Manual de uso de la clase C++ *SUBS*

CIMNE - Servei de Transferència de Teconologia

CIMNE - Aeronàutica

Centre Internacional de Mètodes Numèrics a l'Enginyeria

Terrassa, 31 de Enero de 2013

Índice

1. Introducción

La clase de C++ *SUBS* está orientada a trabajar con los archivos de texto de entrada del *solver* y salida de RMOP, de tal forma que facilite la comunicación del optimizador hacia el *solver*.

En la figura 1 se muestra un diagrama de flujo del caso típico de utilización de las funciones de la clase *SUBS*.

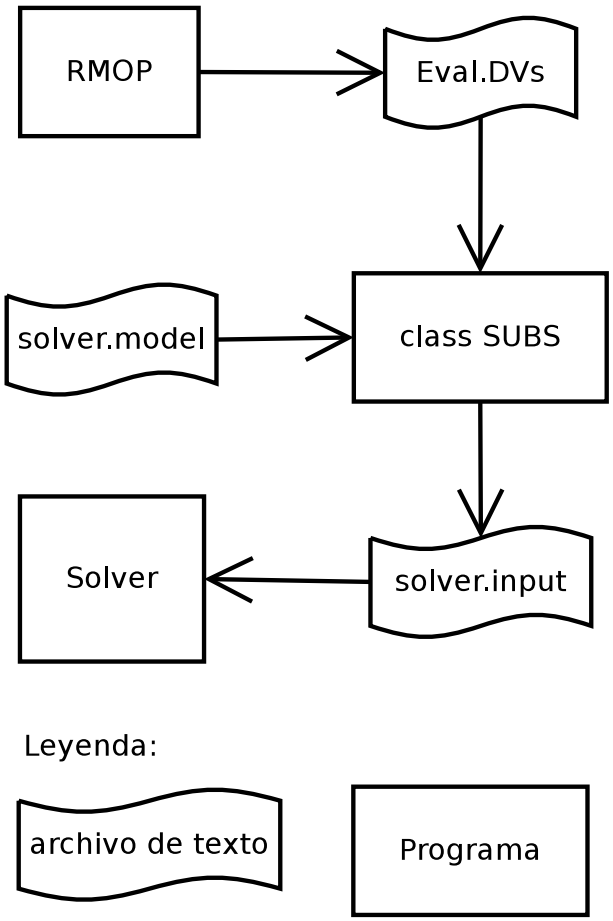

Figura 1: Diagrama de flujo de datos.

Como se muestra en el diagrama, RMOP escribe el archivo *Eval.DVs* con los valores de las variables de diseño que se quieren utilizar, con los valores separados por espacios:

#DVs1 #DVs2 #DVs3 #DVs4 ...

El archivo *solver.model* es el archivo de entrada al *solver* con la peculiaridad de que en las posiciones que debería haber los valores de la variables de diseño,

se deben colocar unas cadenas de texto con el formato:

#RMOP_DVS_n

Donde *n* es el número de variable de diseño. Este archivo servirá de patrón para que la librería lo use para crear el archivo de cálculo, *solver.input*, con los valores de la variables de diseño en sus posiciones, para poder ejecutar el *solver*.

Hay una variante de este procedimiento en el que las variables de diseño, por algún motivo no sean directamente valores que necesita el solver y haya que hacer alguna transformación. Para ello se puede hacer una llamada a la función que substituye los valores de las variables de diseño mediante un vector de *string*.

2. Uso

El uso se puede separar en los siguientes casos:

- **Substitución de valores desde un archivo:** Orientado a substituir, en el archivo del *solver*, los valores que *HPRMOP* escribe en el archivo *Eval.DVs*.
- **Substitución de valores desde un vector:** Preparado para cuando hay que modificar dichos valores. Es decir, cuando las variables de diseño no son directamente valores en los archivos de definición del problema. También se puede utilizar para trabajar cuando la información del archivo de cálculo que se quiere utilizar como variable de diseño no es numérica, sino distintas cadenas de texto. En ese caso se puede declarar una variable de diseño discreta en el optimizador y a través de esta función traducir el valor entero en la cadena de texto correspondiente.

2.1. Funciones públicas

- **constructor:** subs(**ofstream** * *logf_*)
  - **logf_:** Puntero al *ofstream* del archivo en el que se escribirá la información del proceso de las llamadas a las funciones públicas.
- **int** mount_input_file(**string** *EvalDVs*, **string** *model_file*, **string** *out_file*)

  Entrada:

- **model_file:** Nombre del archivo que contiene el modelo de archivo con las palabras clave en los lugares en que hay que substituir .
- **out_file:** Nombre del archivo que se creará con los valores substituidos.
- **EvalDVs:** Nombre del archivo que contiene los valores de las variables de diseño separadas por espacios o líneas.

Salida:

Devuelve 0 si no hay errores, 1 si hay errores.

- **int** mount_input_file(**vector**<*string*> & *DVS_*, **string** *model_file*, **string** *out_file*)

  Entrada:

  - **model_file:** Nombre del archivo que contiene el modelo de archivo con las palabras clave en los lugares en que hay que substituir .
  - **out_file:** Nombre del archivo que se creará con los valores substituidos.
  - **EvalDVs:** Nombre del archivo que contiene los valores de las variables de diseño separadas por espacios o líneas.

  Salida:

  Devuelve 0 si no hay errores, 1 si hay errores.

2.2. Palabras clave y archivo *model_file*

Las palabras clave son unas tramas de texto de la forma *#RMOP_DVS_* seguida, sin espacios, por el número que indica la posición del valor de la variable en el archivo *EvalDVs*. La primera variable es la posición 1.

Éstas no tienen porqué estar ordenadas en el archivo modelo (*model_file*), se pueden repetir y puede faltar alguno de los números. Dado que el optimizador trabaja con un solo archivo de variables de diseño, *Eval.DVs* se puede llamar a las funciones de substituir varias veces con archivos de entrada y salida distintos en caso que se trabaje con más de un *solver* o que este requiera más de un archivo de entrada.

Para ver un ejemplo de utilización de la clase *SUBS* se puede consultar [1], en el que se desarrolla un caso completo de optimización y se usa la clase *SUBS* para hacer la comunicación entre RMOP y el *solver*, XFOIL en ese caso.

Referencias

[1] Tutorial XFOIL-RMOP: (`http://tts.cimne.com/RMOP/Docs/tutorials/T1_RMOP-XFOIL_ES.zip`)