las vainas de los combustibles de reactores MTR, y la determinación de parámetros como el coeficiente de realimentación por potencia de un reactor y los márgenes termohidráulicos a fenómenos críticos.

En este proyecto se estudia la factibilidad de incorporación del código termohidráulico de cálculo de planta RELAP (véase [2]) a un acople con CITVAP y se dan los primeros pasos en el desarrollo de dicho acople.

1.1 MOTIVACIÓN

El estudio de transitorios en un reactor para operaciones rutinarias o situaciones accidentales es sumamente importante para el licenciamiento, puesta en marcha y operación del mismo. Se impone entonces la necesidad de incorporar estos análisis al cálculo neutrónico detallado que ofrece cada línea de cálculo.

En particular el programa RELAP permite el análisis termohidráulico de un reactor nuclear y sus transitorios, pero en general emplea un modelo cinética puntual para estudiar el comportamiento neutrónico del núcleo. Incorporar la capacidad de hacer un análisis más detallado del flujo neutrónico, teniendo en cuenta las tres dimensiones espaciales y los múltiples factores que influyen en su evolución, permitiría obtener resultados más precisos que los correspondientes a un modelo de cinética puntual.

En definitiva, un acople entre el código de núcleo CITVAP con todo el detalle espacial del flujo neutrónico y el código termohidráulico RELAP con las múltiples facilidades que brinda para simulación de planta terminaría dando un panorama completo y preciso de la multifísica inherente al comportamiento de un reactor nuclear.

1.2 DOS ENFOQUES PARA REALIZAR EL ACOPLE

El acople propuesto puede ser llevado a cabo desde dos enfoques distintos:

1. Por un lado, durante la simulación de planta con RELAP se puede emplear realimentación de la neutrónica brindada por CITVAP a través de una distribución en tres dimensiones de la densidad de potencia. En este caso, el acople será *solicitado* por RELAP a medida que necesite actualizar la distribución de densidad de potencia debido a las fisiones.

2. Por otro lado, durante el cálculo del núcleo y la simulación de fenómenos en el mismo con CITVAP se puede requerir a través del cálculo con RELAP una realimentación de los parámetros termohidráulicos para actualizar los valores de de las secciones eficaces. En este caso, el acople es *solicitado* por CITVAP .

1.3 ACOPLE ESTACIONARIO Y ACOPLE DINÁMICO

Si el acople entre los parámetros neutrónicos y los termohidráulicos es realizado para un estado estacionario del núcleo lo denominamos *acople estacionario*.

Si, en cambio, se realizan simulaciones de transitorios en la planta, tales como paradas de bomba en el primario o extracción de barras de control, el acople que aplica en este caso lo denominamos *acople dinámico*. Aquí, las variables neutrónicas y termohidráulicas variarán en el tiempo debido a las perturbaciones introducidas.

1.4 OBJETIVOS

Con este proyecto se pretende dar inicio a la metodología para acoplar los códigos RELAP y CITVAP en cualquiera de los enfoques mencionados en Sección 1.2.

Se buscará avanzar lo más posible en dicho acoplamiento, teniendo como objetivos mínimos lograr el acople estacionario y evaluar la factibilidad de modelar transitorios en CITVAP minimizando las modificaciones de los códigos fuentes.

En la primera parte se expone el desarrollo del acople estacionario entre los dos códigos. El acople desarrollado es verificado y validado contrastando con resultados experimentales.

En la segunda parte se desarrolla un análisis de factibilidad para incorporar el cálculo de transitorios a CITVAP . Este es el primer paso para arribar a un acople dinámico con RELAP . Las siguientes etapas quedan a modo de propuestas futuras. El estudio de transitorios también es sometido a verificaciones mediante modelos sencillos de núcleo.

1.5 PRÁCTICA PROFESIONAL SUPERVISADA

Este proyecto integrador es parte de la Práctica Profesional Supervisada de la carrera de Ingeniería Nuclear en el Instituto Balseiro. Un desglose de las actividades realizadas durante la práctica podrá ser encontrado en el Apéndice B.

Parte I

ACOPLE ESTACIONARIO

En esta parte se describe el desarrollo de un ciclo de cálculo para acoplar `RELAP` y `CITVAP` para un estado estacionario del núcleo. Se desarrollan los programas `relap2citvap` y `citvap2relap` que dan sostén al flujo de información dentro del ciclo. El acople neutrónico termohidráulico es verificado con algunas experiencias de cálculo. Además se modela el núcleo del reactor OPAL y se calcula numéricamente el coeficiente de reactividad por potencia para el mismo. El valor obtenido es contrastado con el correspondiente resultado experimental, obteniéndose una buena congruencia con el mismo. Se concluye entonces que el acople desarrollado resulta satisfactorio y se proponen mejoras y desarrollos posteriores.

2 ACOPLE NEUTRÓNICO TERMOHIDRÁULICO

CITVAP permite realizar cálculo de núcleo por el método de difusión. Con este cálculo se termina conociendo el flujo neutrónico en el núcleo. A partir del flujo se determina la distribución de densidad de potencia en la geometría del reactor.

Para llevar a cabo el cálculo se requiere conocer las secciones eficaces macroscópicas de los distintos materiales que componen al núcleo del reactor (elementos combustibles, refrigerante/moderador, reflectores, etc.). Ahora bien, los valores de las secciones eficaces dependen de varios *parámetros de estado* entre los cuales se pueden destacar la temperatura y densidad en el refrigerante/moderador (en lo adelante T_{ref} y ρ_{ref}) y la temperatura en el combustible (en lo adelante T_{comb}).

Las distribuciones de T_{ref} , ρ_{ref} y T_{comb} pueden ser obtenidas con ayuda de un código termohidráulico (en este trabajo se usará RELAP) en el cual se modele la planta con la distribución de densidad de potencia originada por el reactor nuclear. Esta distribución de potencias, calculada con CITVAP dependerá a su vez de las distribuciones de T_{ref} , ρ_{ref} y T_{comb} .

En definitiva, se requerirá encontrar un conjunto de valores de las variables de estado y de distribución de densidad de potencia en el núcleo que acoplados por las relaciones de dependencia entre ellos y que a su vez caractericen al modelo del reactor nuclear propuesto.

Para arribar al acople requerido se deberá realizar un proceso iterativo de cálculos de núcleo con CITVAP y cálculos termohidráulicos con RELAP hasta que se cumpla una condición de convergencia para alguno de los parámetros de estado o en la distribución de densidad de potencia.

Existen varias metodologías para llevar a cabo dicho acople, variando según el interés final del diseño y el cálculo. Puede querer modelarse un estado estacionario de planta, o bien un estado transitorio debido a perturbaciones introducidas en el reactor. Las distintas metodologías posibles son analizadas en este trabajo en mayor o menor profundidad.

2.1 DIAGRAMA DE FLUJO DEL ACOPLE ESTACIONARIO

El acople estacionario se refiere a la determinación de la distribución de Potencia y de los parámetros de estado T_{ref} , ρ_{ref} y T_{comb} para una configuración *fija* del núcleo de un reactor al que se le puede determinar un Reactor Crítico Asociado.

La metodología seguida para llevar a cabo el acople estacionario[1] se encuentra representada en el diagrama de flujo de la Figura 1. A continuación se describen las distintas etapas del acople.

1 Véase el artículo [3]

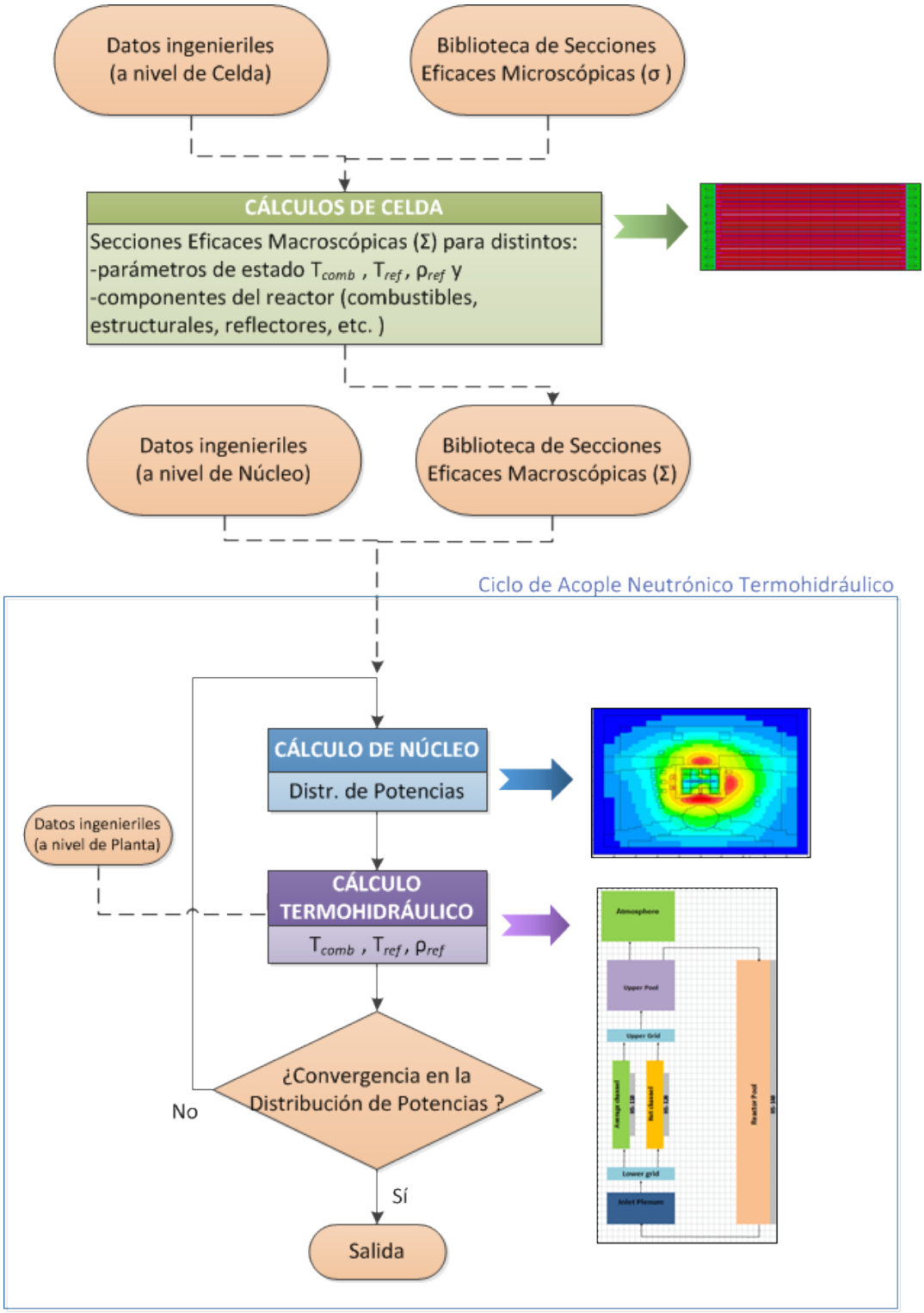


Figura 1: Diagrama de flujo para el ciclo de acople estacionario. A la derecha de cada proceso se muestran ejemplos de modelos empleados para cada etapa de cálculo (Celda,Núcleo y Termohidráulico).

1. El usuario ingresa los datos de ingeniería nivel de celda (Geometría de la celda, Materiales, Etapa de quemado, etc.) y la Biblioteca de Secciones

Eficaces Microscópicas[2] al archivo de entrada del código de Cálculo de Celda. En este proyecto se emplea el programa CONDOR (véase [4]) perteneciente a la línea de cálculo de INVAP S.E.

2. Se realizan los cálculos de las celdas de los distintos componentes del reactor, para las secciones eficaces macroscópicas para distintos conjuntos de valores característicos de los parámetros de estado (T_{ref} , ρ_{ref} y T_{comb}).

3. Se realiza un cálculo de núcleo con CITVAP [1] a partir de la biblioteca de secciones eficaces macroscópicas y datos ingenieriles concernientes al núcleo (distribución geométrica, potencia total, etc.). Con este cálculo se obtiene una distribución inicial de potencias en el núcleo.

4. Se realiza un cálculo termohidráulico con RELAP [2] a partir de la distribución de densidad de potencia obtenida en la etapa anterior . En este caso también se especifican datos de ingeniería a nivel de la planta nuclear como un todo. Del cálculo se obtiene el conjunto (T_{ref} , ρ_{ref} y T_{comb}) correspondiente a la distribución de potencias ingresada.

5. Con los parámetros de estado calculados de la termohidráulica, CITVAP calcula nuevas secciones eficaces macroscópicas, a partir de una interpolación de los datos recibidos del cálculo de celda (véase la Sección 2.2). Con estos valores se repite el cálculo de núcleo obteniéndose una nueva distribución de densidad de potencia.

6. Si la nueva distribución de densidad de potencia es *cercana* a la distribución de densidad de potencia calculada en el paso anterior se detiene el ciclo, ya que se ha arribado a la convergencia. En este contexto *cercana* significa que la distancia entre los vectores de distribución de densidad de potencia por zonas del núcleo es menor a un parámetro de error ε prefijado por el usuario.

7. Si no ha ocurrido la convergencia se procede a repetir el cálculo de los parámetros de estado y se da continuidad al ciclo de acople.

2.2 INTERPOLACIÓN DE SECCIONES EFICACES MACROSCÓPICAS CON citvap

El cálculo de núcleo requiere de las secciones eficaces macroscópicas de cada una de las zonas en que está dividido el modelo del núcleo. Estas dependen de los parámetros (T_{ref} , ρ_{ref} y T_{comb}) que irán variando para cada ciclo del acople neutrónico-termohidráulico.

Para obtener las secciones eficaces adecuadas, CITVAP realiza una interpolación de los valores de secciones eficaces macroscópicas obtenidos con cálculo de celda (CONDOR) para valores específicos predeterminados de los parámetros de estado (T_{ref} , ρ_{ref} y T_{comb}).

La Figura 2 muestra el espacio de fases de las variables de estado (T_{ref} , ρ_{ref} y T_{comb}). Para cada punto (T_{cx},ρ_{rx}, T_{rx}) de este espacio, existe un conjunto de valores Σ_x de secciones eficaces.

2 En este proyecto se empleó la biblioteca ESIN2001 a 69 grupos, que se basa en la Biblioteca WIMS.

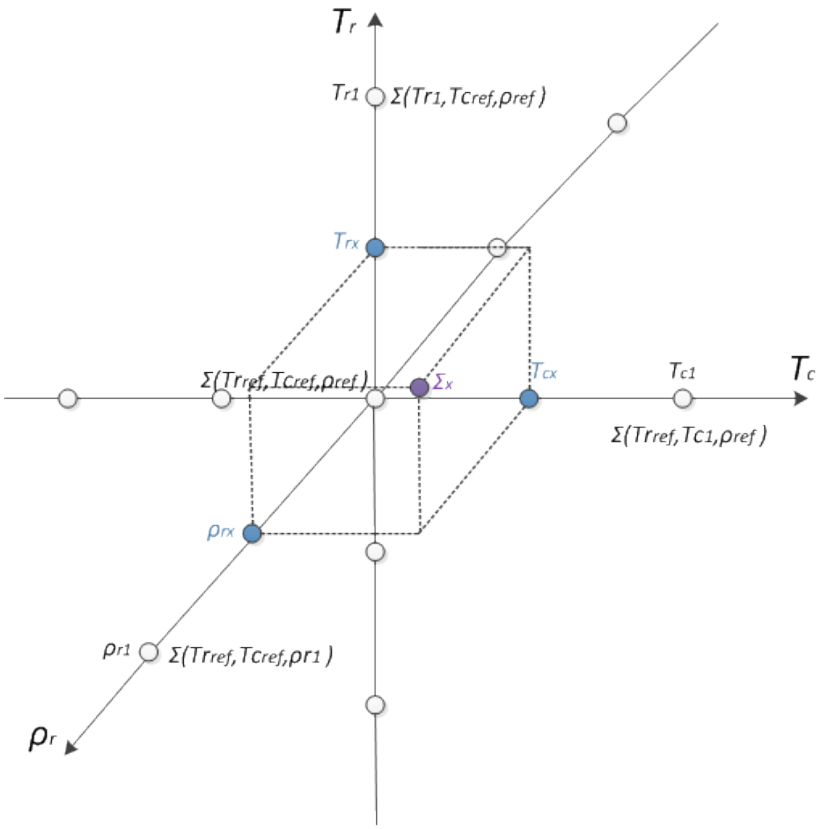


Figura 2: Obtención de la sección eficaz Σ_x en (T_{cx},ρ_{rx}, T_{rx}) por interpolación lineal.

CONDOR pasa a CITVAP algunos conjuntos de secciones eficaces, definidos sólo para puntos específicos del espacio de fase. Por tanto, para determinar los valores de secciones eficaces para un conjunto arbitrario (T_{cx},ρ_{rx}, T_{rx}), CITVAP deberá hacer una interpolación con los datos de secciones eficaces obtenidos de CONDOR.

La interpolación se hace realizando un ajuste lineal entre secciones eficaces correspondientes a puntos adyacentes en cada uno de los ejes del espacio de estados. Así, el conjunto de secciones eficaces Σ_x se calcula:

$$\Sigma_x = \Sigma_{ref} + \Delta T_{cx} * \frac{\partial \Sigma}{\partial T_c} + \Delta T_{rx} * \frac{\partial \Sigma}{\partial T_r} + \Delta \rho_{rx} * \frac{\partial \Sigma}{\partial \rho_r}, \tag{1}$$

donde las secciones eficaces Σ_x se calculan a partir de las secciones eficaces Σ_{ref} en un punto de referencia del espacio de estados y de las derivadas parciales $\frac{\partial \Sigma}{\partial T_c}$,$\frac{\partial \Sigma}{\partial T_r}$ y $\frac{\partial \Sigma}{\partial \rho_r}$. Estas derivadas son calculadas por aproximación de cocientes incrementales entre las secciones eficaces que se tienen de la biblioteca más cercanas en el espacio de fases al punto (T_{cx},ρ_{rx}, T_{rx}) de interés donde se desea determinar Σ.

Por ejemplo, para el caso del punto indicado por Σ_x en la Figura 2 y definiendo por simplicidad

$$\Sigma_{ref} = \Sigma(Tc_{ref}, Tr_{ref}, \rho r_{ref})$$
$$\Sigma_{Tc1} = \Sigma(Tc_1, Tr_{ref}, \rho r_{ref})$$
$$\Sigma_{Tr1} = \Sigma(Tc_{ref}, Tr_1, \rho r_{ref}) \text{ y}$$
$$\Sigma_{\rho r1} = \Sigma(Tc_{ref}, Tr_{ref}, \rho r_1),$$

las correspondientes derivadas en la Ecuación 1 se determinan según:

$$\frac{\partial \Sigma}{\partial T_c} \approx \frac{\Sigma_{Tc1} - \Sigma_{ref}}{T_{c1} - T_{c,ref}},$$
$$\frac{\partial \Sigma}{\partial T_r} \approx \frac{\Sigma_{Tr1} - \Sigma_{ref}}{T_{r1} - T_{r,ref}},$$
$$\frac{\partial \Sigma}{\partial \rho_r} \approx \frac{\Sigma_{\rho r1} - \Sigma_{ref}}{\rho_{r1} - \rho_{r,ref}}.$$

Estas derivadas se basan en los valores de las secciones eficaces calculados por CONDOR para puntos específicos del espacio de estados.

Una descripción detallada de la determinación de secciones eficaces macroscópicas con CITVAP puede ser encontrada en [3].

3

DESARROLLO DE PROGRAMAS PARA EL ACOPLE ESTACIONARIO

Para llevar a cabo el acople entre RELAP y CITVAP según lo expuesto en el Capítulo 2 se necesitó desarrollar programas auxiliares que hicieran la transferencia de información necesaria entre los dos programas de cálculo. En este capítulo se describe el desarrollo de los programas relap2citvap y citvap2relap, que llevan a cabo esta función.

Una descripción de los códigos fuentes de Fortran 95 empleados en el desarrollo de relap2citvap y citvap2relap puede ser encontrada en el Apéndice C.

También se describe en este capítulo la integración de relap2citvap y citvap2relap en la metodología de acople neutrónico termohidráulico propuesta a través del flujo del ciclo de cálculo.

3.1 OBJETIVO DE LOS PROGRAMAS relap2citvap Y citvap2relap

Los programas relap2citvap y citvap2relap sirven de puente entre el código de cálculo de núcleo CITVAP y el de termohidráulica RELAP . Con ayuda de estos programas, se busca la transferencia adecuada de información entre ambos códigos y modelos.

Así, con citvap2relap la información de distribución de densidad de potencia que se obtiene luego de una corrida con CITVAP , es acondicionada para ser leída por el correspondiente modelo de RELAP . También, los datos termohidráulicos de densidades y temperaturas que se obtienen con una corrida de RELAP son acondicionados por relap2citvap para ser leídos por CITVAP .

De esta forma, se garantiza el flujo del ciclo de cálculo y la eventual convergencia de los parámetros neutrónicos acoplados a los termohidráulicos.

3.2 REQUISITOS Y LIMITACIONES DEL ACOPLE NEUTRÓNICO TERMOHIDRÁULICO CON relap2citvap Y citvap2relap

3.2.1 *Modelos requeridos de RELAP y CITVAP*

Para llevar a cabo el acople neutrónico termohidráulico se debe contar inicialmente con dos modelos del reactor en estudio: uno implementado en CITVAP y el otro en RELAP .

El modelo de CITVAP normalmente contendrá información sobre todo el núcleo del reactor. Esta información (secciones eficaces, tiempos de quemado, etc.) vendrá especificada por zonas (véase [5]), cada una de las cuales tiene un número identificatorio. Estos números son asignados secuencialmente para las zonas en un mismo plano axial XY (normalmente el plano perpendicular al flujo de refrigerante) y luego se continúa la numeración con el plano axial inmediata-

mente inferior. Así se distribuyen los números de zonas por plano axial, de arriba hacia abajo.

El modelo de RELAP para el núcleo deberá contener un conjunto de canales por los que se hace fluir el refrigerante. Cada canal será simulado con un pipe[1], dividido en zonas axiales. Así, la información termohidráulica intercambiada (densidad de potencia, densidades, temperaturas) será dada por zonas axiales.

Para transferir calor al refrigerante, se emplea la herramienta de heat structure de RELAP (véase [2]). A cada pipe de canal se le asigna una heat structure que transferirá el calor al refrigerante. Las heat structures también se encontrarán subdivididas en zonas axiales que serán asignadas a las zonas axiales de los pipes. Normalmente esta asignación se hará uno a uno, pero esto no necesariamente debe de ser así. Para referenciar cada zona axial de los pipes y heat structures se cuenta también con números identificatorios únicos.

Para establecer la potencia con que las zonas axiales de las heat structures calentarán las correspondientes zonas axiales de los pipes, RELAP brinda varios mecanismos. De estos, se empleará el de las variables de control para asignar la potencia de cada zona axial de las heat structures. Una variable de control será corresponderá a cada zona axial de heat structure, estableciendo con la misma la potencia que dicha zona entregará. Los valores de potencia son asignados en la zona correspondiente a la definición de variables de control dentro del archivo de entrada de RELAP .

3.2.2 *Limitaciones en el cálculo del acople*

El usuario deberá establecer correctamente las correspondencias entre las zonas del modelo de RELAP y las correspondientes al modelo de CITVAP . El modelo de CITVAP contiene zonas que no son de interés[2] para el estudio termohidráulico (estructurales, reflectores, etc.) y otras que representan precisamente los canales de refrigerante con el elemento combustible en su interior. Estas últimas son las que deben de ser apareadas con las correspondientes zonas axiales de los pipes y las heat structures del modelo de RELAP . Para llevar a cabo esto se emplea un archivo de relaciones, que se describe en la próxima sección.

Aún no se implementó un método de verificación de que la asignación realizada sea compatible físicamente, por lo que el usuario deberá cuidar de la correcta asignación de las zonas para obtener resultados correctos con el acople.

3.3 ARCHIVOS EXTERNOS Y PROGRAMAS QUE INTERVIENEN EN EL ACOPLE

Los archivos auxiliares empleados en el acople son listados en Tabla 1. Los nombres de los archivos son completamente arbitrarios, no así las extensiones.

1 Un pipe es un componente hidrodinámico de RELAP . Para mayor descripción véase [2].

2 En esta etapa del proyecto no se tendrá en cuenta el intercambio de calor con materiales distintos del refrigerante, sin embargo un estudio futuro de la termohidráulica con los mismos se recomienda en próximas etapas.

Los nombres usados se corresponden a los empleados durante el desarrollo de este trabajo.

NOMBRE	DESCRIPCIÓN
`Nucleo.CII`	Archivo de entrada para `CITVAP`. Contiene el modelo del núcleo del reactor nuclear dividido en *zonas espaciales* y las especificaciones necesarias para el cálculo del flujo neutrónico y la distribución de las densidad de potencia en el núcleo. Además, puede contener las órdenes necesarias para llevar a cabo el acople desde `CITVAP` (Una descripción puede ser encontrada en el Apéndice D).
`Nucleo.CDB`	Base de datos `CITVAP`. Contiene información de los resultados del cálculo realizado con `CITVAP`, en particular las densidades de potencia por zonas del modelo que luego serán pasadas a `RELAP` . También es en este archivo de donde se leen o escriben los valores de los parámetros de estado (T_{ref} , ρ_{ref} y T_{comb}) para el uso de las correspondientes secciones eficaces en `CITVAP` .
`LPRR.I`	Archivo de entrada para `RELAP` . Contiene el modelo termohidráulico del núcleo a través de `pipes`,`heat structures` y variables de control, como se describió en la sección anterior. Con las potencias especificadas para cada zona axial en este archivo, `RELAP` calculará la distribución de temperaturas y densidades para el caudal impuesto. Un ejemplo de archivo puede ser encontrado en Apéndice D.
`LPRR.R`	Archivo de Restart para `RELAP` . Este archivo binario contiene información de los resultados de la corrida de `RELAP` que pueden ser empleados para el análisis de la termohidráulica o como punto de partida para corridas posteriores. De este archivo se extraen los valores de los parámetros de estado (T_{ref} , ρ_{ref} y T_{comb}) que deberán ser pasados a la base de datos de `CITVAP` (`Nucleo.CDB`) para dar continuidad al cálculo del acople.
`relaciones`	Archivo auxiliar de texto (sin extensión) empleado por los programas `relap2citvap` y `citvap2relap` para realizar el traspaso de la información de cada zona axial del modelo `CITVAP` a la correspondiente zona del modelo de `RELAP` y relaciona cada zona de `pipe`, `heat structures` y variables de control dentro del modelo de `RELAP` . Este archivo será pasado por el usuario y será esencial para establecer la equivalencia de modelos en los dos programas de cálculo `RELAP` y `CITVAP` .El formato en que debe ser escrito se describe en Apéndice D

Tabla 1: Archivos externos empleados en el ciclo de cálculo del acople neutrónico-termohidráulico con `relap2citvap` y `citvap2relap`

3.4 FLUJO DEL CICLO DE CÁLCULO

Una vez descritos los programas y archivos que intervienen en el acople, se procede a describir el ciclo de cálculo para llevar a cabo el mismo.

En la Figura 3 se muestra un diagrama de flujo indicando las distintas etapas de cálculo por la que se debe pasar para realizar el acople.

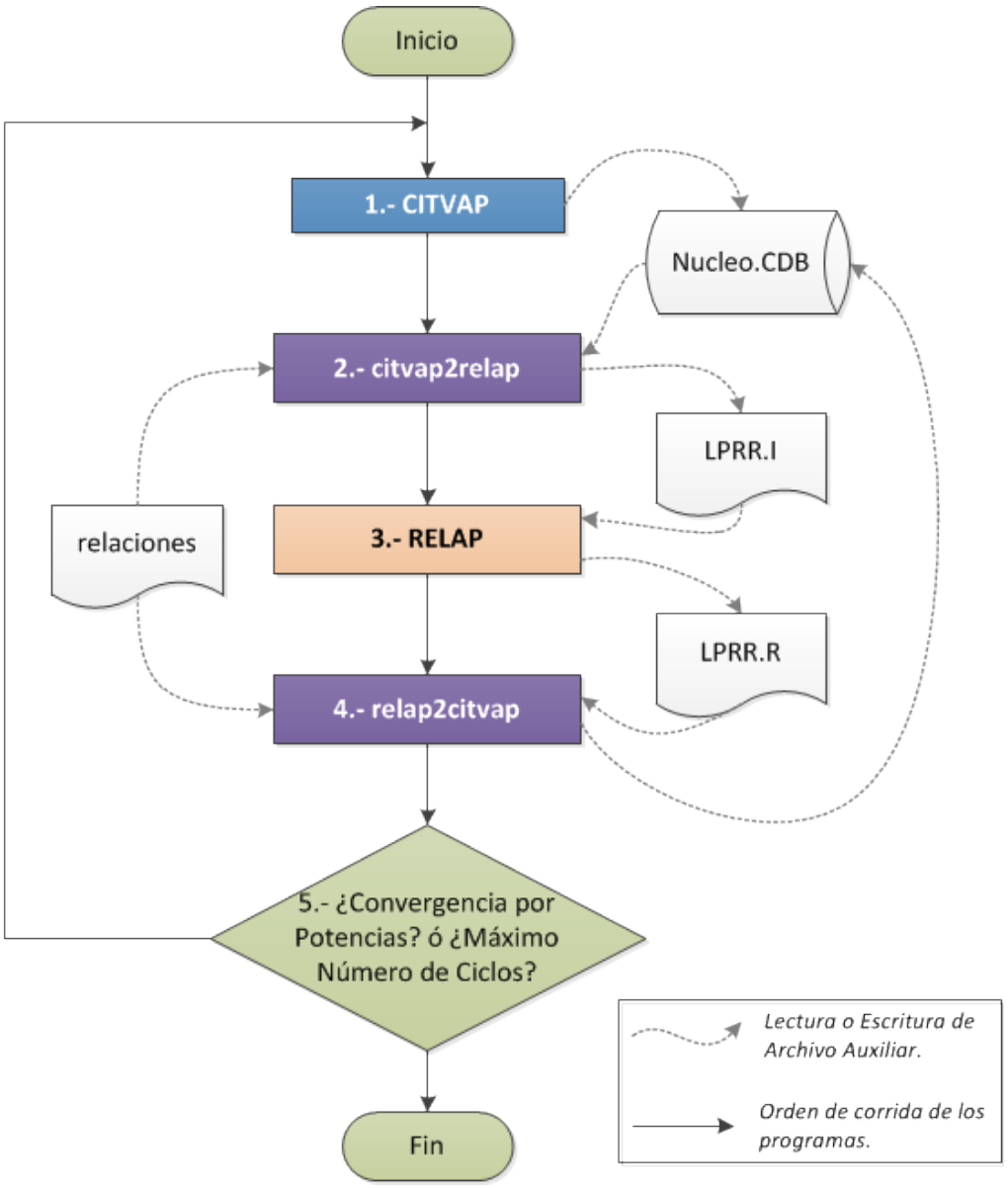


Figura 3: Diagrama de flujo para el ciclo de acople estacionario con los programas `relap2citvap` y `citvap2relap`. También aparecen las relaciones de escritura/lectura con los archivos auxiliares.

A continuación se describe cada una de las etapas, basándose en la numeración de los bloques de la Figura 3.

1. Partiendo del modelo de núcleo según lo descrito en la Sección 3.2, con parámetros de estado iniciales y potencia total establecida, se realiza una

corrida de CITVAP , con la cual en la base de datos Nucleo.cdb se obtiene, entre otras cosas, la distribución inicial de densidades de potencia.

2. Se corre el programa citvap2relap. Éste leerá las densidades de potencias y los volúmenes de las zonas del modelo de CITVAP que aparecen listadas en el archivo relaciones (que corresponden a los canales combustibles). Las potencias para cada cada zona serán calculadas y pasadas a las variables de control del modelo de relap mediante la correspondiente actualización del archivo de entrada LPRR.I.

3. Se corre RELAP. Con este programa, se hace una simulación temporal hasta alcanzar un estado estacionario. Como resultado de esta simulación, se obtienen los correspondientes valores de (T_{ref} , ρ_{ref} y T_{comb}) para cada una de las zonas axiales de los pipes y son almacenados en el archivo de restart LPRR.R.

4. Se corre el programa relap2citvap. El mismo lee los parámetros de estado (T_{ref} , ρ_{ref} y T_{comb}) del archivo LPRR.R para los distintos instantes de tiempo simulados por RELAP . Los valores correspondientes al último instante de tiempo, son almacenados en la base de datos Nucleo.CDB en los lugares correspondientes a las zonas de los canales de refrigerante. Las lecturas y escrituras en los archivos mencionados las hace basándose en los números identificatorios que se encuentran listados en relaciones, lo que le permite hacer el intercambio de información entre zonas de RELAP y de CITVAP equivalentes.

5. Se realiza una nueva corrida de CITVAP . Esta vez, debido a que los parámetros de estado almacenados en la base de datos Nucleo.cdb han cambiado respecto a los valores iniciales, las secciones eficaces serán distintas, calculadas acorde a dichos parámetros según lo descrito en la Sección 2.2. Del cálculo con CITVAP se obtendrá una nueva distribución de densidad de potencia. La distribución de densidad de potencia será comparada con la obtenida en el paso anterior. El módulo del vector diferencia entre las dos distribuciones de potencias es comparado con un ε predefinido. Si resulta menor se acepta que el cálculo convergió, de lo contrario se vuelve al ítem 1 de esta lista.

Si el acople se realiza desde un enfoque neutrónico (véase la Sección 1.2) tiene sentido emplear un parámetro neutrónico como el flujo o la reactividad para determinar convergencia. La reactividad del sistema resulta un parámetro integrado y por tanto no es conveniente para determinar convergencia. Por otro lado emplear el flujo neutrónico implicaría analizar convergencia simultánea de las distribuciones de flujo para los distintos grupos, cosa que en la etapa actual del proyecto no se ha implementado.

Finalmente, se decidió elegir la distribución de densidades de potencias para analizar convergencia.

El flujo del ciclo de acople neutrónico termohidráulico puede ser realizado por un programa externo(un script del sistema operativo empleado) o dentro del propio CITVAP . Durante el proyecto se realizó con instrucciones del CITVAP .

Esto quiere decir que en el mismo archivo `Nucleo.CII`, además de las especificaciones del modelo de núcleo, se incluían las llamadas a los distintos programas para el acople y la verificación de convergencia en densidad de potencia.

4

VERIFICACIÓN Y VALIDACIÓN DEL ACOPLE NEUTRÓNICO TERMOHIDRÁULICO

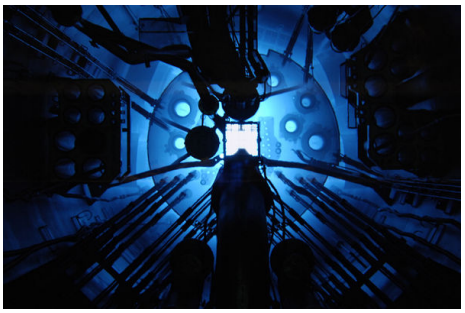
En este capítulo se mostrarán los resultados obtenidos para algunas experiencias realizadas con la metodología de acople neutrónico-termohidráulico descrita previamente. Estas experiencias están dirigidas a comprobar el correcto funcionamiento del acople. En todos los cálculos se emplearon modelos de `CITVAP` y `RELAP` para el reactor OPAL.

4.1 EL REACTOR OPAL

4.1.1 *Descripción general del reactor*

El reactor de investigación Australiano de Agua Liviana en Pileta Abierta (OPAL: Open Pool Light Water Reactor)[1] se ubica en Lucas Heights, Australia. Es un reactor de 20MW_{th} diseñado para múltiples usos. Fue construido y comisionado por `INVAP S.E.` entre los años 2000 y 2006. Es operado por la Organización Australiana de Ciencia y Tecnología Nuclear (ANSTO: Australian Nuclear Science and Technology Organization). La Figura 4 muestra fotos del edificio y el núcleo del reactor.

(a) Edificio del reactor.



(b) Núcleo del reactor en marcha.

Figura 4: El reactor nuclear OPAL.

En noviembre del 2016, OPAL alcanzó los 10 años de operación continua, convirtiéndose en uno de los reactores de su tipo de más confiabilidad y disponibilidad en el mundo, con un récord invicto de estar completamente operacional por 307 días al año.

Varios parámetros neutrónicos fueron medidos durante la puesta en marcha y fueron verificados contra los modelos desarrollados en la etapa de diseño. Además se realizó modelado avanzado de algunas características. Actualmente

1 www.ansto.gov.au/ResearchHub/OurInfrastructure/ACNS/Facilities/OPALReactor/index.htm

se cuenta con datos experimentales de alta calidad, lo cual permite reproducir pruebas realizadas más de diez años antes.

El reactor consiste en un núcleo compacto de 16 combustibles tipo MTR, de Siliciuro de Uranio LEU [2] disperso. El reactor es refrigerado y moderado por agua liviana. Se emplea además agua pesada como reflector, contenida en un tanque alrededor del núcleo.

Los sistemas de apagado del reactor están constituidos por:

- Un Primer Sistema de Parada de actuación rápida, compuesto por cinco Barras de Control de Hafnio, cuatro de ellas con forma de placa y una central con forma de cruz. La última también es usada para regulación.
- Un Segundo Sistema de Parada (independiente, diverso y redundante) consistente en el vaciado parcial del tanque de reflector de agua pesada.

Varias posiciones de irradiación se encuentran ubicadas en el Tanque del Reflector, incluyendo:

- una Fuente Fría de Neutrones con dos haces fríos,
- una Fuente de Neutrones Térmicos con dos haces,
- una región reservada para una futura Fuente Caliente de Neutrones,
- 17 tubos de irradiación vertical con 5 lugares de irradiación cada uno destinados a producción industrial de radioisótopos (tales como ^{192}Ir, ^{99}Mo y ^{131}I),
- 19 posiciones de irradiación neumáticas y
- 6 dispositivos para dopaje de silicio por transmutación neutrónica.

4.1.2 *Modelo del núcleo de OPAL para `CITVAP`*

Se empleó un modelo de núcleo con 58 canales, cada uno con 20 zonas axiales. Cada uno de los 16 elementos combustibles emplea 3 canales, mientras que las cinco barras de control ocupan 2 canales cada una.

En la Figura 5a se muestra un corte axial del modelo del núcleo empleado para `CITVAP` .

Los 3 canales de los elementos combustibles corresponden, dos de ellos a los marcos con venenos quemables y el restante al material combustible propiamente dicho. Este último canal es el que será de interés para acoplar con los correspondientes componente hidrodinámico y estructura de calor del modelo de `RELAP` . Estas zonas aparecen coloreadas de azul celeste en la Figura 5a.

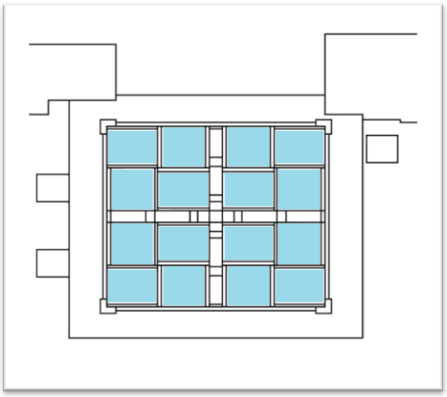
Para cambiar potencia y moverse entre distintas configuraciones críticas, se cuenta con la manipulación de las cinco barras de control en el núcleo. Esto se logra a través del comando `MOVE CONTROL ROD` de `CITVAP` .

2 wgt de ^{235}U $< 20\,\%$

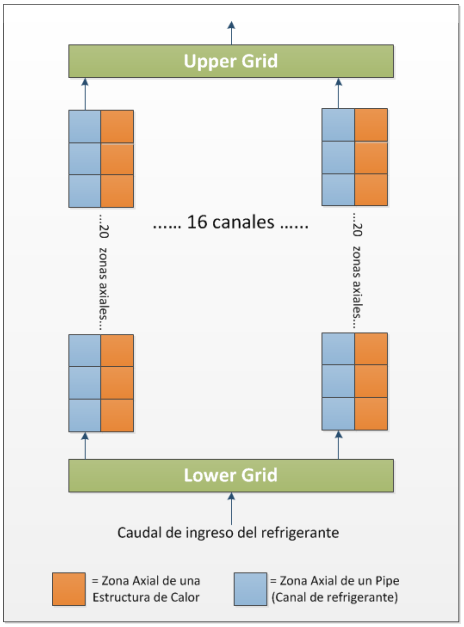
4.1.3 *Modelo del núcleo de OPAL para RELAP*

Para la verificación del acople se decidió no emplear un modelo completo de planta en RELAP para el OPAL, sino solamente de la parte del núcleo. De acuerdo a esta hipótesis, el caudal de entrada del refrigerante al núcleo se impone como condición de frontera al modelo de RELAP .

En la Figura 5b se esquematiza el modelo empleado para RELAP a través de las relaciones entre los distintos componentes que forman parte del modelo. Se muestran en azul los pipes (canales de refrigerante) y en naranja las estructuras de calor que interactúan con los correspondientes canales.



(a) Parte del núcleo del modelo de CITVAP para el OPAL.



(b) Modelo de RELAP para la termohidráulica del OPAL.

Figura 5: Modelos de RELAP y CITVAP para el OPAL.

En el modelo de RELAP se fija la temperatura de entrada al núcleo con un *Time Dependent Volume (tmdpvol)* y el caudal con un *Time Dependent Junction (tmdpjun)*, ambos conectados a un *Single Volume (snglvol)* al que se denomina Lower Grid en la Figura 5b.

Luego, a través de un *Multiple Junction mtpljun*[3] se distribuye uniformemente el caudal a los 16 canales de refrigerante. Cada uno de estos canales intercambia calor con una estructura de calor. Tantos los canales (pipes) como las estructuras de calor están divididos en 20 zonas axiales. Cada zona axial de cada estructura de calor contiene un parámetro que es la potencia transferida a la correspondiente zona axial del canal de refrigerante. Esta potencia puede ser controlada por Variables de Control, las cuales a su vez son modificadas durante el proceso de acople neutrónico termohidráulico como se explicó previamente.

3 tmdpvol,tmdpjun,snglvol y mtpljun son elementos de RELAP . Véase [2].

Finalmente el caudal total después del paso por el núcleo es recibido por otro *Single Volume (`snglvol`)* denominado Upper Grid.

El archivo de `relaciones` necesario para el acople contendrá tabulados los números de identificación de las zonas axiales de los pipes, de las estructuras de calor,de las variables de control en el modelo de `RELAP` y de las correspondientes zonas en el modelo de `CITVAP` .

4.2 VERIFICACIONES INICIALES

El desarrollo de la metodología para el acople neutrónico termohidráulico incluyó varias etapas de verificación de los códigos y programas componentes:

- Se corroboró que la transferencia de información entre la base de datos de `CITVAP` y los archivos de entrada y salida de `RELAP` se hacía de la manera planeada. Para ello hubo de identificarse las zonas del modelo de `RELAP` correspondientes a los canales refrigerantes y determinar las correspondientes zonas de `pipes`,`heat structures` y variables de control de potencia en `RELAP` .
- Se realizó además un estudio de los tiempos de simulación con `RELAP` para que los parámetros de estado se correspondieran con las potencias introducidas luego de una simulación hasta el estacionario.

4.3 VARIACIÓN DE CONDICIONES INICIALES DEL ACOPLE

Una de las primeras pruebas que se pueden realizar con el acople es comprobar que el estado final del núcleo luego de la convergencia, es independiente de los valores iniciales de los parámetros de estado (T_{comb},ρ_{ref}, T_{ref}) usados en el modelo de `CITVAP` antes de empezar el ciclo de acople.

Se plantearon cuatro casos distintos de modelos de núcleos con igual configuración de barras e igual potencia inicial pero con distintos valores iniciales de dos de los parámetros de estados (T_{comb},ρ_{ref}). El que estos parámetros sean distintos hace que para cada modelo exista una reactividad inicial distinta.

En la Tabla 2 se muestran los valores de T_{comb},ρ_{ref} y la correspondiente reactividad inicial de cada uno de los Casos calculados. También se muestran la potencia empleada y la configuración de las 5 barras de control del reactor.

Debe aclararse que las temperaturas T_{comb} y T_{ref} especificadas para cada Caso son impuestas para *todas las zonas* del modelo, o sea, de manera homogénea en todo el reactor. Esto es sólo para establecer un modelo inicial de prueba, que luego del acople, convergerá a la distribución correcta de temperaturas y densidad de potencia.

Luego de realizar el acople, se espera que tantos los parámetros de estado, como la distribución de densidad de potencias y la propia reactividad converjan a valores similares (dentro de un rango de error porcentual ϵ), independientemente del caso inicial empleado en el acople.

En la Tabla 3 se muestran los resultados de las reactividades finales luego de haber realizado el acople para cada uno de los Casos planteados.

CASO	$T_{COMB}[°C]$	$T_{REF}[°C]$	REACTIVIDAD INICIAL [pcm]
I	80.0	30.0	339.9
II	100.0	20.0	973.9
III	120.0	40.0	66.2
IV	140.0	15.0	987.0
Potencia =	0.0106 MWth		
Pos. de barras de control [% de extracción]	BC1=85.2	BC2=30.1	BC3=29.9
	BC4=84.9	BC5=24.6	

Tabla 2: Casos iniciales de Modelo de Núcleo de OPAL para verificación del acople.

Estos resultados dependen claramente del valor de ϵ empleado en el acople neutrónico termohidráulico, siendo ϵ el valor porcentual de error en la densidad de potencia permitido para aceptar convergencia y terminar el acople. En el caso de los resultados de la Tabla 3, se empleó un $\epsilon = 0{,}05\,\%$.

CASO	REACTIVIDAD FINAL [pcm]
I	498.6
II	501.2
III	482.8
IV	502.4

Tabla 3: Valores finales de reactividad luego acople neutrónico termohidráulico para distintos Casos de parámetros iniciales.

Puede observarse que aunque los estados iniciales de cada Caso eran bien diferentes entre sí (Tabla 2), todos convergen a estados finales con reactividades muy similares. La diferencia entre las reactividades finales más alejadas es de 20pcm, que representa un 4 % del valor medio de las mismas.

En lo adelante se compararán los resultados de los Casos que más distaron entre sí según la reactividad final luego del acople. Según la Tabla 3 estos son los casos III y IV.

En la Tabla 4 se muestran los valores medios de las distribuciones de Temperaturas de Combustible $\overline{T}_{comb}$, Temperaturas de Refrigerante $\overline{T}_{ref}$, Densidades de refrigerante $\overline{\rho}_{ref}$ y Densidades de Potencia $\overline{densP}$ para los casos III y IV. Recordemos que la potencia a la que se realizaron los cálculos es muy baja, con lo cual es esperable que las temperaturas de combustible y refrigerante sean similares entre sí.

Se observa que los valores medios de las distintas distribuciones difieren entre sí a partir de la quinta cifra decimal, lo cual es un buen indicador de que las distribuciones son cercanas.

Casos	$\overline{T}_{COMB}$ [°C]	$\overline{T}_{REF}$ [°C]	$\overline{\rho}_{REF}$ [$\frac{kg}{cm^3}$]	$\overline{densP}$ [$\frac{W}{cm^3}$]
III	26,8692817	26,8557311	996,623339	0,187483996
IV	26,8692913	26,8557301	996,623352	0,187483937
Error cuadrático Absoluto	$2{,}52 \times 10^{-4}$	$8{,}07 \times 10^{-5}$	$8{,}63 \times 10^{-5}$	$1{,}66 \times 10^{-3}$
Error cuadrático Relativo	9×10^{-4} %	3×10^{-4} %	9×10^{-6} %	9×10^{-1} %

Tabla 4: Valores medios de los distintos parámetros termohidráulicos para los casos III y IV luego de realizado el acople. También se muestran los errores cuadráticos absolutos y relativos al restar las distribuciones de ambos casos.

Podemos obtener todavía mejor estimación de la cercanía entre las distribuciones calculando los errores cuadráticos entre las mismas. El error cuadrático de una distribución, por ejemplo de Temperaturas de Combustible se calculará según:

$$\Delta T_{comb} = \sqrt{\sum_{r} \left(T^{III}_{comb}(r) - T^{IV}_{comb}(r)\right)^2} \tag{2}$$

donde $T^{III}_{comb}(r)$ es la temperatura del combustible para cada zona r del modelo, para el resultado final del cálculo con el caso III. Así mismo, $T^{IV}_{comb}(r)$ corresponde al caso IV. La sumatoria en la Ecuación 2 se hace para todas las zonas del modelo.

En la Tabla 4 se muestran los errores cuadráticos de las distintas distribuciones para los parámetros estudiados. También se muestran los correspondientes errores relativos en puntos porcentuales, que son el cociente entre los errores cuadráticos y la media de los valores $\overline{T}^{III}_{comb}$ y $\overline{T}^{IV}_{comb}$.

Los errores relativos son todos menores al 1 % con lo cual se concluye que las distribuciones son bien cercanas entre sí y el acople fue satisfactorio ante pruebas con distintas condiciones iniciales.

4.4 DETERMINACIÓN DEL COEFICIENTE DE REALIMENTACIÓN POR POTENCIA DEL OPAL

Anteriormente se mencionó entre las principales aplicaciones prácticas del acople neutrónico termohidráulico la determinación de los coeficientes por realimentación de potencia del reactor.

El coeficiente de potencia α_P describe la variación en reactividad debido a un cambio en la potencia del reactor:

$$\alpha_P \equiv \frac{d\rho}{dP} .$$

Este parámetro está directamente relacionado a la estabilidad inherente del reactor, y debe de ser mantenido negativo para todas las condiciones de operación concebibles.

El acople neutrónico termohidráulico aplicado al cálculo de un reactor nuclear tiene en cuenta precisamente la influencia de la potencia en los parámetros neutrónicos del reactor y por tanto en la reactividad de cada configuración estudiada. Es por esto que el acople representa una herramienta efectiva para calcular el coeficiente de potencia.

El segundo conjunto de cálculos propuestos para validar la metodología de acople propuesta previamente fue la determinación de α_P para el OPAL. Los resultados fueron contrastados con las correspondientes mediciones experimentales realizadas con el reactor y que se describen a continuación.

4.4.1 *Determinación experimental del coeficiente de potencia*

En el reporte [6] se describe la experiencia realizada con el reactor OPAL a fin de determinar experimentalmente el coeficiente de potencia del reactor.

Partiendo de una configuración crítica inicial a Potencia muy baja, el reactor fue hecho pasar por tres configuraciones críticas con distintos niveles de potencia extrayendo la barra de regulación (Barra-5). La reactividad introducida por la extracción paulatina de la barra era compensada con la reactividad negativa introducida por el incremento de la potencia. Así, conociendo la potencia de cada una de las etapas por las que pasó el núcleo, y la reactividad introducida de una etapa a la siguiente (a través de una calibración por reactividad de la Barra-5), se realizó un ajuste lineal de los puntos Potencia vs. Reactividad Introducida respecto a la configuración inicial. La pendiente negativa del ajuste corresponde al valor experimental de α_P

En la tabla 5 se muestran los resultados obtenidos durante la experiencia descrita. Los valores de potencia fueron determinados con detectores calibrados. En la última columna aparecen listadas las reactividades introducidas por los cambio en potencia, que fueron calculadas a partir de la calibración de la barra de regulación BC-5.

Conf. Crítica	Pos. de las Barras de Control [%de extracción]					Potencia [MW]	Reactividad introducida por aumento de Potencia [pcm]
	BC-1	BC-2	BC-3	BC-4	BC-5		
Inicial	85,0	40,0	40,0	85,0	37,7	$1{,}06 \times 10^{-2}$	0
Paso 1	85,0	40,0	40,0	85,0	38,8	6,06	−50,5
Paso 2	85,0	40,0	40,0	85,0	39,8	13,7	−95,8
Paso 3	85,0	40,0	40,0	85,0	40,4	18,0	−127,0

Tabla 5: Configuraciones críticas a distintas potencias para la determinación del coeficiente de potencia del OPAL

En la Figura 6 se muestra el gráfico de los puntos experimentales Reactividad y Potencia para cada una de las configuraciones estudiadas. También aparece el ajuste lineal de estos cuatro puntos. La pendiente de este ajuste, calculada a partir del método de mínimos cuadrados, se corresponde al coeficiente de re-

alimentación por potencia experimental. El valor $\alpha_P^{exp.}$ correspondiente aparece en la Tabla 6.

Se comprueba que el coeficiente es negativo, tal como se demanda para el diseño de cualquier reactor nuclear.

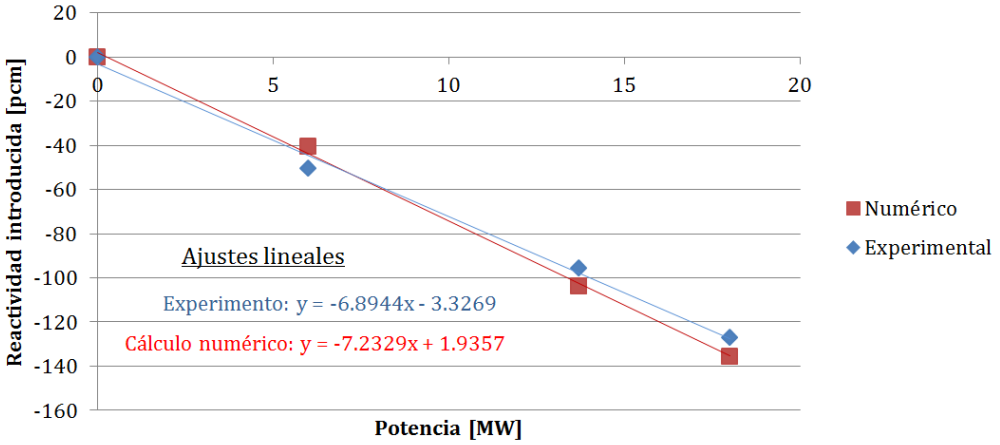


Figura 6: Reactividad introducida vs. Potencia del reactor para las *Mediciones Experimentales* y para el *Cálculo Numérico con el Acople Neutrónico Termohidráulico* en el OPAL.

COEFICIENTE	VALOR $[\frac{pcm}{MW}]$
$\alpha_P^{exp.}$	$(-6{,}89 \pm 0{,}34)$
$\alpha_P^{núm.}$	$(-7{,}23 \pm 0{,}21)$
$\alpha_P^{Termic.}$	$(-7{,}17)$

Tabla 6: Valores de coeficiente de realimentación por potencia para Cálculo experimental ($\alpha_P^{exp.}$) y Cálculo numérico con acople con `RELAP` ($\alpha_P^{núm.}$). También se muestra el resultado de un acople `CITVAP` – `Termic` ($\alpha_P^{Termic.}$)

4.4.2 *Cálculo de α_P a partir del acople neutrónico termohidráulico*

En el cálculo experimental, se estudiaron varias *configuraciones críticas* y se calculó la reactividad introducida por el cambio de potencia a través de una calibración previa de la barra de regulación BC-5 que fue la empleada para compensar la reactividad.

Para calcular numéricamente el coeficiente de potencia, la metodología propuesta es ligeramente distinta. En este caso,se preparan varios modelos del núcleo del reactor *todos con la misma configuración inicial de barras*(véase la Tabla 5),pero con distintas potencias (correspondientes a las potencias de las mediciones experimentales).

Para cada modelo, se realiza el acople neutrónico termohidráulico. Como todos tienen la misma configuración de barras, pero distintas potencias, es de esperar que luego del acople la reactividad de cada modelo sea distinta (gra-

cias a la realimentación por potencia). O sea, cada modelo, a diferencia de de sus contrapartes experimentales de la Tabla 5, representará una configuración subcrítica del reactor. Con los valores de reactividad obtenidos numéricamente y los correspondientes valores de potencia, se realiza el ajuste lineal por mínimos cuadrados y se obtiene un valor de coeficiente de reactividad por potencia numérico.

En la figura Figura 6 se muestra el gráfico con el ajuste lineal para el cálculo numérico de α_P^{num}.

El valor obtenido para $\alpha_P^{núm.}$ y su error asociado aparece listado en la Tabla 6.

Comparando los valores de los coeficientes de potencia en en la Tabla 6 se comprueba que hay congruencia entre los resultados obtenidos numérica y experimentalmente.

Cómo última comparación, se muestra en la Tabla 6 el valor de α_P^{termic} obtenido mediante cálculo numérico con acope termohidráulico `CITVAP - TERMIC` (véase [7]). Este resultado es muy similar al obtenido con el acople con `RELAP` .

El acople neutrónico termohidráulico planteado debe de ser sujeto a muchas modificaciones y validaciones para ser efectivamente incorporado a la línea de cálculo. Aquí se ha mostrado sólo algunas de las experiencias que sirven para llevar a cabo esta tarea. Otras propuestas se hacen en el siguiente capítulo.

Parte II

INCORPORACIÓN DE TRANSITORIOS A CITVAP

En esta parte se analiza la factibilidad de incorporar estudio de transitorios en la población neutrónica a `CITVAP` , como etapa inicial para un acople dinámico entre `CITVAP` y `RELAP` . Se proponen varias vías para lograrlo, una de las cuales es desarrollada y probada. A tal efecto se desarrolla y verifica el programa `Transient` que dentro de un ciclo de cálculos con `CITVAP` , permite realizar cálculo de transitorios. La metodología propuesta es probada con modelos nucleares sencillos, obteniéndose resultados satisfactorios.

5

DESCRIPCIÓN DEL MÉTODO DE CÁLCULO EMPLEADO POR CITVAP-CITATION

5.1 NATURALEZA DEL PROBLEMA RESUELTO POR CITVAP-CITATION

CITVAP emplea como motor de cálculo el código determinístico CITATION, desarrollado por personal de Oak Ridge National Laboratory (véase el manual del usuario [8]) en 1972. Este código calcula la distribución del flujo neutrónico en el núcleo usando la representación en diferencias finitas de la ecuación de difusión multigrupo para distintas geometrías.

5.2 MÉTODO DE SOLUCIÓN

Se describen ahora las ecuaciones y métodos empleados por CITATION para resolver el problema descrito.

La ecuación estacionaria de difusión de neutrones en ausencia de fuentes externas, en una cierta posición r y a una energía E es

$$-\nabla D_{r,E}\nabla\phi_{r,E} + (\Sigma_{a,r,E} + \Sigma_{s,r,E})\,\phi_{r,E} = \\ = \int_{E'}\left(\Sigma_{s,r,E'\to E} + \frac{\chi_{p,E}\,(\nu\Sigma)_{f,r,E'}}{k_{eff}}\right)\phi_{r,E'}dE', \quad (3)$$

donde

$\phi_{r,E}$ = El flujo neutrónico en la posición r a energía E, $\frac{n}{cm^2*s*eV}$,

$\Sigma_{a,r,E}$ = La sección eficaz macroscópica para absorción en la posición r a energía E, cm^{-1},

$\Sigma_{s,r,E}$ = La sección eficaz macroscópica para el scattering en la posición r a energía E,cm^{-1},

$\Sigma_{s,r,E'\to E}$ = La sección eficaz macroscópica para el scattering en la posición r a energía desde la energía E' a la energía E,$\frac{1}{cm*eV}$,

$D_{r,g}$ = El coeficiente de difusión, igual a un tercio del recíproco de la sección eficaz de transporte, cm,

$\nu\Sigma_{f,r,E}$ = La sección eficaz de producción macroscópica (ν es el número medio de neutrones producido por una fisión y Σ_f es la sección eficaz para la fisión) en la posición r a energía E,cm^{-1},

$\chi_{p,g}$ = Espectro energético de aparición de los neutrones prompt $\left(\sum_g \chi_{p,g} = 1{,}0\right)$.

k_{eff} = El factor de multiplicación efectivo, relación entre la tasa de producción de neutrones y tasa de pérdida de neutrones, una incógnita a determinar en el problema.

El continuo de energías es dividido en G grupos discretos tales que el grupo g es caracterizado por las energías $E_{g-1} < E < E_g$. Se supone que el espectro de fisión no presenta dependencia espacial. También se hace una simplificación en el término de transporte (streaming). Estos cambios llevan a

$$-D_{r,g}\nabla^2\phi_{r,g} + \left(\Sigma_{a,r,g} + \sum_{g'=1}^{G}\Sigma_{s,r,g\to g'}\right)\phi_{r,g} = \\ = \sum_{g'=1}^{G}\left(\Sigma_{s,r,g'\to g} + \frac{\chi_{p,g}\,(\nu\Sigma)_{f,r,g'}}{k_{eff}}\right)\phi_{r,g'}, \quad (4)$$

siendo $D_{r,g}$, $\Sigma_{a,r,g}$, $\Sigma_{s,r,g\to g'}$, $(\nu\Sigma)_{f,r,g}$ y $\chi_{p,g}$ los correspondientes parámetros multigrupo obtenidos luego de la condensación del flujo neutrónico por grupos $\phi_{r,g}$.

La Ecuación 4 es usada por `CITATION` discretizada espacialmente en zonas, para determinar el flujo neutrónico $\phi_{r,g}$ y el factor de multiplicación del sistema k_{eff}, mediante un procedimiento iterativo directo (véase la sección 730 de [8]).

A fin de simplificar la notación empleada se introducen los operadores integrales $\mathcal{F}_r$ y $\mathcal{A}_{r,g}$ definidos de la forma

$$\mathcal{F}_r\,\phi_{r,g} \equiv \sum_{g}(\nu\Sigma)_{f,r,g}\;\phi_{r,g}$$

$$\mathcal{A}_{r,g}\,\phi_{r,g} \equiv \Sigma_{a,r,g}\,\phi_{r,g} + \sum_{n}\left(\Sigma_{s,r,g\to n} - + \sum_{g'=1}^{G}\Sigma_{s,r,g\to g'}\right)\phi_{r,g} - \\ - \sum_{g'=1}^{G}\Sigma_{s,r,ng'\to g}\phi_{r,g'}$$

con los que la Ecuación 4 queda de la forma

$$-D_{r,g}\nabla^2\phi_{r,g} + \mathcal{A}_{r,g}\;\phi_{r,g} = \frac{\chi_{p,g}}{k_{eff}}\mathcal{F}_r\;\phi_{r,g}\,. \quad (5)$$

5.3 FUENTE EXTERNA

`CITATION` permite también resolver el problema en que se incluye una fuente externa de neutrones al sistema en estudio. La ecuación que se emplea en este caso se obtiene de la Ecuación 5 igualando k_{eff} a la unidad y añadiendo en el miembro izquierdo la fuente externa:

$$D_{r,g}\nabla^2\phi_{r,g} - \mathcal{A}_{r,g}\;\phi_{r,g} + \chi_{p,g}\mathcal{F}_r\;\phi_{r,g} + S_{r,g} = 0\,. \quad (6)$$

En el caso de problemas con fuente externa, el código no necesita calcular k_{eff} y simplemente determina el flujo neutrónico estacionario resultante de colocar la fuente en el sistema. Debe destacarse que el código exige para problemas de fuente externa que el sistema sea subcrítico ($k_{eff} < 1$), de lo contrario arrojará un error durante el cálculo del flujo.

Es esencial que el sistema sea subcrítico para hacer cálculos con fuente externa en CITVAP

En INVAP se cuenta con poca experiencia previa en el cálculo con fuente externa en CITVAP. Debido a esto se realizaron algunos cálculos de prueba previos, a fin de poner alistar esta funcionalidad para el cálculo de transitorios. El uso de fuentes externas en el cálculo con CITATION se describe en el Apéndice E.

5.4 LIMITACIONES. ESTADOS CUASIESTACIONARIOS Y ESTADOS TRANSITORIOS

La dinámica de un reactor puede ser estudiada para distintas escalas de tiempo. CITVAP-CITATION permite hacer cálculos de gestión de combustible y de quemado, que implican períodos de días, meses o años en la evolución del reactor.

También permite incorporar al cálculo el efecto de algunos productos de fisión como el Xenón y el Samario que afectan a la planta para evoluciones de varias horas.

En los casos anteriores, cada estado del reactor es considerado *cuasiestacionario* (derivada del flujo con respecto al tiempo igual a cero) y por tanto las ecuaciones 5 y 6 son aplicables.

Estas ecuaciones no permiten estudiar la evolución temporal del flujo en períodos de tiempo comparables a los de la vida media de los isótopos precursores y neutrones retardados que son del orden de varios segundos.

En lo que resta de este trabajo se denominarán *transitorios* a los fenómenos que ocurren en el reactor, con tiempos característicos del orden de los segundos.

En los transitorios, la dinámica de neutrones retardados debe de ser tenida en cuenta, por tanto CITVAP-CITATION no presenta una vía directa para el cálculo de transitorios.

6

INCORPORACIÓN DE CÁLCULO DE TRANSITORIOS A CITVAP

6.1 ECUACIÓN DE DIFUSIÓN NO ESTACIONARIA

Para el estudio de transitorios se debe incorporar a la ecuación de difusión el hecho de que el flujo neutrónico puede estar cambiando con el tiempo y que algunos de los neutrones producidos como resultado de eventos de fisión aparecerán retardados por el decaimientode núcleos precursores inestables.

Teniendo esto en cuenta, las ecuaciones que describen los transitorios dentro de la aproximación de difusión de neutrones sin fuente externa son:

$$D_{r,g}\nabla^2\phi_{r,g,t} - \mathcal{A}_{r,g}\ \phi_{r,g,t} + \chi_{p,g}\,(1-\beta)\,\mathcal{F}_r + \\ + \sum_{i=1}^{I}\chi_{i,g}\lambda_i C_{i,r,t} - \frac{1}{\nu_g}\frac{\partial\phi_{r,g,t}}{\partial t} = 0 \quad (7)$$

$$\frac{\partial C_{i,r,t}}{\partial t} = \beta_i\,\mathcal{F}_r\,\phi_{r,g,t} - \lambda_i C_{i,r,t} \qquad i = 1..I\,. \quad (8)$$

donde además de los términos ya definidos para la Ecuación 4 aparecen

$\chi_{i,g}$ = Espectro energético de los neutrones retardados, que provienen de decaimientos del conjunto i-ésimo de productos de fisión (precursores). Normalmente se ajustan los datos nucleares para $I = 6$ grupos de precursores,

β = Fracción total de los neutrones de fisión que aparecen como neutrones retardados, $\beta = \sum_{i=1}^{I}\beta_i$, donde β_i es la fracción correspondiente al i-ésimo grupo de precursores,

$C_{i,r,t}$ = Concentración del i-ésimo grupo de precursores, $\frac{\text{prec}}{\text{cm}^3}$,

λ_i = constante de tiempo del decaimiento β correspondiente al i-ésimo grupo de precursores, seg^{-1}.

Para una explicación detallada de la obtención de la Ecuación 7 y la Ecuación 8 se puede revisar la sección 7.2 de [9].

En las expresiones anteriores se explicita la dependencia temporal sólo de los términos de flujo neutrónico $\phi_{r,g,t}$ y de concentración de precursores $C_{i,r,t}$. Las constantes neutrónicas y secciones eficaces varían apreciablemente (debido a quemado, Xenón, etc.) sólo para períodos largos de tiempo (al menos del orden de horas), lo cual no atañe al estudio de transitorios.

Debe destacarse que los términos β_i y λ_i se refieren a valores efectivos para todos los isótopos físiles presentes en el sistema.

6.2 CONDICIONES INICIALES Y DISCRETIZACIÓN TEMPORAL

En el cálculo de transitorios, se deberá conocer la distribución inicial de flujo neutrónico $\phi_{r,g,t=0}$ y de la concentración de precursores $C_{i,r,t=0}$. El sistema puede partir de una condición de equilibrio inicial (estado de criticidad) en cuyo caso sería válida la relación

$$C_{i,r,0} = \frac{\beta_i}{\lambda_i} \mathcal{F}_r \, \phi_{r,g,0} = \frac{\beta_i}{\lambda_i} \sum_g (\nu \Sigma_{f,r}) \, \phi_{r,g,0} \tag{9}$$

que se obtiene de anular la tasa de cambio de la concentración de precursores en la Ecuación 8.

También puede darse el caso que sea de interés estudiar un sistema que no parte de la criticidad, sino de un transitorio previo, con lo que los términos $\phi_{r,g,t=0}$ y $C_{i,r,t=0}$ no estarán relacionados.

En el cálculo de los flujos como función del tiempo, se discretizará la variable temporal con intervalos Δt. Aplicando dicha discretización a las ecuaciones de difusión para transitorios Ecuación 7 y Ecuación 8 se arriba a

$$\frac{C_{i,r,t+\Delta t} - C_{i,r,t}}{\Delta t} = \beta_i \, \mathcal{F}_{r,g} \, \phi_{r,g,t} - \lambda_i C_{i,r,t} \qquad i = 1..I \tag{10}$$

$$\frac{1}{\nu_g} \frac{\phi_{r,g,t+\Delta t} - \phi_{r,g,t}}{\Delta t} = D_{r,g} \nabla^2 \phi_{r,g,t+\Delta t} - \mathcal{A}_{r,g} \; \phi_{r,g,t+\Delta t} + \\ + (1-\beta) \chi_{p,g} \mathcal{F}_r \; \phi_{r,g,t+\Delta t} + \sum_{i=1}^{I} \chi_{i,g} \lambda_i C_{i,r,t+\Delta t}. \tag{11}$$

El método de resolución de transitorios supondría calcular $C_{i,r,t+\Delta t}$ a partir de $C_{i,r,t}$ y $\phi_{i,r,t}$ con la Ecuación 10. Con este resultado, se podría resolver la Ecuación 11 para el valor de $\phi_{i,r,t+\Delta t}$ con ayuda de algún código neutrónico. Este proceso se repetiría para cada paso de tiempo necesario.

6.3 MECANISMOS PARA INCORPORAR TRANSITORIOS

CITVAP-CITATION sólo permite resolver el flujo neutrónico para estados estacionarios. En el caso de problemas con fuente externa, emplea la Ecuación 6 que repetimos a continuación para mayor claridad:

$$D_{r,g} \nabla^2 \phi_{r,g} - \mathcal{A}_{r,g} \; \phi_{r,g} + \chi_{p,g} \mathcal{F}_r \; \phi_{r,g} + S_{r,g} = 0 \, .$$

Si en la Ecuación 11 para resolver el flujo durante transitorios se introducen las definiciones

$$S'_{r,g,t+\Delta t} \equiv \sum_{i=1}^{I} \lambda_i C_{i,r,t+\Delta t} - \frac{1}{\nu_g} \frac{\phi_{r,g,t+\Delta t} - \phi_{r,g,t}}{\Delta t} \tag{12}$$

$$\mathcal{F}'_r \, \phi_{r,g,t} \equiv (1-\beta) \mathcal{F}_r \, \phi_{r,g,t} = (1-\beta) \sum_g (\nu \Sigma)_{f,r,g} \; \phi_{r,g,t} \tag{13}$$

se llega a una forma similar a la de la Ecuación 6:

$$D_{r,g}\nabla^2\phi_{r,g,t+\Delta t} - \mathcal{A}_{r,g}\ \phi_{r,g,t+\Delta t} + \chi_{p,g}\mathcal{F}'_r\ \phi_{r,g,t+\Delta t} + S'_{r,g,t+\Delta t} = 0\,. \quad (14)$$

Esta expresión reproduce una ecuación estacionaria con *fuente externa equivalente* $S'_{r,g,t+\Delta t}$ y con *fuente de fisiones equivalente* $\chi_{p,g}\mathcal{F}'_r\ \phi_{r,g,t+\Delta t}$, análoga a la Ecuación 6. Puede ser empleada por `CITVAP-CITATION` para calcular el flujo $\phi_{r,g,t+\Delta t}$ a tiempo $t+\Delta t$ a partir de haber calculado previamente $S'_{r,g,t+\Delta t}$ (Ecuación 12) y habérsela pasado al código como fuente externa.

El problema en el cálculo de $S'_{r,g,t+\Delta t}$ según la Ecuación 12 es que depende a su vez del flujo $\phi_{r,g,t+\Delta t}$, que es precisamente la incógnita para el cálculo en $t+\Delta t$.

Se proponen dos vías alternativas para el cálculo de $S'_{r,g,t+\Delta t}$ en cada instante de tiempo $t+\Delta t$:

A. Despreciar el término $\frac{1}{v_g}\frac{\phi_{r,g,t+\Delta t}-\phi_{r,g,t}}{\Delta t}$ en la Ecuación 12. En este caso

$$S'_{r,g,t+\Delta t} = \sum_{i=1}^{I} \lambda_i C_{i,r,t+\Delta t} \quad (15)$$

y sólo depende de $\phi_{r,g,t}$ a tiempo anterior t (a través de $C_{i,r,t+\Delta t}$ según la Ecuación 10). Este método numérico es un *Euler hacia atrás* explícito (véase [10]), y por tanto el error numérico de la solución numérica respecto a la real crecerá proporcionalmente al paso de tiempo Δt empleado en el cálculo.

B. Realizar un cálculo iterativo de $S'_{r,g,t+\Delta t}$ y $\phi_{r,g,t+\Delta t}$ con las expresiones Ecuación 12 y Ecuación 14 a partir de una suposición inicial $\phi^0_{r,g,t+\Delta t}$ y hasta una convergencia al valor real de $\phi_{r,g,t+\Delta t}$.Estas iteraciones habría que realizarlas *para cada instante* $t+\Delta t$ *que se desee calcular*. Éste es un método implícito, que garantiza una estabilidad incondicional respecto al paso de tiempo empleado. Además no necesita de la aproximación plantada en A. Sin embargo, el costo computacional es mucho más alto, pues cada paso de tiempo requeriría de un ciclo de cálculo para la obtención de $\phi_{r,g,t+\Delta t}$. Las posibilidades y limitaciones de este método quedan como propuesta para trabajos futuros en la resolución de transitorios.

De las dos vías propuestas, se decidió implementar la primera (A) para estudiar la resolución de transitorios con `CITVAP-CITATION`.

La hipótesis de que $\frac{1}{v_g}\frac{\phi_{r,g,t+\Delta t}-\phi_{r,g,t}}{\Delta t}$ es despreciable en general se cumple si $\frac{1}{v_g}\frac{\phi_{r,g,t+\Delta t}-\phi_{r,g,t}}{\Delta t} \ll \sum_{i=1}^{I}\lambda_i C_{i,r,t+\Delta t}$, o sea, si la tasa de cambio de la concentración de neutrones es pequeña frente a la tasa de aparición de neutrones retardados. Esto es aplicable a eventos donde no existen cambios demasiado bruscos en la población neutrónica. Para revisar la validez de esta aproximación, se pueden proponer experiencias de validación y verificación del método. En esta etapa del proyecto, se supondrá que se cumple la aproximación.

7 DESARROLLO DE UN PROGRAMA PARA EL CÁLCULO DE TRANSITORIOS

El primer paso realizado para incorporar el cálculo de transitorios a `CITVAP` fue hacer los cálculos con ayuda de un programa externo implementado explícitamente para probar y verificar los mecanismos propuestos en las secciones anteriores.

Se implementó `Transient`, programado en `FORTRAN 95`. En esta sección se describen las pautas generales seguidas para el desarrollo y la validación del programa. En el Apéndice F se presenta una descripción detallada del código fuente para eventuales modificaciones.

7.1 OBJETIVO DEL PROGRAMA `transient`

`Transient` calcula, en conjunción con `CITVAP`, el flujo $\phi_{r,g,t}$ y la concentración de precursores $C_{i,r,t}$ en un núcleo que experimenta un transitorio. El desarrollo de este programa busca presentar un acercamiento inicial a la propuesta para integrar el cálculo de transitorios a `CITVAP`. Hoy, lo que es un código externo, podrá ser integrado en el futuro al código fuente de `CITVAP` para que el cálculo de transitorios pase a ser una más de sus funcionalidades.

7.2 REQUISITOS Y LIMITACIONES DEL CÁLCULO DE TRANSITORIOS CON `transient`

El ciclo de cálculo de transitorios con `Transient` presupone que el sistema parte de un estado crítico ($k_{eff} = 1$).

Para el cálculo de concentración de precursores se desprecia el término de la tasa de variación de la concentración de neutrones (ítem A en la lista de mecanismos planteados en la Sección 6.3), que es la vía menos costosa computacionalmente de las dos propuestas.

Una importante observación para el cálculo de transitorios con `Transient` es que las perturbaciones introducidas no podrán ser tales que la reactividad insertada por las mismas en el sistema sea mayor al valor de β. Esto corresponde a una situación de *prompt critical* y el cálculo arrojará un error.

7.3 ARCHIVOS EXTERNOS Y PROGRAMAS QUE INTERVIENEN EN EL CICLO DE CÁLCULO

En la Tabla 7 se describen los distintos programas que intervienen en el ciclo de cálculo.

Para el cálculo de transitorios, `Transient` emplea varios archivos auxiliares. En la Tabla 8 se describen estos archivos. Los nombres de archivo mostrados

en la primera columna son genéricos, durante la corrida del programa pueden ser cambiados a conveniencia.

NOMBRE DEL PROGRAMA	DESCRIPCIÓN
`CITVAP`	Programa de cálculo de núcleo. A partir de la información que se pasa en el archivo de entrada `Nucleo.CII` y en la biblioteca de materiales `XS.bib` calcula el flujo neutrónico por zonas y por grupos de energía $\phi_{r,g,t}$. Los resultados son guardados en la base de datos `Nucleo.CDB` y en el archivo de salida `Nucleo.CIO`.
`Transient`	Programa para cálculo de transitorios en conjunción con `CITVAP`. A partir de $\phi_{r,g,t}$ extraída de `Nucleo.cdb` calcula $C_{i,r,t+\Delta t}$ y $S'_{r,g,t+\Delta t}$ para el próximo instante de tiempo $t + \Delta t$.
`HXS`	Programa para gestión de bases de datos de materiales de `CITVAP`.

Tabla 7: Programas que intervienen en el ciclo de cálculo de transitorios.

7.4 FLUJO DEL CICLO DE CÁLCULO

En la Figura 7 se muestran las distintas etapas del cálculo de transitorios con `Transient` y `CITVAP`. A continuación se explicará cada una de estas, basándose en la numeración de los cuadros de la figura.

1. Para empezar el ciclo de cálculo (*Cuadro 1 de la Figura 7*), se debe contar con el modelo del núcleo de un reactor en estado crítico inicial ($k_{eff} = 1$) representado en un archivo de entrada de `Nucleo.CII`.

2. Se corre `CITVAP` con el archivo de entrada `Nucleo.CII` debidamente condicionado según los requerimientos aclarados en el punto anterior (*Cuadro 2*).

3. Se corre el programa `Transient` (*Cuadro 3*). Al tratarse de la primera corrida, el programa identificará que no se han calculado las concentraciones de precursores $C_{i,r,t}$ debido a que no existe el archivo `precursors.dat`. Esto lo tomará como una señal de que el sistema se encuentra en estado crítico inicial. Por tanto calculará la concentración de precursores en equilibrio $C_{i,r,t=0} = \frac{\beta_i}{\lambda_i}\mathcal{F}_{r,g}\,\phi_{r,g,t=0}$ a partir del flujo neutrónico $\phi_{r,g,t=0}$ extraído de la base de datos de `CITVAP` `Nucleo.cdb`. También calcula la fuente inicial de precursores y la almacena en `source.dat` .

4. Después de este paso, es el momento de introducir alguna perturbación en el sistema nuclear (*Cuadro 4*). Esta puede ser un desplazamiento de las

NOMBRE DEL ARCHIVO	DESCRIPCIÓN
`Nucleo.CII`	Archivo de entrada para `CITVAP`. Contiene el modelo del núcleo del reactor nuclear dividido en *zonas espaciales* y las especificaciones necesarias para el cálculo del flujo neutrónico $\phi_{r,g,t}$.
`Nucleo.CDB`	Base de datos `CITVAP`. Contiene información de los resultados del cálculo realizado con `CITVAP`. De la base de datos se puede leer el flujo neutrónico calculado $\phi_{r,g,t}$ y las secciones eficaces empleadas en el modelo.
`XS.bib`	Base de datos de materiales para el modelo de núcleo en `CITVAP`. Esta base de datos es creada a partir de cálculos previos con el programa de cálculo de celda `CONDOR` y el programa auxiliar `HXS`.
`NuclearData.DAT`	Archivo con datos nucleares para el cálculo de transitorios. En este archivo creado por el usuario, se deberán especificar el paso de tiempo para el cálculo Δt y constantes necesarias para el cálculo de transitorios (β_i,λ_i,$\chi_{i,j}$). El formato requerido para este archivo se describe en el Apéndice F, Subsección F.3.1.
`precursors.DAT`	Archivo auxiliar empleado por `Transient` para almacenar la concentración de precursores $C_{i,r,t}$ del cálculo a tiempo t para ser usada en el cálculo a tiempo $t + \Delta t$.
`source.DAT`	Archivo auxiliar empleado por `Transient` para almacenar los valores de fuente auxiliar $S'_{r,g,t}$ del cálculo a tiempo t para ser usada en el cálculo a tiempo $t + \Delta t$.
`results.RLT`	Archivo de salida de `Transient` donde se almacenan los resultados de $\phi_{r,g,t}$, $C_{i,r,t}$ y $S'_{r,g,t}$ para todos los instantes de tiempo t en los que se calculó el transitorio. Este archivo puede ser empleado luego para graficar y analizar la evolución temporal de estas magnitudes neutrónicas.

Tabla 8: Archivos externos empleados en el ciclo de cálculo de transitorios con `Transient`

barras de control, una inserción de venenos en el sistema o cualquier otro evento que produzca una variación temporal en el flujo neutrónico.

5. A continuación se debe preparar el archivo de entrada `Nucleo.CII` para el cálculo de transitorios (*Cuadro 5*). Esto implica afectar la fuente de fisiones por un factor $(1 - \beta)$ a fin de que en lo adelante, esta fuente sólo represente la contribución de los neutrones prompt. También debe incluirse en el archivo el problema de fuente externa. Esto se hace incluyendo la opción `NGC 10 -5` en la sección 1 de `Nucleo.CII`, e incluyendo la información

de la sección 026 en el archivo. Esta información se incluye simplemente haciendo referencia con el comando `FILE` al archivo `source.dat`. Una explicación detallada de cómo incluir el problema de fuente externa se da en el Apéndice E.

6. Una vez que se introdujo la perturbación que causará el transitorio, y se hicieron las modificaciones al archivo `Nucleo.cii` para empezar el cálculo, se empieza un ciclo de cálculo con tantas etapas como intervalos de tiempos se quieran simular en el transitorio.

7. Se corre `CITVAP` (*Cuadro 6*)para que resuelva el problema de fuente externa planteado. En este caso, [CITVAP calculará el flujo $\phi_{r,g,t=\Delta t}$ y se notará un cambio respecto al flujo inicial $\phi_{r,g,t=0}$ debido a la perturbación introducida.

8. Se ejecuta `Transient` (*Cuadro 7*). En esta etapa, `Transient` conoce que ya el sistema se encuentra fuera del equilibrio. A partir de los $\phi_{r,g,t}$ obtenidos de `Nucleo.cdb` del cálculo previo con `CITVAP`, calcula con la Ecuación 10 y la Ecuación 12 los valores de $C_{i,r,t+\Delta t}$ y $S'r, g, t + \Delta t$ para el instante $t + \Delta t$ respectivamente . Actualiza con estos valores los archivos `precursors.dat` y `source.dat` respectivamente y también escribe en el archivo de resultados `results.rlt`.

9. Los pasos 7 y 8 son ejecutados una y otra vez con ayuda de un script externo hasta que se satisfaga alguna condición de convergencia requerida o hasta que se tengan todos los resultados para un lapso de tiempo deseado. La condición de convergencia puede referirse a que el flujo no cambie dentro de un valor de error prefijado, cuando se conoce que el estado final del sistema debe de ser de flujo estable.

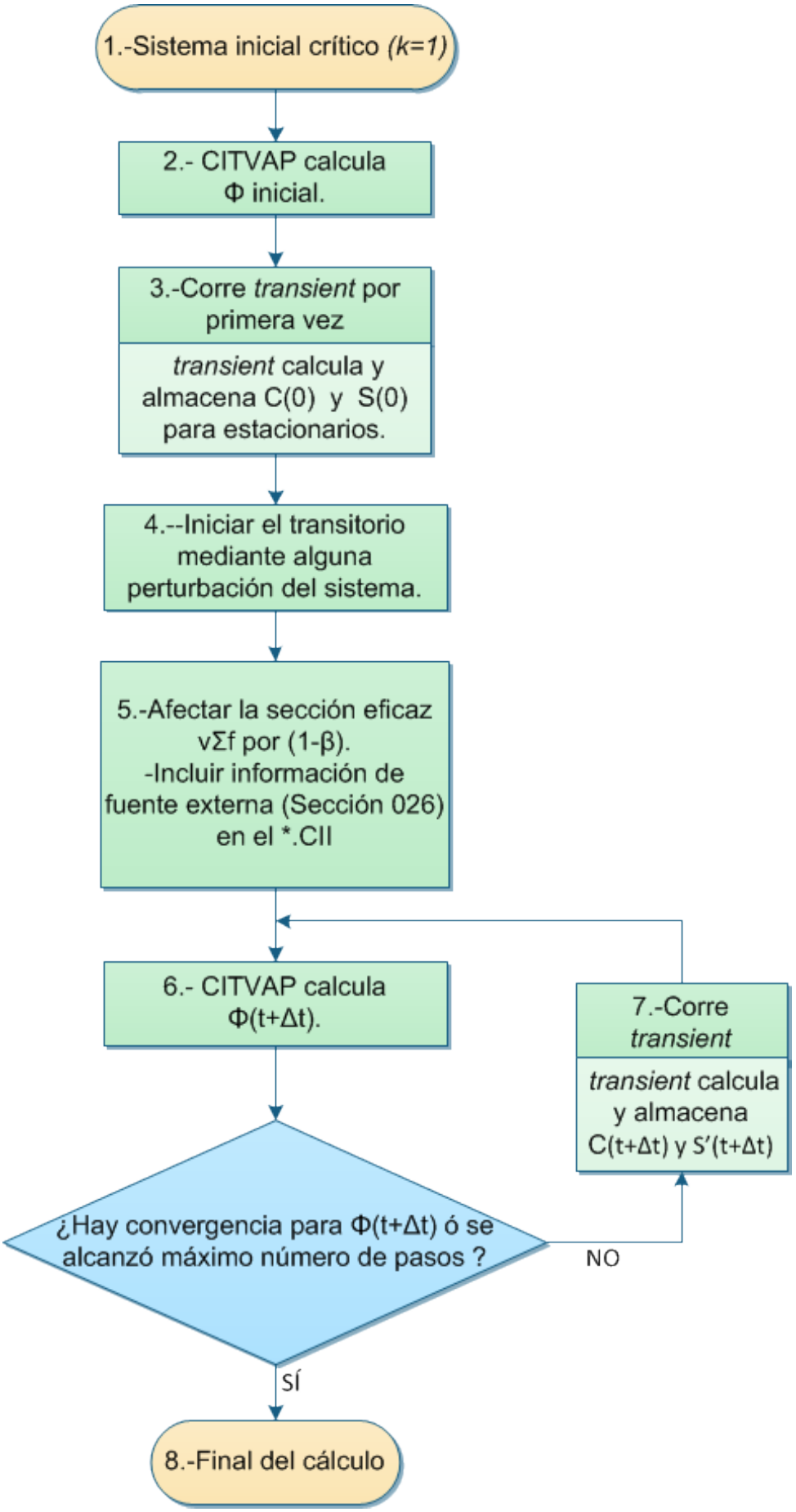


Figura 7: Diagrama de flujo para el ciclo de cálculo de transitorios.

8

VALIDACIÓN DEL CICLO DE CÁLCULO DE TRANSITORIOS

A fin de validar la metodología planteada para resolver transitorios, se realizaron varias pruebas de cálculo de transitorios para sistemas nucleares sencillos de analizar. Los resultados numéricos obtenidos fueron contrastados con las correspondientes soluciones analíticas.

8.1 MODELO DE NÚCLEO EMPLEADO

En todas las experiencias se utilizó un modelo de reactor homogéneo infinito en tres dimensiones.

En el archivo de entrada a `CITVAP` (en lo adelante `HomInf.CII`) se encuentra el modelo empleado en las experiencias realizadas, para el caso del estado inicial crítico del reactor.

Para esto, se empleó un modelo de núcleo con geometría cúbica, de una sola zona, arista $30cm$ y mallado $3X3X3$.

El modelo de reactor infinito se hizo estableciendo condiciones de frontera especulares en toda le geometría del núcleo.

Para modelar el flujo neutrónico se empleó un grupo de energía. Para la concentración de precursores usó un grupo de precursores. El archivo de entrada `Nucleo.CII` empleado en todos los casos fue el mismo para el estado inicial crítico.

La potencia inicial establecida fue de $1 \times 10^{-11} MW$ a fin de obtener mayor precisión en los valores de flujo neutrónico según lo explicado en el Apéndice E, Subsección E.2.4. Esta potencia dio un valor de flujo inicial de $\phi_{1,1,t=0} = 771{,}83$ n/s.

Los parámetros neutrónicos empleados para el cálculo fueron:

$$\beta = 0{,}0065$$
$$\lambda = 7{,}645 \times 10^{-2} s^{-1}$$
$$\chi = 1 \text{ ya que hay un sólo grupo de energías.}$$

Para el sistema inicial crítico:

$$\Sigma_A = \nu\Sigma_f = 2{,}08396 \times 10^{-2} cm^{-1}$$
$$\phi_0 = 771{,}830 \frac{n}{cm^2 s}$$
$$C_0 = 1{,}367 \frac{prec}{cm^3} \text{ calculado según } C_0 = \frac{\beta\nu\Sigma_f}{\lambda}\phi_0$$

Para lograr la criticidad inicial se colocó un material ficticio en el Reactor homogéneo infinito en que se cumpliera $\nu\Sigma_f = \Sigma_a$.Las perturbaciones eran

introducidas cambiando dicho material por otro en el que $k_{eff} \neq 1$. Esto se hacía imponiendo $\Sigma_a \neq \nu\Sigma_f$ para dicho material durante el cálculo de celda con `condor`.

La razón de usar modelos simplificados es que las correspondientes ecuaciones neutrónicas tienen soluciones analíticas con las que resulta fácil contrastar los resultados numéricos del ciclo de cálculo. Además, el tiempo de cálculo se abrevia considerablemente y de esa forma es más sencillo corregir errores en el diseño del ciclo y volver a realizar las corridas.

8.2 SOLUCIÓN ANALÍTICA PARA UN CAMBIO INSTANTÁNEO DE $\nu\Sigma_f$

Determinemos la solución analítica del flujo neutrónico $\phi(t)$ y la concentración de precursores $C(t)$ conociendo sus valores iniciales de equilibrio ϕ_0 y $C_0 = \frac{\beta\nu\Sigma_f}{\lambda}\phi_0$.

Las correspondiente ecuaciones neutrónicas son

$$\frac{1}{\nu}\frac{d\phi}{dt} = (1-\beta)\nu\Sigma_f(t)\phi + \lambda C(t) - \Sigma_a\phi \tag{16}$$

$$\frac{dc}{dt} = \beta\nu\Sigma_f(t)\phi - \lambda C(t). \tag{17}$$

En estas expresiones se ha explicitado la dependencia temporal de $\nu\Sigma_f$. Esto se ha hecho así para indicar que se introducirán perturbaciones en esta sección eficaz para producir transitorios.

Si en la Ecuación 16 se desprecia el término $\frac{1}{\nu}\frac{d\phi}{dt}$ según la aproximación de variación lenta del flujo (descrita en el punto *A* de Sección 6.3) y resolvemos para $\phi(t)$ y $C(t)$ se obtiene

$$C(t) = C_0 \exp\left[\int_0^t \frac{\beta\nu\Sigma_f(\tau)\,d\tau}{\Sigma_a - (1-\beta)\,\nu\Sigma_f(\tau)} - \lambda t\right] \tag{18}$$

$$\phi(t) = \frac{\beta\nu\Sigma_f\phi_0}{\Sigma_a - (1-\beta)\,\nu\Sigma_f(t)} \exp\left[\int_0^t \frac{\beta\nu\Sigma_f(\tau)\,d\tau}{\Sigma_a - (1-\beta)\,\nu\Sigma_f(\tau)} - \lambda t\right]. \tag{19}$$

En las experiencias realizadas para la validación del ciclo de cálculo de transitorios, se introdujeron perturbaciones al valor de la sección eficaz $\nu\Sigma_f$. Las perturbaciones se introdujeron aumentando su valor inicial en una fracción p de manera que el valor final tome la forma $(1+p)\nu\Sigma_f$ siendo $\nu\Sigma_f$ el valor inicial.

Si la perturbación es introducida a tiempo $t = 0$, las correspondientes expresiones analíticas para el flujo neutrónico y la concentración de precursores es:

$$\phi(t) = \left\{ \begin{array}{ll} \phi_0, & \text{para } t \leqslant 0 \\ \frac{\beta\nu\Sigma_f\phi_0}{\Sigma_a-(1-\beta)(1+p)\nu\Sigma_f} \exp\left[\left(\frac{\beta\nu\Sigma_f}{\Sigma_a-(1-\beta)(1+p)\nu\Sigma_f} - 1\right)\lambda t\right], & \text{para } t > 0 \end{array} \right\} \tag{20}$$

$$C(t) = C_0 \exp\left[\left(\frac{\beta\nu\Sigma_f}{\Sigma_a - (1-\beta)\,(1+p)\,\nu\Sigma_f} - 1\right)\lambda t\right] \tag{21}$$

Se obtiene entonces que el flujo neutrónico sufrirá un salto instantáneo (prompt jump) y luego seguirá una evolución exponencial con el tiempo. La concentración de precursores seguirá también una evolución exponencial. El prompt jump se debe a la rápida evolución de los neutrones prompt ante la introducción de la perturbación.

8.3 ESCALÓN POSITIVO DE REACTIVIDAD

La primera prueba realizada consistió en aumentar en una fracción $p = +0{,}001$ el valor de $\nu\Sigma_f$ respecto a su valor inicial. Esto introduce una inserción positiva de reactividad en el sistema.

8.3.1 *Determinación del paso de tiempo Δt óptimo para el cálculo*

El método numérico empleado para resolver transitorios con `Transient` es un *Euler hacia adelante*. Por tanto, el error ε cometido después del n-ésimo paso de tiempo es de orden $h = 1$ respecto a la magnitud Δt del mismo. Esto es:

$$\varepsilon(t_n) \equiv |\phi(t_n) - \phi_n| = k\Delta t^h \text{ con } h = 1, \tag{22}$$

donde $\phi(t_n)$ representa la solución analítica (Ecuación 20) de las ecuaciones diferenciales que definen el problema (Ecuación 16 y Ecuación 17) ; y ϕ_n la solución numérica del ciclo de cálculo para el mismo tiempo $t_n = n \times \Delta t$. La constante k depende del intervalo de tiempo t_n pero es independiente del paso de tiempo Δt. Una deducción del orden de las soluciones numéricas de algoritmos de *Euler* puede ser encontrada en [10].

La Ecuación 22 deja claro que la magnitud del paso de tiempo Δt empleado para el cálculo no debe de ser demasiado grande como para comprometer la exactitud de la solución numérica. Por otro lado un paso de tiempo Δt muy chico implicaría un alto costo computacional, debido al tiempo que tarda cada cálculo con `CITVAP` para modelos de núcleos más complejos. Por tanto, es importante encontrar un paso de tiempo Δt de cálculo que permita un buen compromiso entre la precisión de la solución y el costo computacional para determinarla.

Se realizaron cálculos del flujo neutrónico ante el escalón positivo de reactividad descrito anteriormente, para distintos pasos de tiempo Δt de cálculo.

En la figura Figura 8 se muestran los resultados numéricos del flujo neutrónico ϕ_n al problema de cambio positivo de reactividad ($p = 0{,}001$) para distintos pasos de tiempo Δt de cálculo. También se muestra la correspondiente solución analítica $\phi(t_n)$ (Ecuación 20).

En la Tabla 9 se muestran los correspondientes errores absoluto ε y relativo ε_r para ϕ_n para un tiempo típico de transitorios $t_n = 150s$, para los distintos Δt usados.

Se observa que un paso de tiempo $\Delta t = 0{,}90s$ introduce un error de un 2,7 % con respecto a la correspondiente solución analítica. Este error se puede tomar como aceptable al tiempo estudiado y elegir $\Delta t = 0{,}90s$ cómo óptimo para los cálculos de este modelo particular.

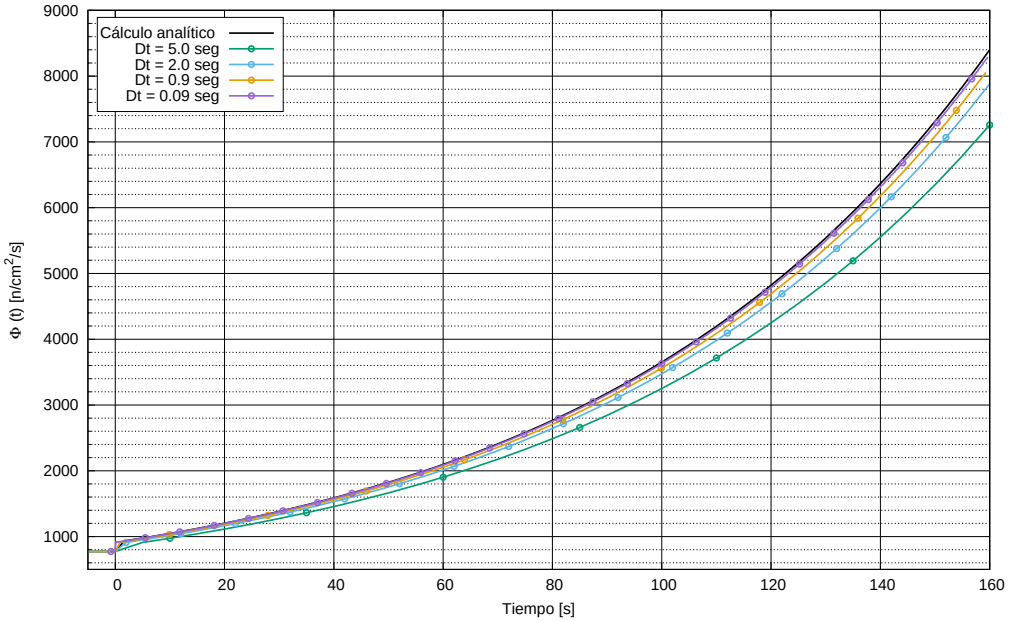


Figura 8: Comparación del cálculo analítico de $\phi(t)$ con cálculos numéricos de distintos Δt

$\Delta t[s]$	$\phi_n \left[\frac{n}{cm^2 s}\right]$	$\varepsilon \left[\frac{n}{cm^2 s}\right]$	ε_r [%]
5,00	6347,86	964,63	13,2 %
2,00	6876,70	435,79	6,0 %
0,90	7117,90	194,59	2,7 %
0,09	7253,77	58,72	0,8 %
$\phi(150s) =$	7312.49		

Tabla 9: Errores absoluto $(\varepsilon = |\phi_n - \phi(t_n)|)$ y relativo $(\varepsilon_n = \varepsilon/\phi(t_n))$ entre solución analítica $\phi(t_n)$ y cálculos numéricos ϕ_n para distintos Δt a tiempo $t_n = 150s$.

El análisis aquí realizado sólo pretende ilustrar la problemática de la discretización temporal en la exactitud de los resultados obtenidos y en el costo computacional de los cálculos. Actualmente muchos algoritmos incluyen pasos de tiempo variables.Estos algoritmos buscan pasos de tiempos mayores cuando el flujo cambia muy poco de un paso a otro. Una modificación de este tipo es recomendable en versiones futuras del método de cálculo de transitorios.

8.3.2 *Cálculo de flujo neutrónico y la concentración de precursores para $\Delta t = 0{,}02s$. Prompt Jump.*

En la figura Figura 9 se muestran los valores de flujo neutrónico y concentración de precursores para un paso de tiempo $\Delta t = 0{,}02s$. Se puede observar el prompt jump inicial en la población neutrónica. El flujo ϕ inicial tiene un valor de $771{,}830 \frac{n}{cm^2 s}$, mientras que luego del prompt jump toma un valor

$911{,}00\frac{n}{cm^2 s}$ para un cálculo de $\Delta t = 0{,}02s$. Comparando la relación que guardan estos dos valores con la relación teórica.:

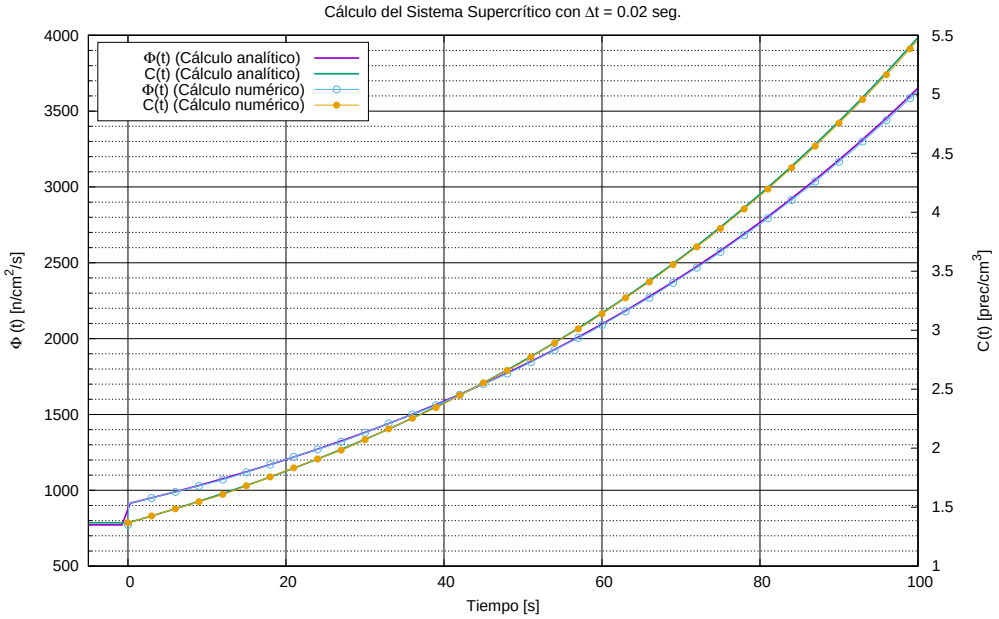


Figura 9: Resultados numérico y analítico de ϕ y C para una inserción positiva de reactividad. $\Delta t = 0{,}02s$.

$$\frac{\phi_{t=0^+}}{\phi_{t=0^-}} = \frac{\beta \nu \Sigma_f}{\Sigma_a - (1-\beta)(1+p)\nu\Sigma_f} = 1{,}18 \tag{23}$$

obtenida según la Ecuación 20 se obtiene una total correspondencia entre la relación numérica y la analítica para el prompt jump.

8.3.3 *SCRAM*

Otra experiencia planteada fue introducir un SCRAM en el sistema. Esto se logró directamente igualando a cero el valor de $\nu\Sigma_f$ en la biblioteca empleada por `CITVAP`. En las ecuaciones Ecuación 20 y Ecuación 21 se corresponde a $p = -1$. Los gráficos de flujo y concentración de precursores obtenidos numéricamente se muestran en Figura 10 , junto a las correspondientes soluciones analíticas.

Se observa nuevamente una muy buena correspondencia entre la solución analítica y los resultados calculados numéricamente. En el flujo neutrónico se observa un salto instantáneo negativo debido a la rápida desaparición de los neutrones prompt luego del SCRAM.

8.4 CONCLUSIONES DE LA VALIDACIÓN

En todas las experiencias mostradas se obtuvieron resultados satisfactorios al comparar las soluciones numéricas con las correspondientes soluciones ana-

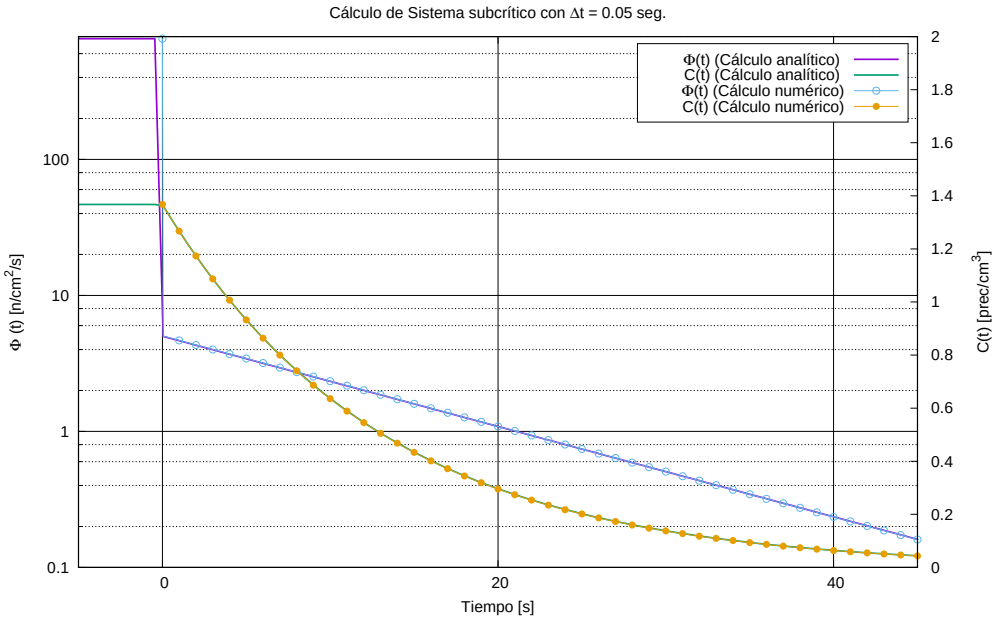


Figura 10: Resultados numérico y analítico de ϕ y C para SCRAM. $\Delta t = 0{,}05s$.

líticas. El modelo de Núcleo empleado era simplificado para manejar con facilidad las distintas variables neutrónicas y analizar el comportamiento del flujo. Se impone realizar más experiencias con modelos más complejos para ampliar la validación de la metodología de cálculo de transitorios.

TRABAJO FUTURO Y CONCLUSIONES

9 MEJORAS Y PROPUESTAS FUTURAS

Para dar conclusión al proyecto, se presentan en este capítulo algunas propuestas a las metodologías planteadas para darle continuidad en próximas etapas de desarrollo.

9.1 PROPUESTAS PARA EL DESARROLLO DEL ACOPLE ESTACIONARIO (PARTE I)

En la primera parte de este trabajo se desarrolló una metodología para realizar un acople entre los códigos `RELAP` y `CITVAP` . Como parte de ello se implementaron los programas `relap2citvap` y `citvap2relap`. Además se expuso el flujo de cálculo, con sus capacidades y limitaciones. A continuación se presentan algunas mejoras que se pueden añadir a esta metodología:

- El archivo `relaciones` incluye múltiples líneas vinculando componentes de los modelos de `relap` y `citvap` para el núcleo del reactor. Es imprescindible que las relaciones entre los modelos sean coherentes para obtener resultados satisfactorios. Se propone desarrollar un algoritmo de chequeo de compatibilidad de las relaciones, a fin de prevenir errores en la creación del archivo y facilitar el uso del acople por parte del usuario final.
- Se podría implementar criterios múltiples de convergencia en el ciclo del acople neutrónico termohidráulico. Por ejemplo, se podría exigir que tanto la reactividad y la distribución de densidad de potencia o la de flujo neutrónico por grupos se encuentren convergidos. Esto haría más robusto el método de acople.
- Actualmente los parámetros de temperaturas y densidades empleados para el acople sólo aplican a las zonas correspondientes a canales refrigerantes. Esto quiere decir que los materiales estructurales y reflectores que componen al reactor, no intervienen en la realimentación termohidráulica y sus temperaturas se mantienen constantes. Esto influye en los resultados numéricos obtenidos con el acople. Queda entonces proponer un modelo de transferencia de calor hacia los materiales estructurales y reflectores del núcleo para incluirlo en una versión mejorada y más realista del acople neutrónico termohidráulico.

9.2 PROPUESTAS PARA EL DESARROLLO DEL CÁLCULO DE TRANSITORIOS (PARTE II)

La metodología mostrada en la segunda parte de este trabajo para incorporar el cálculo de transitorios a `CITVAP` no pretende sino ser un acercamiento

inicial a la solución de esta tarea. Queda mucho por resolver. Aquí se muestran algunas propuestas para darle continuidad a la solución del problema.

9.2.1 *Modificación del programa `CITVAP-CITATION` y su pre-procesador `precit`*

El código fuente de `CITVAP` deberá ser modificado convenientemente para incorporar la funcionalidad de cálculo de transitorios al programa, y así quitar la necesidad de programas externos (como `Transient`). Esta modificación se basará en el código de `Transient` descrito en el Apéndice F.

Actualmente la precisión que ofrece la sección 026 de `CITATION` para definir la intensidad de la fuente neutrónica es demasiado baja (tal como se describió en el Apéndice E,Subsección E.2.4). Este problema se deberá resolver cambiando el formato de lectura en el código fuente de `CITATION`. Así mismo, se deberá cambiar el número de dígitos para especificar las zonas en la sección 026.

También se deberá modificar el preprocesador `precit` a fin de cumplir una serie de tareas previas al cálculos, que incluirán

- La determinación automática de las constantes neutrónicas (β_i, λ_i , $\chi_{i,g}$) correspondientes al problema planteado.De esta manera se quita la necesidad de un archivo externo escrito por el usuario como `NuclearData.dat`.
- Permitir al usuario establecer un paso de tiempo de cálculo Δt. Se puede estudiar la posibilidad de determinar un paso de tiempo óptimo de manera automática (sin intervención del usuario) según las condiciones propuestas en Subsección 8.3.1.
- Extraer la información del número de precursores (variable `NoOfPrecursors`) del cálculo de celda con `CONDOR`. Se deberá emplear las últimas versiones de `HXS` para incorporar esta información en el modelo de `CITVAP` .
- La creación de un campo en la base de datos de `CITVAP` para almacenar la concentración de precursores. Así se remplazaría el uso de archivos externos auxiliares como `precursors.dat`. Al agregar los precursores en la base de datos estos se almacenaran automáticamente en función del tiempo lo mismo que los flujos,potencias, temperaturas, densidades, etc etc. Con lo que el flux automáticamente los presentará para graficar. en función del tiempo y del espacio.
- Queda evaluar si es necesario aumentar la capacidad de `CITVAP` para introducir perturbaciones en el sistema que devengan en transitorios de la población neutrónica. En particular, se debería poder modelar de manera sencilla un cambio termohidráulico como cambios en la temperatura de entrada del refrigerante al núcleo o cambios en el caudal.

9.2.2 *Mejoras y extensiones al método de solución de transitorios*

- Buscar una solución al problema de que el espectro energético de la fuente externa es universal (descrito en la Subsección E.2.3) y no puede ser

especificado un espectro por zona. Se debe realizar un estudio de las implicaciones de esta limitación.

- Evaluar si es necesario resolver la ecuación temporal de difusión (Ecuación 11) sin la aproximación

$$\frac{1}{\nu_g}\frac{\phi_{r,g,t+\Delta t}-\phi_{r,g,t}}{\Delta t} \ll \sum_{i=1}^{I} \lambda_i C_{i,r,t+\Delta t}.$$

- Todas las modificaciones al método de cálculo deberán ser respaldadas con un proceso de validación. En particular, deberían contrastarse los resultados obtenidos con datos experimentales de experiencias con reactores.

- Introducir la posibilidad de que el estado inicial del sistema antes del transitorio a calcular no seá un estado de criticidad y por lo tanto no se cumpla la relación de equilibrio entre flujo neutrónico y concentración de precursores (Ecuación 9).

9.3 DESARROLLO DE UN ACOPLE DINÁMICO ENTRE RELAP Y CITVAP

Como se planteó en la introducción del proyecto, el mismo representa la fase inicial de un desarrollo más amplio, que apuntará a aprovechar las capacidades de modelado de transitorios y accidentes de planta que brinda RELAP para combinarlas con las capacidades de modelado de flujo neutrónico en tres dimensiones que brinda CITVAP .

El paso lógico siguiente es incorporar al cálculo de transitorios que se desarrolló para CITVAP en la Parte II, la posibilidad de realimentación termohidráulica con RELAP . Así, las constantes neutrónicas y las secciones eficaces, que dependen de parámetros termohidráulicos (T_{ref} , ρ_{ref} y T_{comb}) variarán en el tiempo debido a que dichos parámetros también lo harán. La simulación con RELAP permitirá conocer la variación temporal de dichos parámetros, pero para ello se necesitará conocer el comportamiento de la distribución de densidad de potencia que depende del flujo neutrónico. Se arribará entonces a un acople neutrónico termohidráulico entre parámetros neutrónicos y termohidráulicos que a su vez varían en el tiempo debido a las perturbaciones introducidas en el sistema.

Un acople así permitiría modelar accidentes y transitorios de planta con mucha mayor precisión que la que actualmente brinda el modelo de cinética puntual que emplea actualmente RELAP .

10 CONCLUSIONES

10.1 ACOPLE ESTACIONARIO

En la primera parte del proyecto se realizó el acople estacionario entre CITVAP y RELAP . Para ello se implementó un ciclo de cálculo que incluía los programas desarrollados relap2citvap y citvap2relap.

El desarrollo de los programas fue llevado a cabo mediante:

- Estudio de los archivos de entrada y salida de CITVAP y RELAP . Desarrollo de una metodología para relacionar los modelos descritos para los dos programas.
- Desarrollo de una biblioteca utilites.lib en Fotran95 con las rutinas necesarias para la manipulación de los archivos de entrada y salida de CITVAP y RELAP .
- Desarrollo de los programas relap2citvap y citvap2relap, que usando la biblioteca utilites.lib y las relaciones pasadas por el usuario final permiten el flujo de información entre los programas RELAP y CITVAP .
- Verificación del correcto funcionamiento de los programas relap2citvap y citvap2relap. Un proceso de prueba y error llevó a optimizar el funcionamiento de los mismos.

Con los programas desarrollados, se pasó a realizar una verificación del funcionamiento de la metodología planteada para el acople estacionario.

Se comprobó que los resultados del acople eran independientes de las condiciones iniciales que se impusieran a los parámetros de estado termohidráulico del núcleo.

Se realizó un cálculo del coeficiente de realimentación por potencia del núcleo del reactor OPAL y se comparó con el correspondiente resultado experimental. Los valores coincidían dentro de los correspondientes márgenes de error. También coincidía con el resultado obtenido del acople neutrónico termohidráulico entre CITVAP y otro código termohidráulico (TERMIC)

10.2 ESTUDIO DE TRANSITORIOS

Para la segunda parte del proyecto, se desarrolló una manera de incorporar el estudio de transitorios a CITVAP . Para ello:

- Se analizó el método de cálculo que empleado por CITVAP-CITATION para resolver estados estacionarios del núcleo.

- Se propuso una vía para incluir los términos transitorios de la ecuación de difusión a través de la opción de fuente externa de neutrones que permite `CITVAP-CITATION`.
- Se desarrolló el programa `transient` y la metodología necesaria para llevar a cabo el cálculo de transitorios con `CITVAP` , haciendo uso de la idea de la fuente externa.
- Se verificó el correcto funcionamiento del cálculo de transitorios con modelos sencillos de núcleo y perturbaciones. En todos los casos hubo un buen compromiso entre los resultados numéricos y los correspondientes resultados analíticos.

10.3 ETAPAS POSTERIORES

Como se expuso en la introducción, este proyecto fue la etapa inicial para llegar al acople dinámico entre los códigos `CITVAP` y `RELAP` , con el cual se podrá dar máximo provecho a las facilidades que brindan estos códigos.

También quedaron varias propuestas de mejoras para las metodologías desarrolladas en las dos partes del proyecto, a fin de aumentar la robustez de las mismas y la facilidad de interacción con el usuario.

APÉNDICES

A

LÍNEA DE CÁLCULO NEUTRÓNICO DE INVAP S.E.

En esta sección se realiza una breve descripción de los principales programas y componentes de la Línea de Cálculo empleada por INVAP S.E. en el diseño de reactores nucleares. Una presentación más detallada de la misma puede ser encontrada en [5].

A.1 CÁLCULO DE CELDA Y CÁLCULO DE NÚCLEO

La mayoría de las líneas de cálculo para reactores nucleares aprovechan el hecho de que los mismos se encuentran generalmente divididos en estructuras periódicas de elementos combustibles y canales de refrigerantes. Esto permite dividir el cálculo neutrónico en dos pasos: Cálculo de Celda y Cálculo de Núcleo:

CÁLCULO DE CELDA: se lleva a cabo el cálculo del flujo neutrónico para un sistema de dimensiones pequeñas (comparadas con el tamaño total del núcleo) pero con gran detalle espacial y energético. En este paso se emplea la ecuación de transporte neutrónico, la cual deberá ser resuelta por algún método con mucha precisión (Probabilidad de Colisiones, Ordenadas Discretas, Montecarlo, etc.).

CÁLCULO DE NÚCLEO: en este caso el cálculo se realiza para todo el núcleo, pero sin el nivel de detalle empleado en el cálculo de celda. El núcleo se supone dividido en *zonas*, para cada una de las cuales se deberá tener la información de las secciones eficaces homogeneizadas y condensadas a partir de los resultados obtenidos con el cálculo de celda correspondiente a cada zona. En este paso la ecuación de transporte puede ser resuelta con un método menos preciso como el de Difusión Neutrónica.

A.2 DESCRIPCIÓN GENERAL DE LA LÍNEA DE CÁLCULO

La Figura 11 muestra los Línea de Cálculo distintos componentes de la Línea de Cálculo empleada por `INVAP S.E.` y varios de sus clientes en el diseño, optimización y mantenimiento de varios reactores a lo largo del mundo.

La Línea de Cálculo incluye entre otros:

- `CONDOR`: [4], este es el código que realiza el cálculo de celda de los elementos combustibles o cualquier otro componente del núcleo del reactor. `CONDOR` calcula las secciones eficaces homogeneizadas y condensadas que luego serán usadas para el cálculo de núcleo. También permite generar otros parámetros neutrónicos para ser usados por otros códigos.
- `HXS`: (*H*and *XS*): Administrador de bibliotecas de secciones eficaces. Este programa funge como interfaz entre el código de celda y el de núcleo.

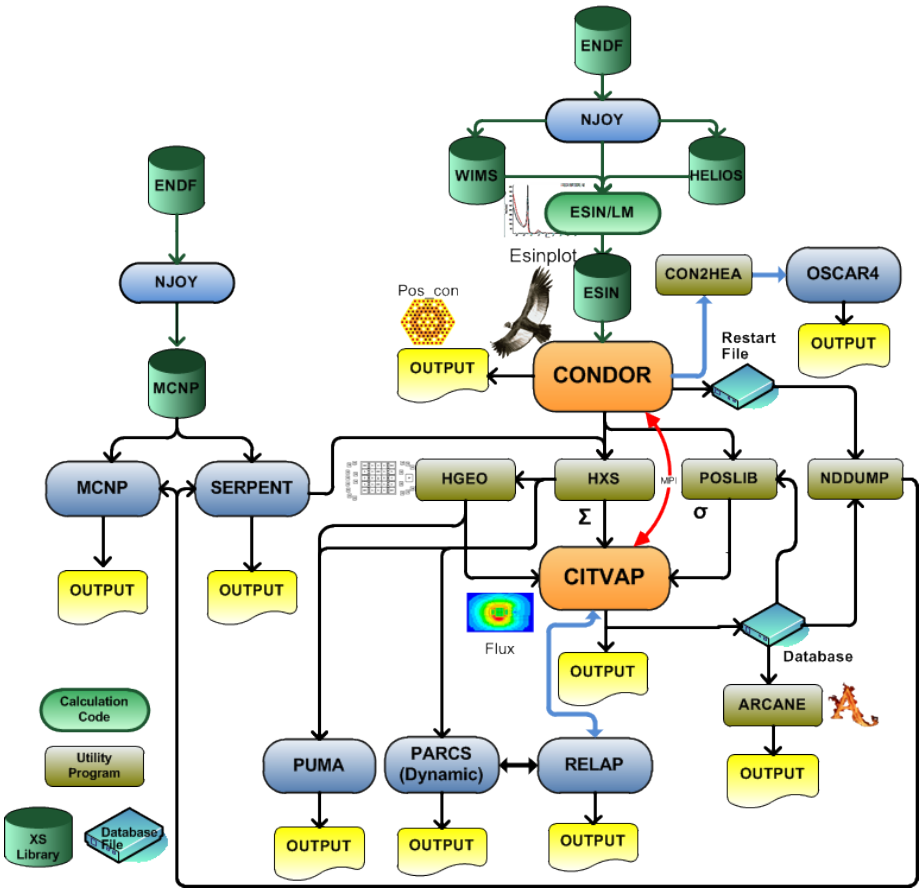


Figura 11: Diagrama de flujo para el ciclo de cálculo de transitorios.

Permite importar secciones eficaces calculadas por distintos códigos de celda y exportarlas a distintos códigos de núcleo.

- `CITVAP`: [1] : Este es el código que realiza el cálculo de núcleo. Este código incorpora gestión de combustibles, cálculo de quemado, influencia de productos de fisión y venenos, entre otras prestaciones. El cálculo de núcleo arrojará los distintos parámetros nucleares que resultan de interés para el análisis del reactor.
- `POSCON` y `FLUX`: post-procesadores gráficos de los resultados de `CONDOR` y `CITVAP` respectivamente. Permiten la visualización de los resultados obtenidos con estos códigos. Son una importante herramienta para el análisis de ingeniería.

A.3 CÓDIGO DE CELDA `condor`

El código de cálculo de celda `CONDOR` emplea los métodos multigrupo de Probabilidades de Colisión y el Método de Respuesta Heterogénea (HRM por sus siglas en inglés) para resolver la ecuación de transporte neutrónico. Este último método implica dividir al sistema en elementos espaciales y resolver la ecuación de transporte de forma acoplada con el método de Probabilidad de Colisiones.

De esta manera CONDOR permite calcular de manera rápida y precisa para geometrías tan complicadas como se quiera. Incluye además distintas opciones para realizar cálculos en los que el estado del componente varía según un determinado parámetro o variable de estado. Como ejemplo, CONDOR permite generar secciones eficaces homogeneizadas y condensadas para distintos estados de quemado, o para distintas distribuciones de temperaturas o densidades.

A.4 CÓDIGO DE NÚCLEO citvap

El cálculo de núcleo es el otro paso clave para el estudio de la física de un reactor. Este cálculo es normalmente realizado para un volumen mucho más grande que en el caso del cálculo de celda, normalmente el núcleo entero y sus alrededores (o una porción del mismo, en caso de existir simetrías).

CITVAP se vasa en código de Difusión CITATION [8] pero incorpora nuevas capacidades y da al usuario una interfaz más amigable y de formato libre. CITVAP puede resolver geometrías de 1 a 3 dimensiones, en coordenadas rectangulares, cilíndricas,triangulares o hexagonales. Permite calcular el flujo neutrónico y el flujo adjunto (útil en el cálculo de parámetros cinéticos).

Para el cálculo con CITVAP se emplea, entre otras herramientas externas, el programa HXS.El mismo importa las secciones eficaces macroscópicas generadas con CONDOR y las almacena en una biblioteca (archivo de extensión .bib) para luego ser empleadas por el código CITVAP. HXS permite realizar varias tareas administrativas con las secciones eficaces macroscópicas. Además, en esta biblioteca se almacenan variables de estado como temperaturas, densidades,concentraciones de Boro, etc.

El archivo de entrada a CITVAP (de extensión *.CII) contiene toda la información necesaria para realizar el cálculo de núcleo y las operaciones deseadas dentro del mismo. La información obtenida con el cálculo se almacena en varios archivos especializados que luego pueden ser empleados para postprocesamiento o cálculos posteriores. Entre ellos, una base de datos (extensión *.CDB) contiene información para las distintas zonas geométricas en que se divide el modelo de núcleo estudiado. Así, de este archivo se puede extraer, para cada zona, información neutrónica (como el flujo y las seccione eficaces) o termohidráulica (como las temperaturas y densidades).

B

ACTIVIDADES RELACIONADAS CON LA PRÁCTICA PROFESIONAL SUPERVISADA

Para llevar a cabo el presente proyecto integrador, se realizaron varias actividades en las oficinas de la División de Ingeniería Nuclear de Invap S.E.

A. Cursado de la Materia de Cálculo y Análisis de Reactores Nucleares, en el Instituto Balseiro. Con esta materia se adquirieron las habilidades para trabajar con los códigos de cálculo neutrónico `CONDOR`,`CITVAP` y`SERPENT` así como programas de pre y postprocesamiento para análisis de datos nucleares. *96 horas*.

B. Estudio introductorio del programa `RELAP` . En esta etapa se adquirieron nociones del trabajo con los archivos de entrada y salida de `RELAP` para el análisis termohidráulico de reactores nucleares. *20 horas*.

C. Estudio del lenguaje de programación `Fortran 95`. Para desarrollar los programas necesarios para llevar a cabo el acople neutrónico-termohidráulico se eligió empleó el lenguaje `Fortran 95` por compatibilidad con los códigos fuentes de `CITVAP` y `RELAP` . *40 horas*.

D. Análisis detallado de la estructura de los archivos de entrada y salida de `CITVAP` y `RELAP` . Debido a que es esencial manipular estos archivos, se requirió comprender su estructura para localizar los datos necesarios para el acople. En particular el archivo de restart para `RELAP` presenta información en formato binario y se debió entender dicho formato para acceder al mismo. *20 horas*.

E. Desarrollo de los programas `relap2citvap` y `citvap2relap`, así como de la biblioteca de rutinas auxiliares `utilities.lib` para llevar a cabo el acople. Esta etapa incluyó múltiples pruebas y ajustes para alcanzar el correcto funcionamiento de los programas integrados al ciclo de cálculo de acople neutrónico-termohidráulico. *100 horas*.

F. Validación del acople. En esta etapa se realizaron múltiples verificaciones del funcionamiento del ciclo y se validaron los resultados obtenidos a través de la comparación del coeficiente de realimentación numérico para el modelo del OPAL con el correspondiente valor experimental. *20 horas*.

G. Estudio de factibilidad de incluir transitorios en el cálculo con `CITVAP` . Estudio y prueba de la sección 026 de `CITATION` para inclusión de fuente externa en el cálculo. *40 horas*.

H. Desarrollo del programa `transient` para el cálculo de transitorios con `CITVAP` . Prueba y verificación del funcionamiento del ciclo de cálculo con modelos neutrónicos simples. *80 horas*.

I. Estudio de propuestas y mejoras a las metodologías diseñadas para avanzar hacia el desarrollo de un *acople dinámico* entre `RELAP` y `CITVAP` .*10 horas*.

J. Escritura de informe del Proyecto Integrador y desarrollo de charlas de pre-defensa y defensa. *100 horas*.

C

DESCRIPCIÓN DE LOS CÓDIGOS FUENTES DE RELAP2CITVAP Y CITVAP2RELAP

En este apéndice se describe el código fuente de los programas `relap2citvap` y `citvap2relap`. Estos programas fueron desarrollados en `Fortran95` y compilados con Lahey/Fujitsu Fortran 95 Express Release 7.20.00. Los correspondientes archivos fuente `*.f95` aparecen listados en el Código 3 y el Código 4 en las páginas 75 y 76 respectivamente. En estos códigos se hace uso extensivo de rutinas que se encuentran en el módulo `utilities.mod`, el cual debe de ser incluido en el proceso de compilación de los programas. El archivo fuente de este módulo aparece listado en el Código 5 de la página 77 , con todas las rutinas que son llamadas desde `relap2citvap.f95` y `citvap2relap.f95`.

Todos los códigos listados al final del apéndice tienen una numeración de líneas, que será empleada para describir las distintas rutinas y bloques a lo largo de las siguientes secciones.

C.1 CÓDIGO FUENTE DE `citvap2relap`

C.1.1 *Definición de variables e inclusión de bibliotecas*

De las líneas 1 a la 4 del Código 3 se encuentra el módulo `WORK_AREA` donde se definen algunas variables a ser empleada dentro del programa. Así mismo, entre las líneas 7 y 20 se encuentran otras definiciones de variables e inclusión de bibliotecas externas.

La biblioteca CDBASE contiene rutinas para la manipulación de la base de datos `Nucleo.CDB` de `CITVAP`. Una guía de los usos de estas rutinas puede ser hallada en [11].

El módulo `UTILITIES` contiene las principales rutinas de los programas `relap2citvap` y `citvap2relap` para la manipulación de los distintos archivos que intervienen en el acople neutrónico termohidráulico.

Los módulos `UTIL` y `IGL_MOD` contienen rutinas misceláneas y son empleadas comúnmente en los programas desarrollados en la División Nuclear de INVAP.

C.1.2 *Lectura de la línea de comandos*

De las líneas 22 a la 40 se inicializan las variables necesarias para la lectura de información desde la línea de comandos.

Un ejemplo de llamada del programa `citvap2relap` desde la línea de comandos de Windows se muestra en el Código 1. Nótese cómo no se escriben las extensiones de los correspondientes archivos.

En el ejemplo listado en el Código 1 no se usa la primera variable `-TIME`, por lo cual esta toma su valor por defecto `' '`. En la versión actual del programa, esta variable aún no tiene uso práctico. Está pensada para en versiones

Código 1: Ejemplo de llamada de `citvap2relap` desde la línea de comandos de Windows

```
1 citvap2relap -FILES:Nucleo -INPUT:LPRR -RELMAP:relaciones
```

futuras incluir información de distintos tiempos de quemado dentro de la base `Nucleo.CDB` de `CITVAP`.

La segunda variable que se pide por línea de comandos es el nombre de la raíz de la base de datos de `CITVAP`. En el ejemplo anterior, se usa el nombre `Nucleo`, por lo que el programa interactuará con la correspondiente `Nucleo.CDB`.

Los nombres de los archivos que se pasen a `citvap2relap` por línea de comando varían de un problema a otro, sin embargo en lo que resta del apéndice usaremos los mostrados en el ejemplo anterior a fin de tener mayor claridad en las explicaciones.

La tercera variable es la raíz del nombre de los archivo de `RELAP`. En el caso del ejemplo anterior, los archivo correspondientes serán `LPRR.r` y `LPRR.i`, en referencia al modelo del reactor egipcio LPRR.

Finalmente se pide el nombre del archivo de *relaciones*. Este archivo no tiene extensión.

En las líneas 38 a la 40 se definen las variables con los nombres de los archivos entrados por línea de comando, ya con las correspondientes extensiones.

C.1.3 *Llamada de rutinas*

De las líneas 41 a la 43 se llaman las rutinas para leer la información necesaria del archivo `Nucleo.CDB` y colocarla en el archivo de entrada a `RELAP` (en el ejemplo aquí tratado `LPRR.i`).

Estas rutinas se encuentran implementadas en el módulo `UTILITIES` y serán descritas en las siguientes secciones.

C.1.4 *Rutina `readRelations(*)`*

Esta rutina es llamada en la línea 41 del Código 3 y aparece desarrollada entre las líneas 38 y 51 del módulo `UTILITIES` listado en el Código 5.

Luego de unas definiciones iniciales entre las líneas 40 y 42, en la línea 43 se abre el archivo de relaciones `relaciones` para ser leído de forma secuencial.

En el Apéndice D, Sección D.3 se especifica el formato requerido por este archivo. En particular el primer número que en él aparece es el número de relaciones que existen. Este número es almacenado en la variable `noOfRelations` en la línea 44. Luego se asignan los espacios necesarios para almacenar los datos en un arreglo de dimensión $4 \times$ `noOfRelations`.

Luego, entre las líneas 46 y 49 el archivo `relaciones` es leído según el formato preconcebido para el mismo, y almacenado en el arreglo pertinente.

En la línea 50 se cierra el archivo `relaciones`.

C.1.5 *Rutina `readCDB(*)`*

Esta rutina es llamada en la línea 42 del Código 3 y aparece desarrollada entre las líneas 162 y 196 del módulo `UTILITIES` listado en el Código 5.

Las definiciones de variables dentro de esta rutina se encuentran en las líneas 165 a la 169.

Entre las líneas 171 a la 175 se realiza una verificación inicial de la existencia de la biblioteca de `CITVAP`, en este caso el archivo `Nucleo.CDB`. En caso de que no exista se cierra automáticamente el programa.

En las líneas 179 y 180 se leen de la base los números de zonas axiales y números de canales, y son almacenados en las variables NAXIAL y NCHANNEL respectivamente.

Las siguientes tres líneas sirven para asignar la memoria suficiente a las matrices `powDens`, `vols` y `powArray` que contendrán las densidades de potencia, los volúmenes de las zonas y la potencia por zona.

En las líneas 186 a la 189 son leídos de la base de datos y almacenados en las correspondientes variables los volúmenes de las zonas (en cm^3) y las densidades de potencia (en mW/cm^3) del modelo de núcleo de `CITVAP`.

Luego, en la línea 193 se calculan las potencias multiplicando las correspondientes densidades por los volúmenes de las zonas. Los valores resultantes son almacenados en el arreglo `powArray`.

Finalmente en la línea 194 se cierra la base de datos `Nucleo.CDB`.

C.1.6 *Rutina `writeInput(*)`*

Esta rutina es llamada en la línea 43 del Código 3 y aparece desarrollada entre las líneas 198 y 252 del módulo `UTILITIES` listado en el Código 5.

Esta rutina hace uso de una biblioteca de manipulación de cadenas denominada `strings`. La biblioteca puede ser descargada libremente de `http://www.gbenthien.net/strings/index.html`. En el mismo sitio se encuentra la documentación necesaria para su manipulación. [1]

Entre las líneas 201 y 217 aparecen las declaraciones de variables que serán usadas para la escritura del archivo de entrada a `text` (en este caso `LPRR.I`).

En la línea 219 se abre el archivo de entrada a `RELAP` para ser leído. En la línea 220 se crea un archivo auxiliar *SCRATCH* que será empleado para contener la versión modificada con las nuevas potencias del archivo de entrada de `RELAP`. Esto es necesario para poder actualizar al archivo de entrada al final de su lectura.

A continuación, entre las líneas 222 y 239 se extiende un ciclo de procesamiento lectura y escritura de los datos del archivo de entrada a `RELAP` (`LPRR.I`). Las operaciones realizadas dentro del ciclo se describen en los siguientes items:

- Al principio de cada etapa del ciclo, se lee una línea del archivo `LPRR.I` y se almacena en la variable `line`. Esto se hace en la línea 223.
- En la línea 224 se hace una verificación de que no se ha alcanzado el final del archivo. Si ese es el caso, se sale del ciclo con la intrucción `exit`.

1 Examples taken from Walter Schmidt's great gallery: `http://home.vrweb.de/~was/mathfonts.html`

- De las líneas 226 a la 236 se extiende un ciclo interno que se repetirá tantas veces como relaciones se hallan encontrado en el archivo `relaciones` (variable `noOfRelations`).
- Para cada etapa del ciclo interno se revisará (línea 229) si la línea actual del ciclo externo (en la variable `line`) contiene a alguno de los números de *Variable de control* localizados en la tercera columna del archivo `relaciones` (y del correspondiente arreglo) . Como se describe en el Apéndice D, Sección D.2, algunas de las líneas de `LPRR.I` deberán tener instrucciones de control de potencia con el formato especificado.

  Cuando se encuentre alguno de los números de *Variable de control* (línea 229), se reescribirá (línea 233) la variable `line` con el formato para variables de control descrito en el D,Sección D.2, pero con la nueva potencia, según la información leída de la base de datos `Nucleo.CDB` y almacenada en el arreglo `powArray`.

  Para acceder al índice correcto del vector de potencias `powArray` se emplea el valor de la cuarta columna del archivo `relaciones` (línea 231). Este índice no es más que el número de zona del modelo `CITVAP` del cual se lee la correspondiente potencia.
- Luego del ciclo interno, en la línea 238 se guarda el valor de la variable `line` en el archivo auxiliar *SCRATCH*. De esta forma, este archivo contendrá la misma información que el archivo de entrada `LPRR.I` a excepción de las líneas con las variables de control, donde la potencia será actualizada a la correspondiente a la base de datos `Nucleo.CDB` almacenada en el arreglo `powArray`.

En las líneas 241 y 242 se rebobinan tanto el archivo de entrada `LPRR.I` como el archivo *SCRATCH* para ser leídos desde el principio.

De las líneas 245 a la 258 se tiene un ciclo en el cual se copia la información del archivo `SCRATCH` al archivo `LPRR.I` para que este último contenga la información actualizada de las potencias por zonas del modelo del núcleo de `CITVAP`.

C.2 CÓDIGO FUENTE DE `relap2citvap`

El archivo fuente `relap2citvap.f95` aparece listado en el Código 4. A continuación se describe el contenido del mismo, si bien en varias ocasiones se hará alusión a las explicaciones dadas en la sección anterior para `citvap2relap.f95` dada la similaridad entre los archivos.

C.2.1 *Definición de variables e inclusión de bibliotecas*

De la línea 1 a la 19 del Código 4 se encuentran las definiciones de las variables necesarias para el programa y las bibliotecas a ser empleadas. Esta sección es similar a la del archivo `relap2citvap.f95` y se describen en la Subsección C.1.1. Las bibliotecas usadas son las mismas que para `relap2citvap.f95`.

C.2.2 *Lectura de la línea de comandos*

De las líneas 22 a la 37 se encuentra la parte del código encargada de preparar la lectura de información de la línea de comandos. La filosofía de programación es la misma que la descritaa en la Subsección C.1.2 para `relap2citvap.f95`.

En el Código 4 se ejemplifica una llamada al programa `relap2citvap` para el problema del `LPRR`.

Código 2: Ejemplo de llamada de `citvap2relap` desde la línea de comandos de Windows

```
relap2citvap -FILES:Nucleo -RESTART:LPRR -RELMAP:relaciones
```

De manera similar al caso de `relap2citvap.f95`, el programa recibe cuatro argumentos por línea de comandos. El primero es el tiempo, y es opcional. El segundo es la raíz del nombre del archivo de base de datos de `CITVAP`. En el caso del Código 1 la base de datos sería `Nucleo.CDB`. El tercer argumento es la raíz del nombre del archivo de `restart` para `RELAP`, en este caso `LPRR.R`. Finalmente se pide el nombre del archivo de relaciones, sin extensión.

C.2.3 *Llamada de rutinas*

De las líneas 40 a la 44 se realizan las llamadas a las rutinas auxiliares, que se enceuntran desarrolladas en el módulo `UTILITIES`. A continuación se describen las distintas rutinas empleadas en el programa.

C.2.4 *Rutina `readRelations(*)`*

Esta rutina fue descrita en Subsección C.1.4. La misma se emplea para leer la información contenida en el archivo de relaciones y cargarla en un arreglo de dimensión $4 \times$noOfRelations.

C.2.5 *Rutina `readRestart (*)`*

Esta rutina es llamada en la línea 41 del Código 3 y se desarrolla entre las líneas 73 y 148 del Código 5 en el módulo `UTILITIES`. La rutina realiza la lectura del archivo binario de restart de `RELAP` (en el ejemplo aquí tratado se denominará `LPRR.R`). Una extensa desripción del formato de este tipo de archivos puede ser encontrado en [2].

De las líneas 74 a la 87 se encuentra la zona de definiciones de las variables internas a la rutina. En particular se define el arreglo constante `heads` que contiene los nombres de algunas variables de interés a ser leídas del restart: el tiempo, la temperatura de la estructura de calor, las temperaturas del fluido, del vapor, la densidad total, la densidad del fluido, la densidad del vapor y la fracción de vacío, en ese orden.

Luego, en la línea 87 se abre el archivo `LPRR.R` de formato binario, para ser leído.

A continuación, en la línea 92 se localiza el record `"plotinf "` con la rutina auxiliar de búsqueda `locateRecord`. Esta rutina se encuentra definida entre las líneas 53 y 71 del módulo `UTILITIEs` y sirve para mover el cursor de lectura del archivo de restart hasta el punto donde encuentre una determinada cadena de caracteres que se le pase como argumento (`recName`).

En las líneas 93 y 94 se lee la información del record `"plotinf "`, a saber, la longitud de los records `plotalf` y `plotnum` que serán leídos más adelante. La variable `Numvar` usada en la línea 95 contiene la cantidad de variables distintas (no todas de interés para `relap2citvap`) que se encuentran en los registros del restart.

Luego en las líneas 96 y 97 se asigna memoria suficiente para contener los nombres de las variables y los correspondientes números de los componentes de `RELAP` a que estas variables caracterizan. Los arreglos empleados para este fin son `alf` y `num` respectivamente.

En la línea 99 se mueve el cursor al siguiente record `"plotalf "` del archivo. Luego, en el ciclo que se encuentra entre las líneas 102 y 112 se realiza un escaneo de este record, guardando los nombres e índices (dentro del record) de las variables de interés que aparecen listadas en el arreglo `heads`. Se emplea un vector auxiliar `IndexVec` para almacenar los índices de las variables de interés, para luego poder acceder a los datos de las mismas con facilidad. Nótese que si bine el arreglo `alf` tiene un tamaño suficiente para contener todas las variables que se enceuntran en los registros del `restart`, sólo contendrá los correspondientes a las variables de interés al programa. En `nOfProperties` se almacenará al final del ciclo la cantidad de variables de interés encontradas, que será menor que `numvar`, o sea, que la cantidad total de variables.

En la línea 114 se localiza el record `"plotnum "` y entre las líneas 117 y 123 se realiza la lectura y guardado de los números de los componentes del modelo de `RELAP` cuyas propiedades se encuentran almacenadas en las variables con nombres guardados en `alf`. Para guardar solamente los números de interés se emplea el arreglo `indexVec` que contiene los índices de los números dentro del record. El truco es que los índices de los nombres en el registro `"plotalf"` deben coincidir con los correspondientes índices de los números en el registro `"plotnum"`. Los números quedan guardados finalmente en el arreglo `num`.

Luego de los ciclos descritos en los párrafos anteriores, los arreglos `alf` y `num` contendrán el nombre y el número de las variables de interés y los correspondientes componentes del modelo de `RELAP`.

Queda leer los valores que irán tomando estas variables para los distintos instantes de tiempo de la simulación que se encuentren listados en el restart. A priori no se puede conocer la cantidad de instantes de tiempos simulados, por lo que se empleará una lista enlazada, de dimensión variable. Los nodos de la lista contienen arreglos para con los datos de las variables de interés. Habrá un nodo en la lista por cada instante de tiempo de simulación que se localice en el `restart`. Para la implementación de la lista se emplean los punteros `first`, `current` y `final` definidos en la línea 9, así como el tipo predefinido `entry` en

la línea 4 para contener cada nodo de la lista que a su vez contienen al arreglo `value` con los valores de las variables al tiempos de interés.

A partir de la línea 124 y hasta el final de la rutina en la línea 148 se realiza la lectura de los valores de las variables de interés, que se encuentran en records de nombre `"plotrec "`.

En la línea 127 se localiza al primero de los `"plotrec"`.

Entre las líneas 129 y 148 se encuentra un ciclo indefinido para la lectura de todos los `"plotrec"`. Con la condicional de la línea 133 se verifica que sólo se almacenen en el arreglo `value` los valores de las variables de interés, cuyos índices se encuentran guardados en el arreglo `indexVec`. Luego en las líneas 138 y 139 se asigna memoria para otro nodo de la lista, y en la línea 140 se localiza el próximo `"plotrec"` para ser leído. Eventualmente se llegará al final del archivo de restart. En ese momento, la condicional de la línea 141 identifiará que no existen más `"plotec"` y saldrá del ciclo indefinido.

C.2.6 *Rutina `writeCDB (*)`*

Esta rutina es llamada en la línea 42 del Código 4 y definida entre las líneas 162 y 196 del Código 5 del módulo `UTILITIES`.

Con ella se escriben los datos necesarios en la biblioteca (en el ejemplo aquí descrito `Nucleo.CDB`) de `CITVAP`. Actualmente la rutina sólo almacena los datos correspondientes al último instante de tiempo leído del archivo `restart` (véase la sección anterior). En la rutina se hace uso extensivo de funciones de la biblioteca de manipulación de base de datos de `CITVAP`, `CDBASE`, la cual se describe en el manual [11].

Entre las líneas 255 y 269 del Código 5 se encuentra la zona de definición de variables.

Entre las líneas 271 y 276 se abre la base de datos `Nucleo.CDB`, o se sale del programa en caso de que la misma no exista.

Entre las líneas 278 y 281 se leen el número de zonas axiales, de canales y de materiales en el correspondiente directorio dentro de la base de datos de `CITVAP`, `Nucleo.CDB`.

En las líneas 291 a la 300 se realiza la lectura de las temperatura del combustible (`'FUELTEMPS'`), temperatura del refrigerante (`'COOLTEMPS'`) y densidad del refrigerante (`'COOLDENSITIES'`). Esta lectura se hace luego de revisar que exista esta información en la base de datos `Nucleo.CDB`.

A continuación, entre las líneas 304 y 321 se desarrolla un ciclo que será repetido tantas veces como variables de interés se hayan encontrado en el restart. Recordemos que los nombres de estas variables se encuentran almacenados en el arreglo `alf` y los correspondientes números de componentes en el arreglo `num`.

Dentro del ciclo, en la línea 305 se llama a la rutina `findInd()` la cual se encuentra desarrollada entre las línea 19 y 36 del Código 5 de `UTILITIES`. Esta rutina almacena en un arreglo `indexes` los índices (número de zona en la cuarta columna del archivo `relaciones`) de los componentes que aparezcan listados con el número `numID` (ya sea en la primera o en la segunda columna) en el archivo `relaciones`. Recordemos que en el arvhico `relaciones` puede aparecer

listado varias veces un mismo componente, refiriéndose por ejemplo a que una misma estructura de calor alimenta a distintos volumenes de pipe en el modelo de RELAP.

Finalmente en las líneas 312 a la 318 se almacenan los valores de las variables requeridas en relaciones en los correspondientes arreglos DCOOL, TCOOL y TFUEL según los índices encontrados con findInd(). Los valores de temperaturas son convertidos a grados Celcius para ser ingresados en la base de datos.

Entre las líneas 324 y 327 se guarda la informaión de los arreglos DCOOL, TCOOL y TFUEL en la base de datos Nucleo.CDB. Finalmente la base de datos es cerrada.

C.2.7 *Rutina eraseData()*

Esta rutina es llamada en la línea 43 del Código 4 y definida entre las líneas 150 y 160 del Código 5 del módulo UTILITIES. Básicamente es una rutina de limpieza, con la cual se libera la memoria dispuesta para la lista enlazada que se utilizó en la lectura del restart.

Código 3: Listado del archivo `citvap2relap.f95`

```
MODULE WORK_AREA
 INTEGER :: NZK
 CHARACTER*64 FILECDB,FILEINPUT,FILERELMAP
END MODULE

program citvap2relap
 USE CDBASE
 USE UTIL
 USE WORK_AREA
 USE IGL_MOD
 USE UTILITIES
 implicit none
 REAL  ATIME
 integer iop,i
 integer,parameter :: MAX_KEY=4
 CHARACTER*6  KWRD(MAX_KEY)
 CHARACTER*64 VALUES(MAX_KEY),KHLP(MAX_KEY)!
 CHARACTER*40 PROGRAM_ID
 real,dimension(:),allocatable:: powArray
 integer::nOfRelations

 KWRD(1)   = ' '
 VALUES(1) = ' '
 KHLP(1) = 'Calculation Time'
 KWRD(2)   = 'FILES'
 VALUES(2) = '../../files/Nucleo'
 KHLP(2) = 'Database for CITVAP'
 KWRD(3)='INPUT'
 VALUES(3)='../../files/LPRR'
 KHLP(3) = 'Input file for RELAP'
 KWRD(4)='RELMAP'
 VALUES(4)='../../files/relaciones'
 KHLP(4) = 'Relations Map File'

 CALL PARVALE(MAX_KEY,KWRD,VALUES,PROGRAM_ID,KHLP,' ')
 READ(VALUES,'(F32.0)') ATIME
 FILECDB = TRIM(VALUES(2))//'.cdb'
 FILEINPUT=TRIM(VALUES(3))//'.i'
 FILERELMAP=TRIM(VALUES(4))
 CALL readRelations(FILERELMAP,nOfRelations)
 CALL readCDB(FILECDB,powArray)
 CALL writeInput(FILEINPUT,powArray,relaciones,nOfRelations)
 print *,"Citvap2Relap ran succesfuly!!!"
end program citvap2relap
```

Código 4: Listado del archivo relap2citvap.f95

```
MODULE WORK_AREA
 INTEGER :: NZK
 CHARACTER*64 FILECDB,,FILERESTART,FILERELMAP
END MODULE
program relap2citvap
 USE WORK_AREA
 USE CDBASE
 USE IGL_MOD
 USE UTIL
 USE UTILITIES
 IMPLICIT NONE
 integer::nOfRelations

 REAL  ATIME
 integer iop
 integer,parameter :: MAX_KEY=4
 CHARACTER*6  KWRD(MAX_KEY)
 CHARACTER*64 VALUES(MAX_KEY),KHLP(MAX_KEY)!
 CHARACTER*40 PROGRAM_ID

 KWRD(1)   = ' '
 VALUES(1) = ' '
 KHLP(1) = 'Calculation Time'
 KWRD(2)   = 'FILES'
 VALUES(2) = 'Nucleo'
 KHLP(2) = 'Database for CITVAP'
 KWRD(3)='RESTART'
 VALUES(3)='LPRR'
 KHLP(3) = 'Relap restart file'
 KWRD(4)='RELMAP'
 VALUES(4)='relaciones'
 KHLP(4) = 'Relations Map File'
 CALL PARVALE(MAX_KEY,KWRD,VALUES,PROGRAM_ID,KHLP,' ')
 READ(VALUES,'(F32.0)') ATIME
 FILECDB = TRIM(VALUES(2))//'.cdb'
 FILERESTART=TRIM(VALUES(3))//'.r'
 FILERELMAP=TRIM(VALUES(4))

 CALL readRelations(FILERELMAP,nOfRelations)
 CALL readRestart (FILERESTART)
 CALL writeCDB(FILECDB,nOfRelations)
 CALL eraseData
 print *, "Relap2Citvap ran succesfuly!!"
end program relap2citvap
```

Código 5: Listado del archivo `utilities.f95`

```
module utilities
 implicit none

 type entry
  real*4,pointer:: value(:) =>null()
  type(entry), pointer::next
 end type entry

 type(entry), pointer :: first, current,final

 integer,dimension(10000) :: indexVec
 character(len=8) , dimension (:), allocatable :: alf
 integer, dimension(:), allocatable ::num
 integer,dimension(:,:),allocatable::relaciones

 integer::nOfProperties

 contains
  subroutine findInd (numID,array,maxRep,indexes,nOfRelations)
   implicit none
   integer,dimension(:,:),allocatable,intent(in)::array
   integer,intent(in) :: numID,maxRep,nOfRelations
   integer,intent(inout)::indexes (maxRep+1)  ! array que contendra los
       indices encontrados
   integer::i,actual

   actual =1
   Indexes(1)=-1

   do i=1,nOfRelations
    if ((array(1,i).eq.numID) .or. (array(2,i).eq.numID)) then
     Indexes(actual)=array(4,i)
     actual=actual+1
     Indexes(actual)=-1
    end if
   end do
  end subroutine findInd

  subroutine readRelations (fileName,nOfRelations)
   implicit none
   character*64,intent(in)::fileName
   integer,intent(out)::nOfRelations
   integer :: i
   open (unit=21, file=trim(fileName),status="old", access="SEQUENTIAL",
       FORM="FORMATTED",ACTION="READ")
   read(21,FMT="(i)") nOfRelations
   allocate(relaciones(4,nOfRelations))
   do i=1,nOfRelations
    read(21,*)  relaciones(1,i),relaciones(2,i),relaciones(3,i),relaciones
        (4,i)
   end do

   close(21)
 end subroutine readRelations
```

```
subroutine locateRecord(fileUnit, recName, position)
 implicit none
 integer, intent(in)::fileUnit
 character(len=8), intent(in)::recName
 integer,intent(inout)::position
 character(len=4):: texto
 integer ::ierr
 do
  read (fileUnit, iostat=ierr) texto
  if (texto .eq. recName(5:8) ) exit
  if (ierr .ne. 0) then
   position = -1
   exit
  end if

  position=position +8
 end do

end subroutine locateRecord

subroutine readRestart (fileNAme)
 implicit none
 character*64,intent(in)::fileNAme
 integer, parameter :: numberOfHeads =8
 character(len=8) , dimension(numberOfHeads), parameter:: heads =(/"time
       ","htvat    ","tempf    ","tempg    ","rho      ","rhof     ","rhog
    ","voidg    "/)

 integer plotalfLength
 integer numvar
 integer plotrecLength
 integer*4 i,j,k,aux,currentPos
 character:: texto*8
 real*4:: dataAux,dataAux1
 integer*1::auxiliar
 integer*4:: auxInt
 integer::IOP
 open (unit=10, file=trim(fileName),status="old", access="TRANSPARENT",
    FORM="UNFORMATTED",ACTION="READ")
 currentPos=0
 read(10)(auxiliar,i=1,4)

 call locateRecord(10,"plotinf ",currentPos)
 read(10) aux, plotalfLength
 read(10) aux, plotrecLength
 numvar= plotalfLength-1
 allocate(alf(numvar))
 allocate(num(numvar))

 call locateRecord(10,"plotalf ", currentPos)
 nOfProperties=0
```

```
 do i=1,numvar
  read(10) texto
  do k=1,numberOfHeads
   if (texto .eq. heads(k)) then
    nOfProperties=nOfProperties+1
    alf(nOfProperties)=texto
    indexVec(nOfProperties) = i
    exit
   end if
  end do
 end do

 call locateRecord(10,"plotnum ", currentPos)
 j=1

 do i=1,numvar
  read(10) aux,auxInt
  if (i .eq. indexVec(j)) then
   num(j)=auxInt
   j=j+1
  end if
 end do
 allocate(current)
 allocate(current%value(numvar))
 first=>current
 call locateRecord(10,"plotrec ",currentPos)

 do
  j=1
  do i=1,numvar
   read (10) dataAux
   if (i .eq. indexVec(j)) then
    current%value(j) =dataAux
    j=j+1
   end if
  end do
  allocate(current%next)
  allocate(current%next%value(numvar))
  call locateRecord(10,"plotrec ",currentPos)
  if (currentPos .eq. -1) then
   final=>current
   exit
  end if
  current=>current%next
 end do
 close(10)
end subroutine readRestart

subroutine eraseData
!Eliminar la memoria usada
 current=>first
 do
  if (.not.associated(current)) exit
  first=>current
  current=>current%next
  deallocate(first%value)
  deallocate(first)
 end do
end subroutine eraseData
```

```
subroutine readCDB(fileNAme, powArray)
 USE CDBASE
 implicit none
 character*32,intent(in)::fileNAme
 REAL, DIMENSION(:), ALLOCATABLE,intent(out) :: powArray
 REAL, DIMENSION(:), ALLOCATABLE :: powDens,vols
 integer:: NAXIAL,NCHANNEL,IOP
 integer:: i

 CALL CDB_INI()
 IOP = CDB_OPEN(30,fileNAme,'OLD')
 IF (IOP.NE.0) THEN
  STOP
 END IF


 CALL CDB_CHANGE_DIR('/CITVAP/MODEL')
 CALL CDB_READ (NAXIAL,'NAXIAL')
 CALL CDB_READ(NCHANNEL,'NCHANNEL')

 ALLOCATE(powDens(NCHANNEL*NAXIAL))
 ALLOCATE(vols(NCHANNEL*NAXIAL))
 ALLOCATE(powArray(NCHANNEL*NAXIAL))

 CALL CDB_READ (vols,'ZONEVOLUMES')

 CALL CDB_CHANGE_DIR('/CITVAP/STEP_0001')
 CALL CDB_READ(powDens,'POWERDENSITIES')


 powArray=vols*powDens

 CALL CDB_CLOSE(30)

end subroutine readCDB

subroutine writeInput(fileNAme,powArray,relArray,nOfRelations)
 use strings
 implicit none
 character*64,intent(in)::fileNAme
 real,intent(in)::powArray(:)
 integer,intent(in)::relArray(:,:)
 integer,intent(in):: nOfRelations
 integer::i,j,ierr
 integer:: dim_relArray
 character(len=80) :: tokens(10)
 integer :: ntokens,indice
```

```
210
    character (len=140) :: line
    character(10)::aux

    type item
215 integer*4:: numvar
    real*4   :: potencia
    end type item

    OPEN(UNIT=10,FILE=fileNAme,FORM="FORMATTED",STATUS="OLD",ACTION="READWRITE
        ")
220 OPEN(UNIT=20,FORM="FORMATTED",STATUS="SCRATCH")

    do
      read(10,'(A)',iostat=ierr) line
      if(ierr .ne. 0) exit
225
      do j=1,nOfRelations
       if (relArray(3,j) .eq. 0) cycle
       call writenum (relArray(3,j),aux,'i10')
       if ( index(line,adjustl(trim(aux))) >0 ) then
230      call parse(line, ' ', tokens, ntokens)
         indice=relArray(4,j)
         call writenum (powArray(indice),aux,'f10.2')
         line = trim(tokens(1))//' '//trim(tokens(2))//' '//trim(tokens(3))//'
              '//aux
         exit
235    end if
      enddo

      write(20,*) trim(line)
    enddo
240
    rewind 20
    rewind 10

     do
245   read (20,'(A)',iostat=ierr) line ! lee cada una de las lineas del
          archivo
      if(ierr .ne. 0) exit ! si encuentra el fin del archivo sal del ciclo
      write (10,*) adjustl(trim(line))
     enddo
     close(10)
250  close (20)

     end subroutine writeInput
```

```
      SUBROUTINE writeCDB (fileName,nOfRelations)
      ! en esta subrutina se modifica el archivo .cdb de CITVAP
255   USE CDBASE
      USE WORK_AREA
      implicit none
      integer,intent(in)::nOfRelations
      character*32,intent(in)::fileName
260   integer,parameter::maxReps=5
      integer ::i,K
      integer,parameter,DIMENSION(maxReps)::indexesNul=(/ (0, i = 1,maxReps) /)
      integer,DIMENSION(maxReps)::indexes
      integer::kode
265
      REAL, DIMENSION(:), ALLOCATABLE :: TFUEL,TCOOL,DCOOL
      INTEGER IOP
      integer::ind
      integer:: NAXIAL,NCAN,MATMAX
270
      CALL CDB_INI()   ! Inicializa la libreria cdbase.
      IOP = CDB_OPEN(30,FILECDB,'OLD')
      IF (IOP.NE.0) THEN
       PRINT *,'Database file: ',FILECDB
275    STOP
      END IF

      CALL CDB_CHANGE_DIR('/CITVAP/MODEL')
      CALL CDB_READ (NAXIAL,'NAXIAL')
280   CALL CDB_READ(NCAN,'NCHANNEL')
      CALL CDB_READ(MATMAX,'MATMAX')

      allocate(TFUEL(MATMAX)) ; TFUEL =0
      allocate(TCOOL(MATMAX)) ; TCOOL =0
285   allocate(DCOOL(MATMAX)) ; DCOOL =0

      CALL CDB_CHANGE_DIR('/CITVAP/STEP_0001')

      kode=0
290   CALL CDB_ARRAY_SEARCH ('FUELTEMPS',kode)
      if(kode .eq. 1) CALL CDB_READ(TFUEL,'FUELTEMPS')

      kode=0
      CALL CDB_ARRAY_SEARCH ('COOLTEMPS',kode)
295   if(kode .eq. 1) CALL CDB_READ(TCOOL,'COOLTEMPS')

      kode=0
      CALL CDB_ARRAY_SEARCH ('COOLDENSITIES',kode)
      print *,kode
300   if(kode .eq. 1) CALL CDB_READ(DCOOL,'COOLDENSITIES')
```

```
 indexes=indexesNul
 do K=1,nOfProperties
  call findInd(num(K),relaciones,maxReps,indexes,nOfRelations)
  ind=1
  do
   if (indexes(ind).eq.(-1)) then
    exit
   end if

   if (alf(K).eq.'rho     ') then
    DCOOL(indexes(ind)) = final%value(K)
   else if (alf(K).eq.'tempf    ') then
    TCOOL(indexes(ind))=final%value(K)-273.16
   else if (alf(K) .eq. 'htvat    ') then
    TFUEL(indexes(ind)) = final%value(K)-273.16
   end if

   ind=ind +1
  end do
 end do

 CALL CDB_SAVE(TFUEL,'FUELTEMPS','FRESH')
 CALL CDB_SAVE(TCOOL,'COOLTEMPS','FRESH')
 CALL CDB_SAVE(DCOOL,'COOLDENSITIES','FRESH')
 CALL CDB_CLOSE(30)

 END SUBROUTINE writeCDB
end module utilities
```

.

D

DESCRIPCIÓN DE ARCHIVOS PARA EL ACOPLE NEUTRÓNICO TERMOHIDRÁULICO

En este apéndice se revisan las estructuras recomendadas para los distintos archivos que intervienen en el ciclo de cálculo del acople neutrónico estacionario, tal como está descrito en la Sección 3.2. Los nombres de los archivos empleados son arbitrarios, y coinciden con los que se usaron durante el desarrollo de la metodología del acople, descritos en la Tabla 1.

D.1 DESCRIPCIÓN DEL ARCHIVO `nucleo.cii`

Como se mencionó en la última parte de la Sección 3.4, existen varias formas de controlar el acople neutrónico termohidráulico, una de las cuales es dirigiéndolo desde el mismo `CITVAP` . Este fue el mecanismo que se usó en el proyecto, sin embargo, acomodar la metodología para emplear un script externo no presenta ninguna complicación.

En la Figura 12 se muestra una parte del archivo `Nucleo.CII`, en particular aquella donde se lleva a cabo efectivamente el acople neutrónico termohidráulico con las llamadas a programas externos.

D.2 DESCRIPCIÓN DEL ARCHIVO `lprr.i`

En el archivo `LPRR.I` se especifica el modelo termohidráulico del núcleo del reactor y generalmente de la planta como un todo. Normalmente este tipo de archivo es extenso: leerlos y entenderlos requiere de mucha experiencia con el uso de `RELAP` o de consultas frecuentes al manual (véase [2]).

Si bien para llevar a cabo el acople se permitirá cierta libertad en la descripción del modelo del núcleo y la planta, sí existen requerimientos para que la metodología de acople planteada en el Capítulo 2 sea efectiva.

En particular, la potencia a las `heat structures` deberá ser pasada con variables de control.

En la Figura 13 se muestra una porción del archivo `LPRR.I` que cumple con dichos requerimientos. Se distingue una zona donde se especifican los `pipes`, otro para las `heat structures` y finalmente una zona para las variables de control de potencia.

Los programas `relap2citvap` y `citvap2relap` esperan un formato para la zona de variables de control tal como el especificado en la Figura 13.

D.3 DESCRIPCIÓN DEL ARCHIVO `relaciones`

Con el archivo `relaciones` los programas `relap2citvap` y `citvap2relap` hacen posible el traspaso de información entre zonas equivalentes de los modelos

de núcleo de CITVAP (especificado en Nucleo.CII) y de RELAP (especificado en LPRR.R).

En la Figura 14 se muestra una porción del archivo relaciones ejemplificando cómo se conforma el mismo.

La primera línea del archivo contiene el número de relaciones a leer. Luego aparecen listadas estas relaciones en líneas de cuatro columnas.

Cada línea expresará la relación entre un zonas de pipes,heat structures, variables de control de potencia y zonas de CITVAP.

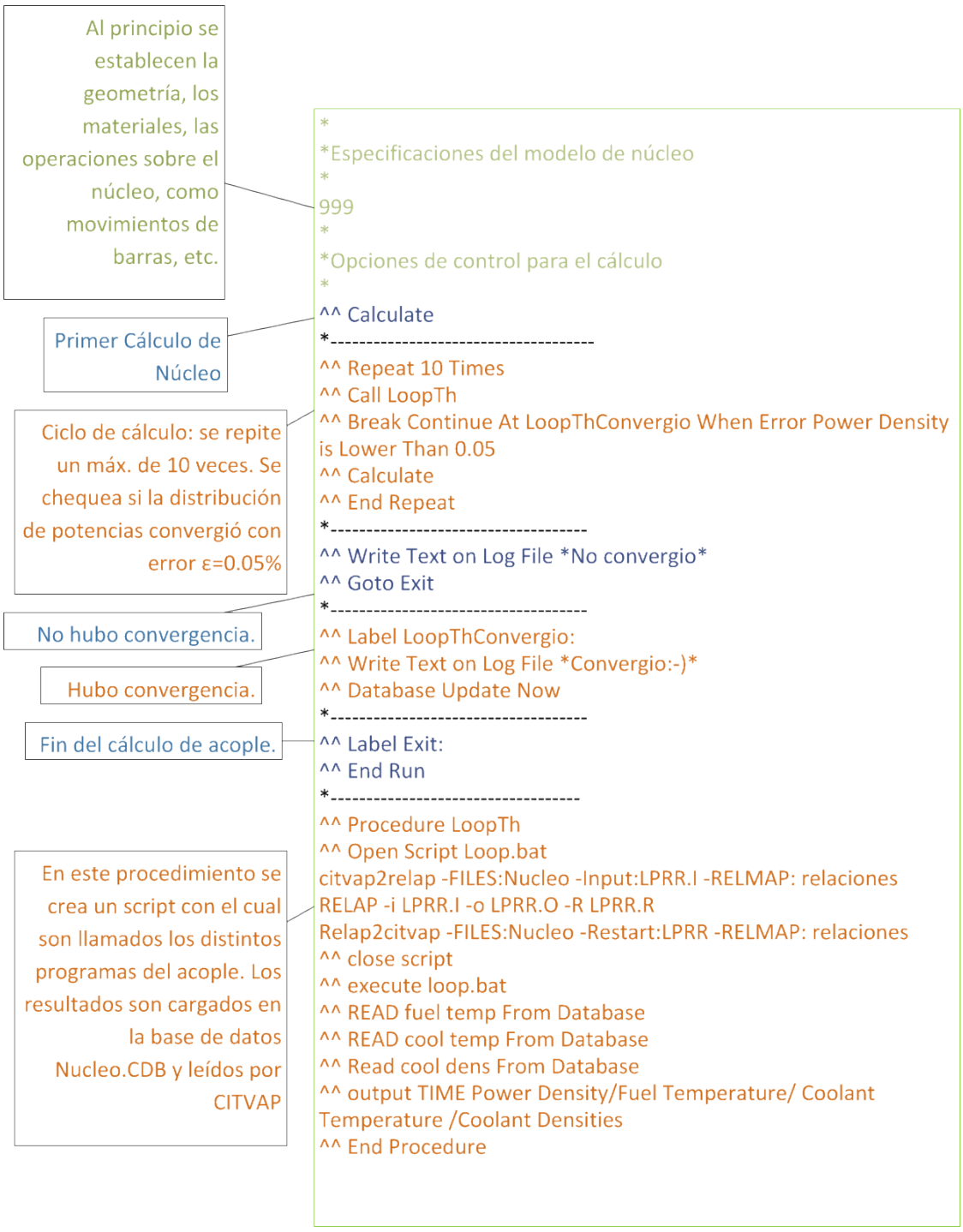


Figura 12: Ejemplo de archivo `Nucleo.CII` con las líneas de control necesarias para llevar a cabo el acople *desde* `CITVAP` .

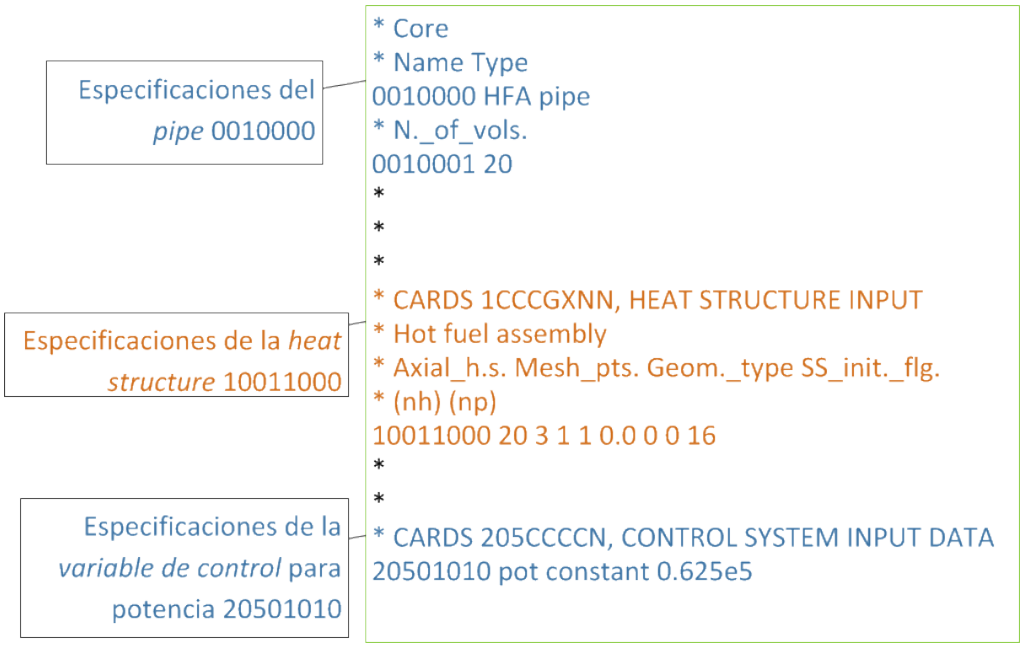


Figura 13: Parte del archivo `LPRR.I`

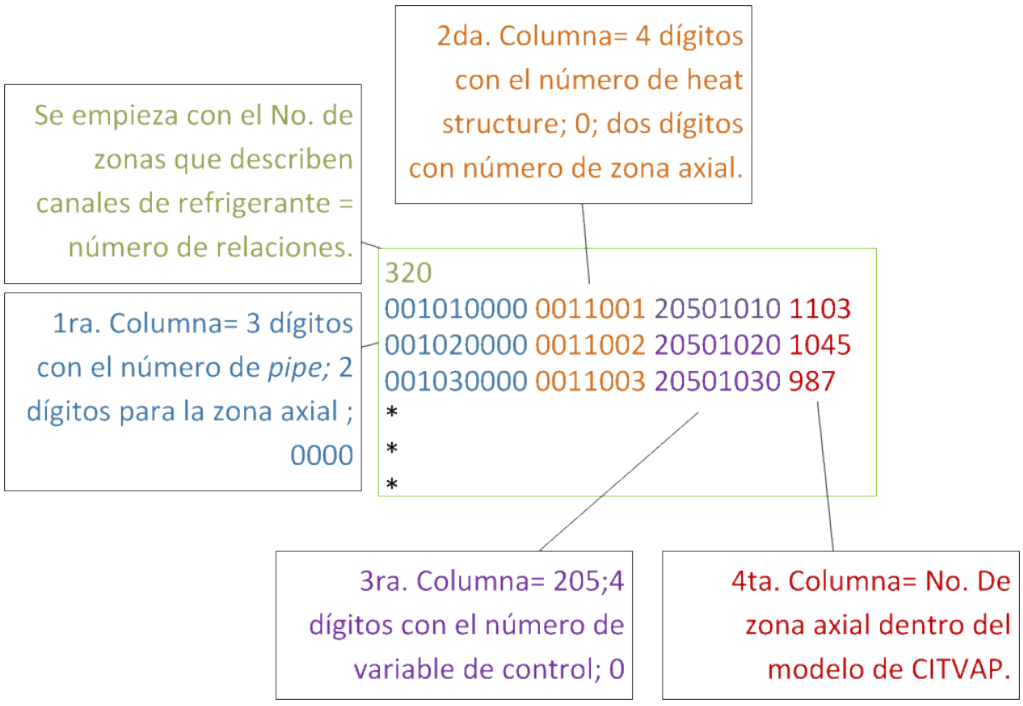


Figura 14: Parte del archivo `relaciones`.

E

RESOLUCIÓN DE PROBLEMAS CON FUENTE EXTERNA CON CITVAP-CITATION

En este Apéndice se describe el uso de fuentes externas en el modelado de sistemas neutrónicos con CITVAP-CITATION.

La metodología planteada en la Sección 6.3 para resolver transitorios con CITVAP-CITATION incluyen el uso de una fuente externa equivalente $S'_{r,g,t}$.

En INVAP no se ha tenido mucha experiencia previa en el uso de fuentes externas con CITVAP-CITATION. Por lo tanto parte de la resolución de transitorios fue encontrar la posibilidades y limitaciones al resolver este tipo de problemas con CITVAP-CITATION.

El problema de fuente externa puede ser incluido en el cálculo neutrónico con CITVAP-CITATION usando la sección 026 del código CITATION. Una descripción minuciosa de la sección 026 aparece en el manual de este código (véase [8]). Aquí sólo se mostrarán las características competentes al uso de fuente externas en la resolución de transitorios.

E.1 INCLUSIÓN DE LA SECCIÓN 026 EN EL ARCHIVO DE ENTRADA A CITVAP

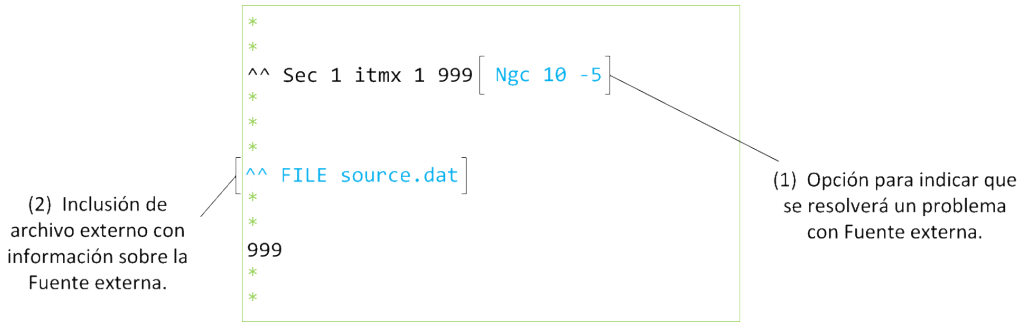


Figura 15: Ejemplo de archivo Nucleo.cii preparado para resolver un problema de Fuente externa.

En la Figura 15 se muestra un un fragmento de archivo de entrada a CITVAP (denominado genéricamente Nucleo.cii) en el que se indica que se resolverá un problema de fuente externa. El código ajeno a las especificaciones de Fuente externa se ha omitido por cuestiones de claridad.

Para indicar que se resolverá un problema de fuente externa, debe asignarse a la variable NGC10 el valor -5 en la sección 1 del archivo de entrada CITVAP nucleo.cii. Esto se ve en el cuadro (1) de la Figura 15.

La información de la intensidad de la fuente externa y su distribución energética puede ser incluida directamente en el archivo Nucleo.cii. Otra alternativa es escribir esta información en un archivo externo (en lo adelante source.dat) y usar el comando de CITVAP FILE para integrar su contenido al de Nucleo.cii.

Esta vía fue la empleada en el cálculo de transitorios que se describe en este trabajo. En el cuadro (2) de la Figura 15 se muestra la orden correspondiente para la inclusión de `source.dat`.

E.2 ESPECIFICACIÓN DE LA FUENTE EXTERNA EN EL ARCHIVO `source.dat`

En la Figura 16 se muestra el contenido de un archivo con datos de fuente externa denominado `source.dat`. Se eligió describir una fuente externa para un modelo de `CITVAP` de una sola zona y a dos grupos para facilitar la explicación.

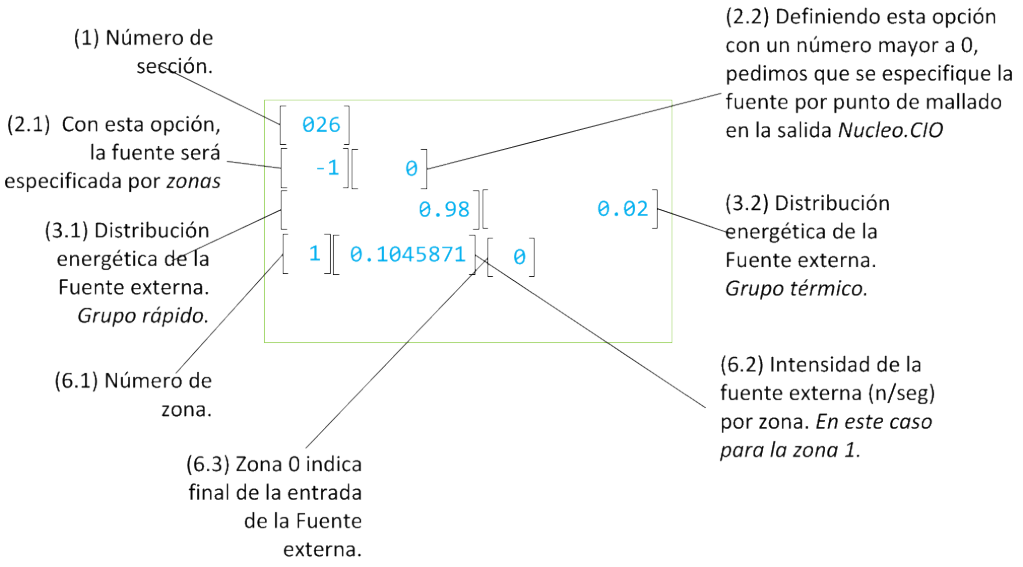


Figura 16: Ejemplo de archivo `source.dat` con información necesaria para un problema de Fuente externa con `CITVAP-CITATION`.

Según el formato exigido por `CITATION`, la información de la sección 026 se brinda en tarjetas. Cada tarjeta se comprende de al menos un campo numérico. Los campos numéricos tienen algún formato que limita la cantidad de espacios empleados y el tipo de número que permite.

A continuación se describen las tarjetas empleadas en el cálculo de transitorios para describir la fuente externa.

E.2.1 *Tarjeta 1:* 026. *Formato* `(I3)`

Se encabeza el código de la sección precisamente con los dígitos 026 (Estrictamente en las tres primeras columnas de la línea). Podemos verlo en el cuadro (1) de Figura 16.

E.2.2 *Tarjeta 2: Opciones. Formato* `(2I3)`

El primer campo de esta tarjeta se extiende por tres espacios. En él se especifica un número para establecer si los valores de intensidad de fuente externa serán introducidos por zona o por volumen de mallado. En los estudios realizados en este trabajo, se decidió que especificar la fuente externa por zona es

suficientemente preciso. Para este caso, el valor a colocar en este campo es -1, tal como se ve en el cuadro (2.1) de la Figura 16.

El segundo y último campo de la Tarjeta 2 (también de tres espacios) contendrá el valor 1 para establecer que se desea la información de la fuente externa por volumen de mallado a la salida de `CITVAP` (archivo Nucleo.cio) o 0 si esto no es necesario.

Dadas las opciones elegidas para la tarjeta 2, se requiere solamente colocar la información de las tarjetas 3 y 6 en las siguientes líneas.

E.2.3 *Tarjeta 3: Espectro energético de la fuente externa. Formato `(6E12.0)`*

El formato especificado para los datos a llenar en esta tarjeta es `6E12.0`, O sea, hasta 6 números reales en notación exponencial cada uno ocupando 12 espacios.

Los números especificados indican la fracción de los neutrones de la fuente externa que aparecerán en cada grupo de energía, empezando con el grupo de más energético. La suma de estos números debería ser 1, aunque son normalizados a la unidad por el código.

En los cuadros (3.1) y (3.2) de la Figura 16 se especifica el espectro energético para el caso de un problema a dos grupos.

Es importante destacar que el código sólo permite especificar un espectro energético general para la fuente externa, independientemente de la región. Esto será un problema con el cual se deberá lidiar para resolver transitorios con ayuda de la fuente externa.

La sección 026 sólo permite especificar un espectro energético universal para la fuente externa, independientemente de la posición.

E.2.4 *Tarjeta 6: Intensidad de fuente externa por zona.Formato `6(I3,E9.0)`*

El formato especificado para los datos a llenar en esta tarjeta permite al menos 6 campos de dos números cada uno, por línea. Cada campo contendrá un entero que ocupe tres espacios especificando el número de zona y un real que ocupe nueve espacios especificando la fuente externa en esa zona en neutrones/segundos.

El espacio permitido para especificar el número de zona resulta demasiado limitante: sólo tres lugares. Actualmente los modelos nucleares que se desarrollan incluyen mucho más de 999 zonas.

Por otro lado, es para especificar la intensidad de la fuente externa en cada zona se permiten solamente nueve espacios.Si en la zona 1 se producen $1{,}045871 * 10^{10} n/s$, la forma más eficiente para especificarlo en la tarjeta 6 es escribiendo:

$$0011.0459 + 10$$

O sea, en los primeros tres espacios se especifica el número de zona (001) y en los siguientes nueve espacios la intensidad de la fuente en neutrones/segundo. Se deberán dedicar como mínimo tres espacios a la parte del exponente (también se podría escribir $E + 10$, pero serían ya cuatro espacios). Esto dejará

solamente seis espacios (incluyendo el punto decimal) para especificar la base del número.

En la resolución de problemas de fuente externa aplicados al cálculo de transitorios, resulta que sólo cinco dígitos para especificar la fuente externa permite un resultado con limitada precisión.

En definitiva, las limitaciones de espacio para los datos de la Tarjeta 6 deberán ser tratadas modificando el código fuente de `CITATION` a fin de poder estudiar transitorios con mayor precisión en los resultados y más alcance en el número de zonas disponibles.

E.2.5 *Final de la entrada de datos*

Para indicar que se terminó la entrada de datos en la sección 6 se escribe el número 0 como número de zona. En el ejemplo mostrado en la Figura 16, el modelo cuenta solamente con una zona, por lo que directamente después de los campos correspondientes a la misma, se introduce el número 0. Así se ve en el cuadro (6.3) de la Figura 16.

F

DESCRIPCIÓN DEL CÓDIGO FUENTE DE TRANSIENT

En este Apéndice se describe el contenido del archivo `Transient.f95` para facilitar futuras modificaciones del mismo. El archivo contiene código programado en `Fortran 95`. En el mismo se emplean las bibliotecas `UTIL.lib` y `CDBASE.lib` compiladas con Lahey/Fujitsu Fortran 95 Express Release 7.20.00. Estas bibliotecas contienen subrutinas para manipulación de textos y bases de datos de `CITVAP`. Un listado del archivo se puede encontrar en el Código 7, al final de este Apéndice. Las siguientes descripciones del código fuente se basarán en los números de líneas listados en este código.

F.1 DECLARACIÓN DE VARIABLES

De las líneas 6 a la 40 se definen las distintas variables que serán empleadas dentro del código. Estas se encuentran separadas por bloques, según la función que cumplen dentro del código.

F.2 LECTURA DE LA LÍNEA DE COMANDOS

De las líneas 42 a la 72 se realiza la lectura de argumentos del programa de la línea de comando.

Un ejemplo de cómo llamar al programa`Transient` se muestra en el Código 6

Código 6: Ejemplo de llamada del programa `Transient` desde la línea de comandos de Windows

```
./Transient -TIME:0001 -NU_DATA:NuclearData.dat -FILECDB:Nucleo.cdb -
    EXTSOURCE:source.dat -PRECURSORS:precursors.dat
```

La primera variable que se pide por línea de comandos es el tiempo en formato entero de cuatro dígitos (aún con ceros a la izquierda). Este parámetro resultará útil cuando sea integrado con el código fuente de `CITVAP`, para almacenar la información en los distintos directorios con que cuenta la base de datos de `CITVAP` (`Nucleo.cdb`).

Los demás cuatro argumentos son los nombres del archivo de datos nucleares, el archivo de la base de datos de `CITVAP`, el archivo de fuente externa y el archivo para almacenar la concentración de precursores. En el Código 6 y en lo que queda de este Apéndice estos archivos serán denominados con los nombres genéricos `NuclearData.dat`, `nucleo.cdb`, `source.dat` y `precursors.dat` respectivamente. Véase una descripción de las funciones que cumplen estos archivos en la Tabla 8.

Antes de cada cálculo de transitorios los archivos `source.dat` y `precursors.dat` *Al inicio de un ciclo de cálculo de transitorios los archivos `source.dat` y `precursors.dat` no deben existir en disco.*

no deben existir en disco. De esta forma se informa a `Transient` que el sistema se encuentra en su estado inicial de criticidad (antes del transitorio). `Transient` creará estos archivos automáticamente después de la *primera corrida*, con los nombres que se le pase como argumentos.

F.3 LLAMADA DE RUTINAS AUXILIARES

De las líneas 83 a 88 se llama a las distintas rutinas auxiliares de las que se compone el programa.

F.3.1 *rutina* `readNuclearData()`

La rutina `readNuclearData()` se encuentra definida entre las líneas 92 y 120.

En ésta se lee el contenido de `NuclearData.dat`. Se espera de este archivo que contenga información esencial para el cálculo de transitorios en un formato especificado. Este formato viene especificado con comentarios entre las líneas 75 y 82 del Código 6.

En la Figura 17 se muestra el contenido de un archivo de datos nucleares de ejemplo. En este caso se trata de un problema de tres grupos de energía, seis grupos de precursores y para un modelo de sólo una zona. En la práctica los modelos de núcleo reales tienen muchas más zonas, el archivo `NuclearData.dat` será más complejo, pero seguirá el mismo formato.

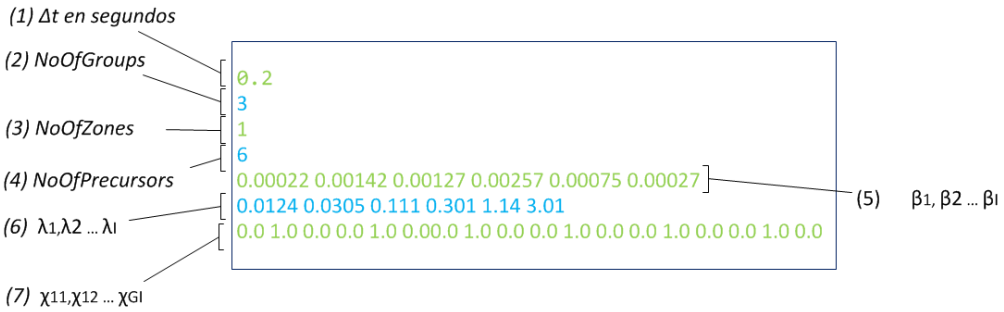


Figura 17: Ejemplo de archivo `NuclearData.dat` con para el caso de un problema de una zona, a tres grupos de energía y seis grupos de precursores.

A continuación se describe el contenido de cada una de las líneas del archivo `NuclearData.dat` tomando como apoyo la Figura 17.

- En la línea (1) se coloca el paso de tiempo con el que se realizará el cálculo (Δt) en formato `REAL*4`.
- En la línea (2) el entero `NoOfGroups` para el número de grupos de energía en que se dividirá le flujo neutrónico.
- En la línea (3) el entero `NoOfZones` para la cantidad de regiones en que se divide el modelo de núcleo en el archivo de entrada a `CITVAP` `Nucleo.cii` (la discretización espacial del flujo neutrónico y la fuente externa se hará según este número de regiones (zonas)).

- En la línea (4) el entero `NoOfPrecursors` para la cantidad de precursores con la que se realizará el cálculo (usualmente 1 ó 6).
- En la línea (5) se encuentran las fracciones β_i con i variando desde 1 hasta I =`NoOfPrecursors`. Tienen formato `REAL*4`.
- En la línea (6) se toman los valores de λ_i con i variando de 1 a I =`NoOfPrecursors`. Tienen formato `REAL*4`.
- Finalmente en la línea (7) se toman los valores de $\chi_{i,j}$, o sea los espectros de fisión de cada uno de los $i = 1..$`NoOfGroups` grupos de precursores para cada uno de los $j = 1..$`NoOfPrecursors` grupos de energía. Tienen formato `REAL*4`.

En la rutina `readNuclearData()` se asigna la memoria necesaria para los arreglos multidimensionales para contener los datos nucleares leídos de `NuclearData.dat`.

F.3.2 *Rutina `readCDB()`*

La rutina `readCDB()` lee el flujo neutrónico $\phi_{r,g,t}$ y las nuFisiones $\nu\Sigma_{f,r,g}$ de la base de datos `Nucleo.cdb` del cálculo con `CITVAP`. Se encuentra entre las líneas 122 y 145.

Esta función usa rutinas de la biblioteca `CDBASE` para manipulación de bases de datos de `CITVAP`. Antes de la lectura, hace un cambio al directorio correspondiente al tiempo que se solicita en la línea de comandos, dentro de la base de datos. Esto lo hace con la rutina `locateDirAtTime()`.

Para leer el flujo $\phi_{r,g,t}$ (por zona y por grupo de energía) y las nuFisiones $\nu\Sigma_{f,r,g}$ (también por zona y grupo de energía) se emplean arreglos de dos dimensiones ordenados de forma que se asigne correctamente los valores a los índices de los arreglos durante la lectura de la base de datos.

Finalmente se lee la concentración de precursores a través de la rutina `readPrecursors()` la cual se describe en la siguiente sección.

F.3.3 *Rutina `readPrecursors()`*

Esta rutina se encuentra definida entra las líneas 157 y 180.

Primeramente se revisa si existe el archivo `Precursors.dat`. De no existir, se entiende que el sistema se encuentra en equilibrio (criticidad inicial) y en ese caso se asigna `FALSE` a la variable booleana `NotEquilibrium`.

En la línea 172 se toma una decisión (con una estructura condicional `IF`) según si el sistema se encuentra o no en equilibrio inicial.

Si el sistema se halla en equilibrio, con el ciclo de las líneas 175 a 181 se calcula la concentración incial de equilibrio de los precursores según la Ecuación 9 , empleando los datos nucleares ya leídos del archivo `NuclearData.dat`. Los datos calculados son almacenados en un archivo nuevo con el nombre `precursors.dat`. Creando este archivo, se asegura que para la próxima corrida del programa, se pase a cálculos correspondientes a transitorios y no a equilibrio con criticidad.

Si ya el sistema no se encuentra en su estado de equilibrio, se espera que los valores de $\nu\Sigma_{f,r,g}$ leídos de `Nucleo.cdb` estén ya multiplicados por $(1-\beta)$.

Si el sistema no se halla en equilibrio, entonces el procedimiento a seguir es distinto. Primeramente, se presupone que las nuSigmaF leídas con la función readCDB no son realmente los valores de $\nu\Sigma_{f,r,g}$ sino que se encuentran afectados por un $(1-\beta)$, o sea, los valores leídos en realidad son $(1-\beta)\nu\Sigma_{f,r,g}$. Recordemos que al usuario se le exige que cambie los valores de las nuFisiones en este factor para que el cálculo sea correcto, tal como se describió en la Sección 7.4. en las líneas 172 y 173 se calculan los valores reales de $\nu\Sigma_{f,r,g}$ dividiendo los valores leídos de la base de datos por $(1-\beta)$.

Luego se lee los valores de concentración de precursores del archivo `Precursors.dat`.

F.3.4 *Rutina `CalculateC()`*

Esta rutina se encuentra entre las líneas 182 y 197.

En esta rutina se calcula la concentración de precursores $C_{i,r,t+\Delta t}$ según los valores de $\phi_{r,g,t}$ leídos de `Nucleo.cdb` y los valores de $C_{i,r,t}$ leídos de `precursors.dat` . Se emplea la Ecuación 10. Los valores resultantes son almacenados en el archivo `precursors.dat`, aplastando a los anteriores.

F.3.5 *Rutina `calculateQ()`*

Esta rutina se encuentra entre las líneas 202 y 212.

En este caso se emplea la Ecuación 12 para calcular la fuente externa equivalente $S'_{r,g,t+\Delta t}$ a ser usada en la Ecuación 14. Luego se llama a la rutina `WriteQ()` que se encarga de escribir el archivo `source.dat`.

Esta rutina emplea el formato específico exigido por el manual de CITATION (véase [8] sección 026) para describir un problema de fuente externa (ver Apéndice E). De esta forma, se puede directamente incluir el archivo `source.dat` en el archivo de entrada a `CITVAP` `Nucleo.cii` con el comando `FILE` tal como está escrito para incluir el cálculo de fuente externa.

Aquí debe destacarse que la sección 026 exige que se pase un espectro de energía para la fuente externa, sin embargo no da la posibilidad de que este espectro varíe de una región a otra. Esto es un problema, ya que en el caso de cálculo de transitorios el espectro de la fuente equivalente $S'_{r,g,t}$ calculada según la Ecuación 14 deberá variar de una zona a otra ya que depende de la concentración de precursores, que varía a su vez con la zona espacial r donde se calcule.

En la metodología de cálculo de transitorios aquí presentada se usa el espectro correspondiente a la primera de todas las zonas $r = 1$ como espectro común a todas las zonas. Esto demostró no introducir errores notables en la verificación del código.

F.3.6 *Rutina `outputResults()`*

Esta rutina se encuentra entre las líneas 225 y 232.

Con `outputResults()` se escriben en el archivo `results.rlt` los resultados del flujo neutrónico ϕ_{r,g,t_n}, la concentración de precursores C_{i,r,t_n} y la fuente

externa equivalente S'_{r,g,t_n} para cada tiempo de cálculo t_n, . Los resultados son agregados al final del archivo después de cada paso de tiempo.

Actualmente la rutina está programada para escribir solamente los resultados correspondientes al problema de una zona , un grupo de energía y un grupo de precursores. No sería complicado extender su funcionalidad para problemas de más grupos de energía, más cantidad de precursores y más zonas.

F.4 OBSERVACIONES FINALES

El programa `Transient` y su código fuente necesariamente deberá ser modificado a fin de incluir nuevas capacidades:

- Actualmente las constantes neutrónicas, almacenadas en los arreglos `beta(:),lambda(:),chi(:,:)` sólo son especificadas por grupo de energía y/o por grupo de precursor, según corresponda. Agregándole una dimensión más a estos arreglos y cambiando los bloques de código correspondientes a la manipulación de los mismos, se podría incluir la capacidad de que también sean especificadas por zona, permitiendo así una variación espacial de estas constantes.
- Las variables `NoOfGroups` y `NoOfZones` actualmente pueden ser extraídas de la base de datos `Nucleo.CDB` por lo que no hay necesidad de que sean introducidas en el archivo `NuclearData.dat`. Deberán hacerse las modificaciones pertinentes en el código para obtenerlas de la base de datos. Eventualmente la variable `NoOfPrecursors` también estará contenida en la base de datos y podrá ser obtenida de la misma manera.

Código 7: Listado del archivo `Transient.f95`

```
program Transient
 use UTIL
 use CDBASE
 implicit none

 !Variables para la entrada por linea del comando
 integer, parameter:: MAX_KEY=5
 CHARACTER*12  KWRD(MAX_KEY)
 CHARACTER*64 VALUES(MAX_KEY),KHLP(MAX_KEY)
 CHARACTER*64 FILEDAT,FILECDB,FILESRC,FILEPREC
 integer TIME !time
 CHARACTER*40 PROGRAM_ID
 integer IOP !variable para flag de error
 logical NotEquilibrium

 !Variables entradas por el usuario del programa
 Real Dt
 integer NoOfGroups
 integer NoOfZones
 integer NoOfPrecursors
 Real,allocatable:: beta(:)
 Real,allocatable:: lambda(:)
 Real,allocatable:: chi(:,:)


 !Variables para entrada por archivo CDB
 Real,allocatable:: NuSigmaF(:,:)
 Real,allocatable:: Flux(:,:)


 !Variables auxiliares
 Real,allocatable:: dCdt(:,:)              !variable auxiliar para
     calcular el c(:,:)
 REAL,allocatable:: C(:,:)
 REAL,allocatable:: C0(:,:)
 Real,allocatable:: q(:,:)              !Fuente externa
 Real,allocatable:: qTotal(:)  !Fuente externa total (sumada a todos los
     grupos) por zona
 real :: betaTotal

 !iteradores
 integer prec,zone,group,i

 ! Lectura de la linea de comandos

 KWRD(1)   = 'TIME'
 VALUES(1) = '0001'
 KHLP(1) = 'TIME'

 KWRD(2)   = 'NU_DATA'
 VALUES(2) = 'NuclearData.dat'
 KHLP(2) = 'Nuclear Data needed for calculations'
```

```
KWRD(3)   = 'FILECDB'
VALUES(3) = 'base.cdb'
KHLP(3) = 'CITVAP Database for flux and cross sections'


KWRD(4)   = 'EXTSOURCE'
VALUES(4) = 'source.dat'
KHLP(4) = 'Holder for section 026 CITVAP-external source'

KWRD(5)   = 'PRECURSORS'
VALUES(5) = 'precursors.dat'
KHLP(5) = 'Holder for precursors concentrations'


CALL PARVALE(MAX_KEY,KWRD,VALUES,PROGRAM_ID,KHLP,' ')
READ(VALUES(1),*) TIME
FILEDAT = TRIM(VALUES(2))
FILECDB = TRIM(VALUES(3))
FILESRC = TRIM(VALUES(4))
FILEPREC = TRIM(VALUES(5))
!---------------------------------------
!---Formato esperado para el archivo NuclearData.dat
!1: Dt
!2: NoOfGroups
!3: NoOfZones
!4: NoOfPrecursors
!5: beta(1) beta(2)... beta(NoOfPrecursors)
!6: lambda(1) lambda(2) ... lambda(NoOfPrecursors)
!7: chi(1;1) chi(1;2)  chi(1;3)....chi(2;1)......chi(NoOfPrecursors,
    NoOfGroups)
call CDB_INI()
call readNuclearData()
call readCDB()

call calculateC() ! calcula la concentracion final de precursores (igual
    a la inicial si sistema critico(equilibrio))
call calculateQ() ! calcula la fuente externa
call outputResults()

contains

subroutine readNuclearData()
 implicit none
 !lectura del archivo de datos nucleares
 open (unit=10, file=trim(FILEDAT),status="old", access="SEQUENTIAL",FORM
     ="FORMATTED",ACTION="READ")
 read(10,*) Dt
 read(10,FMT="(i)") NoOfGroups
 read(10,FMT="(i)") NoOfZones
 read(10,FMT="(i)") NoOfPrecursors
 allocate(beta(NoOfPrecursors))
```

```
 allocate(NuSigmaF(NoOfZones,NoOfGroups))
 allocate(Flux(NoOfZones,NoOfGroups))
 allocate(lambda(NoOfPrecursors))
 allocate(chi(NoOfPrecursors,NoOfGroups))
 allocate(C0(NoOfPrecursors,NoOfZones))
 allocate(C(NoOfPrecursors,NoOfZones))
 allocate(dCdt(NoOfPrecursors,NoOfZones))
 allocate(q(NoOfZones,NoOfGroups))
 allocate(qTotal(NoOfZones))

 read(10,*) (beta(i), i=1, NoOfPrecursors)

 read(10,*) (lambda(i), i=1, NoOfPrecursors)

 read(10,*) ( (chi(prec,group), group=1, NoOfGroups) , prec=1,
     NoOfPrecursors)

 close(10)
end subroutine readNuclearData

subroutine readCDB()
 implicit none
 !lectura de la base de datos CDB de CTIVAP, se leen el flujo y las
     nufisiones

 IOP = CDB_OPEN(30,FILECDB,'OLD')
 IF (IOP.NE.0) THEN
  PRINT *,'Database file: ',FILECDB
  STOP
 END IF

 CALL locateDirAtTime ()
 CALL CDB_READ(Flux,'FLUX')
 ! si existe 'PRECURSORS.dat' lee PRECURSORS
 ! de lo contrario llama a la rutina calculateEqPrecursors
 CALL CDB_CHANGE_DIR('./CROSS_SECTIONS')
 CALL CDB_READ(NuSigmaF,'NUFISSION')

 CALL CDB_CHANGE_DIR('..')
 CALL readPrecursors()
 !se presupone que las nuFisiones ya estan multiplicadas por (1-beta)
 !entonces se divide las nuSigmaF por 1-beta, para trabajar
 !en lo adelante con las nuSigmaF reales.

end subroutine readCDB

subroutine locateDirAtTime()
implicit none
character(len=4) timeString
write(timeString,"(I4.4)") TIME

CALL CDB_CHANGE_DIR('/CTIVAP/STEP_' // timeString  )
end subroutine locateDirAtTime
```

```
subroutine readPrecursors()
 implicit none
 inquire (file=trim(FILEPREC),exist=NotEquilibrium)
 if (NotEquilibrium .eqv. .FALSE.) then ! si el sistema esta en
     equilibrio
  ! calcula los precursores de equilibrio:
  do zone=1,NoOfZones
   do prec=1,NoOfPrecursors
     C0(prec,zone)=beta(prec)/ lambda(prec) *SUM( NuSigmaF(zone,:)*Flux(
         zone,:))
   end do
  end do
  open (unit=17, file=trim(FILEPREC),status="NEW", access="SEQUENTIAL",
      FORM="FORMATTED",ACTION="WRITE")
  WRITE(17,"(6E12.5)") ((C0(prec,zone) ,prec=1,NoOfPrecursors),zone=1,
      NoOfZones)
  close(17)
 else
  !si el sistema no esta en equilibrio, se presupone que los NuSimgaF
      estan afectados
  betaTotal=SUM(beta(:))
  NuSigmaF=NuSigmaF/(1 - betaTotal)

  open (unit=17, file=trim(FILEPREC),status="OLD", access="SEQUENTIAL",
      FORM="FORMATTED",ACTION="READ")
  READ(17,"(6E12.5)") ((C0(prec,zone) ,prec=1,NoOfPrecursors),zone=1,
      NoOfZones)
  close(17)
 end if

end subroutine readPrecursors

subroutine calculateC()
implicit none
do prec=1,NoOfPrecursors
do zone=1,NoOfZones
dCdt(prec,zone)=beta(prec) * SUM ( NuSigmaF(zone,:) * Flux(zone,:) ) -
    lambda(prec) * C0(prec,zone)
end do
end do

C(:,:) = C0(:,:) + Dt * dCdt (:,:)

open (unit=17, file=trim(FILEPREC),status="REPLACE", access="SEQUENTIAL",
    FORM="FORMATTED",ACTION="WRITE")
WRITE(17,"(6E12.5)") ((C(prec,zone) ,prec=1,NoOfPrecursors),zone=1,
    NoOfZones)
close(17)

CALL CDB_CLOSE(30)
end subroutine calculateC
```

```
 subroutine calculateQ()
  implicit none
  do zone=1,NoOfZones
   do group=1,NoOfGroups
    q(zone,group) =SUM(chi(:,group) * lambda(:)* C(:,zone) )
   end do
   qTotal(zone)= SUM(q(zone,:))
  end do

  call writeQ()
 end subroutine calculateQ

 subroutine writeQ()
  implicit none
  open (unit=15, file=trim(FILESRC),status="REPLACE", access="SEQUENTIAL",
      FORM="FORMATTED",ACTION="WRITE")
  WRITE (15,"(I3.3)")26
  WRITE (15,"(2I3)")-1,1
  WRITE (15,"(6F12.3)") ((q(1,group)/qTotal(1)) ,group=1,NoOfGroups)
  WRITE (15,"(6(I3,F9.6))",advance='no') (zone,qTotal(zone),zone=1,
      NoOfZones)
  WRITE (15,"(I3)",advance='no') 0
  close(15)
 end subroutine writeQ

 subroutine outputResults()
  implicit none
  open (unit=15, file="Results.rlt",status="OLD", access="SEQUENTIAL",FORM
      ="FORMATTED",ACTION="WRITE",POSITION="APPEND")
  WRITE(15,*) FLUX(1,1) , C(1,1), q(1,1)
  WRITE(*,*) FLUX(1,1) , C(1,1), q(1,1)
  close (15)
 end subroutine outputResults
end program Transient
```

BIBLIOGRAFÍA

[1] Ignacio Mochi Eduardo A. Villarino Carlos A. Lecot. *CITVAP*. Version 3.9. Bariloche, Río Negro, Argentina: Nuclear Engineering Division, INVAP, 2014.

[2] *RELAP5/MOD3.3 CODE MANUAL*. Vol. II: User's Guide and Input Requirements. Idaho Falls, Idaho, USA: Nuclear Safety Analysis Division, Information Systems Laboratories Inc., 2001.

[3] Eduardo A. Villarino e Ignacio Mochi. «Thermal Hydraulic Models for Neutronic and Thermal Hydraulic Feedback in CITVAP Code». En: (2012).

[4] *Línea de Cálculo Neutrónica del Núcleo del Reactor Carem*. Bariloche, Río Negro, Argentina: División de Ingeniería Nuclear, INVAP, 2013.

[5] Ignacio Mochi Eduardo A. Villarino Pablo Sartorio. *INVAP Neutronic Calculation Line*. Bariloche, Río Negro, Argentina: División de Ingeniería Nuclear, INVAP, 2015.

[6] «Power Feedback Coefficient Measurement, OPAL,Documento RRRP 7534 EDEIN 065 A; por Ansto e INVAP SE». En: (7/11/2006).

[7] Darío Pieck. «Desarrollo y validación experimental de un algoritmo de acople neutrónico-termohidráulico para reactores de investigación, Tesis de Maestría en Ingeniería». En: (2010).

[8] D. R. Vondy G. W. Cunningham T. B. Fowler. *Nuclear reactor core analysis code: CITATION*. 2.ª ed. Oak Ridge, Tennessee, USA: Oak Ridge National Laboratory, 1971.

[9] A. F. Henry. *Nuclear-Reactor analysis*. 4.ª ed. Cambridge, Massachusetts, USA: The MIT Press, 1986.

[10] R.L.Burden y R.L. Faires. *Numerical Analysis*. 7.ª ed. Pacific Grove, California, USA: Brooks/Cole Publishing Company, 2001.

[11] Eduardo A. Villarino. *CDATABASE*. Version 2.0.2. Bariloche, Río Negro, Argentina: Nuclear Engineering Division, INVAP, 2012.

COLOFÓN

Este documento fue escrito en LaTeXusando el estilo tipográfico `classicthesis` desarrollado por André Miede e inspirado en el libro de Robert Bringhurst "*The Elements of Typographic Style*". `classicthesis` está disponible para LaTeX y LyX:

https://bitbucket.org/amiede/classicthesis/