UCA Universidad de Cádiz

Escuela Superior de Ingeniería de Cádiz
Dirección

C/ Chile, 1. 11002 Cádiz.
Tel. 956015100. Fax. 956015101
direccion.esi@uca.es

Portada del TFG/M

UCA Universidad de Cádiz

Escuela Superior de Ingeniería

ESCUELA SUPERIOR DE INGENIERÍA

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES.

ANÁLISIS DEL COMPORTAMIENTO DINÁMICO DE UNA INSTALACIÓN DE COGENERACIÓN USANDO EL SOFTWARE MODELICA

AUTOR: RAÚL MACÍAS RUIZ

Cádiz, febrero 2017

Universidad de Cádiz

Escuela Superior de Ingeniería de Cádiz
Dirección

C/ Chile, 1. 11002 Cádiz.
Tel. 956015100. Fax. 956015101
direccion.esi@uca.es

Primera página del TFG/M

Universidad de Cádiz

Escuela Superior de Ingeniería

ESCUELA SUPERIOR DE INGENIERÍA

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES.

ANÁLISIS DEL COMPORTAMIENTO DINÁMICO DE UNA INSTALACIÓN DE COGENERACIÓN USANDO EL SOFTWARE MODELICA

DIRECTOR: FRANCISCO JOSÉ SÁNCHEZ DE LA FLOR
AUTOR: RAÚL MACÍAS RUIZ

Cádiz, febrero 2017

Resumen

La experimentación es un proceso fundamental en ingeniería y en las ciencias físicas. Se hace fundamental conocer el comportamiento dinámico de las instalaciones sin llegar a tener que trabajar sobre una instalación real. Una simulación es un experimento realizado sobre un modelo de un sistema. Estas simulaciones suelen realizarse sobre modelos de software con las herramientas necesarias de cómputo.

En este TFG, el sistema objeto de estudio es una instalación de cogeneración. Este tipo de instalaciones son capaces de obtener simultáneamente energía eléctrica y energía térmica útil, a partir de la energía contenida en un combustible. La mayor eficiencia energética de estos procesos respecto a procesos convencionales, hace indispensable el uso de este tipo de instalaciones.

Para modelar la instalación de cogeneración, se hará uso del lenguaje Modelica. Modelica es un lenguaje mantenido por la Modelica Association para el modelado de sistemas que nos permite describir sus propiedades. El diseño del lenguaje se ha hecho para adecuarlo al modelado de sistemas físicos dinámicos. Aunque no es un lenguaje de programación, los modelos se simulan en ordenadores y con las herramientas adecuadas permite la generación automática de código fuente en un lenguaje de programación. El entorno de modelado usado en este trabajo es la suite OpenModelica, una suite totalmente libre y gratuita.

El objetivo de esta trabajo es el de conocer el comportamiento de una instalación de cogeneración. Ello conlleva a saber cuáles son sus componentes, que variables son las fundamentales para describir su dinámica y como sus partes se conectan para cubrir una determinada demanda energética. Para abordar el estudio, se modela la instalación con OpenModelica. Para apoyarse en una instalación real, se emplea datos de una planta situada en Montilla (Córdoba) perteneciente al grupo CIAT.

Algunos problemas con los modelos de las distintas librerías gratuitas que ofrece la suite, ha llevado a profundizar en el código de programación de los diferentes modelos. Esto ha conducido a un estudio más completo de cada uno de ellos, y evaluar el estado de funcionamiento de los modelos pertenecientes a las diferentes librerías sobre sistemas térmicos que ofrece Modelica.

Además, se han puesto de manifiesto algunos errores y consideraciones que son de suma importancia para conseguir hacer funcionar a los modelos. Aunque se ha conseguido solucionar algunos de ellos, sigue estando pendiente ofrecer mejores alternativas para que cualquier modelador pueda realizar simulaciones sobre las instalaciones sin ningún tipo de error, ya sea de traducción o simulación.

Índice

Índice de figuras

Índice de tablas

1. Introducción, objetivos y estructura

1.1. Introducción

En el presente TFG se estudiará el comportamiento dinámico de una planta de cogeneración. Pero antes empecemos definiendo que es la cogeneración o un sistema de cogeneración. La cogeneración es un proceso mediante el cual se obtiene simultáneamente dos tipos diferentes de energía a partir de la misma fuente de energía primaria. Las energías generadas por un sistema de cogeneración vienen dadas generalmente en forma de energía eléctrica y energía térmica. Dicha energía térmica puede ser recuperada mediante fluidos caloportadores (refrigerantes, agua, vapor, aire...) y aprovecharse, usualmente, para fines de calefacción o refrigeración (máquinas de absorción).

La principal ventaja de la cogeneración es su mayor eficiencia energética ya que se aprovecha tanto el calor como la energía mecánica o eléctrica en un único proceso, en vez de utilizar una central eléctrica y caldera convencional. En los procesos convencionales la mayoría de la energía contenida en el combustible se transforma en calor y se pierde en el ambiente. A través de la cogeneración es posible aumentar la eficiencia energética del proceso hasta un rendimiento global que puede oscilar entre el 75% y el 90% de la energía química contenida en el combustible.

Otro beneficio importante de los sistemas de cogeneración es la reducción en gran medida de la contaminación que producen los sistemas convencionales siendo un instrumento clave para tener emisiones de GEI más bajos. Las emisiones de dióxido de carbono disminuyen entre un 30 y 40%.

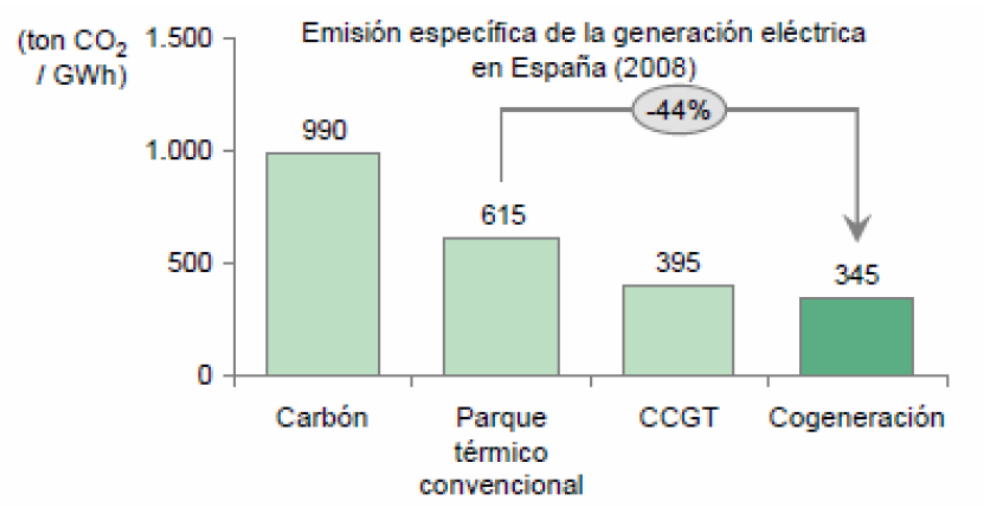


Figura 1.1: Emisiones de dióxido de carbono de diferentes sistemas de producción de energía eléctrica

Y por último, la cogeneración es fundamental para la seguridad de suministro y la reducción de la dependencia energética. Al tratarse de generación distribuida, la cogeneración produce la electricidad en el punto de consumo o en su entorno cercano, por lo que el suministro no depende de posibles fallos en las líneas eléctricas de transporte y distribución.

El funcionamiento de una planta o instalación de cogeneración se puede resumir de la siguiente forma: Se dispone de un elemento motor encargado de convertir energía química en mecánica. Puede tratarse de turbinas de gas, turbinas, de vapor o motores alternativos. En función del sistema de cogeneración elegido, el proceso puede variar ligeramente, incluyéndose diferentes componentes. La fuente de energía primaria suele ser de gas natural, gasóleo o fuelóleo. Dicha energía mecánica es aprovechada por un alternador que la transforma en energía eléctrica. La energía eléctrica se puede utilizar para autoconsumo o también se puede verter a la red. El calor inherente generado en el proceso es aprovechado mediante calderas recuperadoras, secaderos, intercambiadores de calor, etc. Una parte de la energía térmica no puede ser aprovechada y debe ser evacuada al ambiente. Elementos habituales de las plantas de cogeneración son intercambiadores de calor, máquinas de absorción (para producir frio), torres de refrigeración, aerocondensadores, etc. El sistema de aprovechamiento de calor tanto para calefacción o refrigeración requiere de sistemas de tratamiento y control.

La energía térmica tratada se usa finalmente para climatización (calefacción o refrigeración) y ACS.

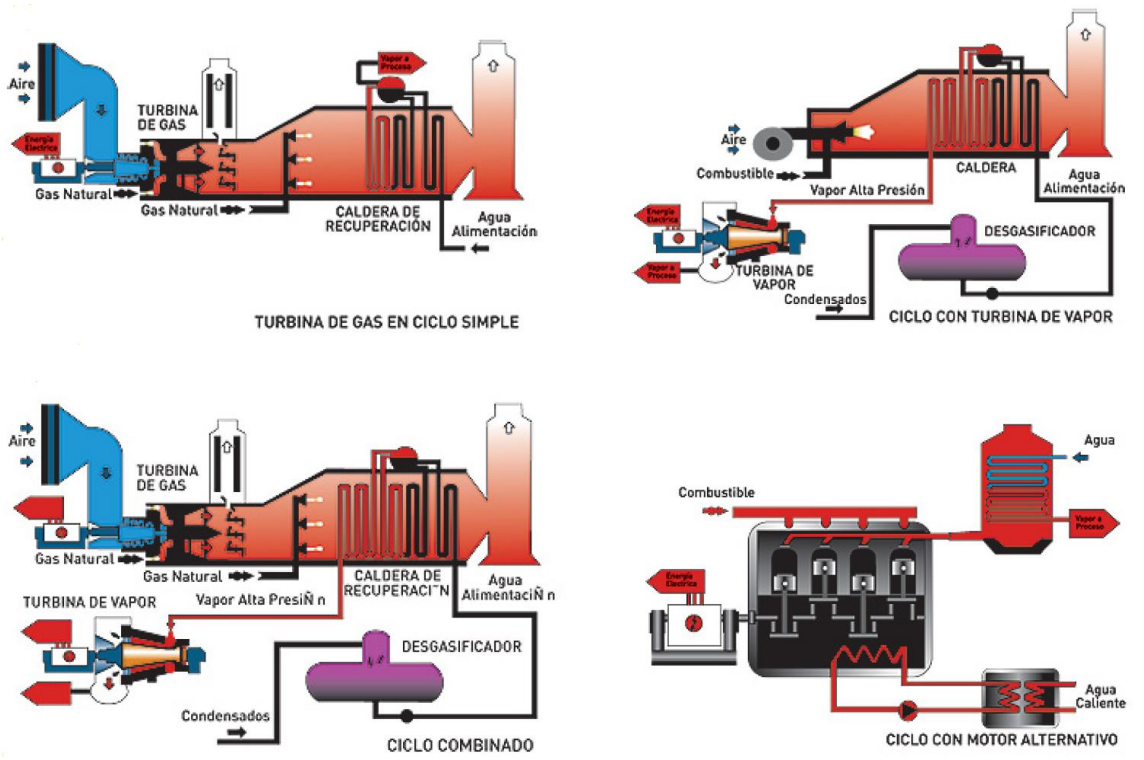


Figura 1.2: Tipos de plantas de cogeneración

Una vez explicado resumidamente un sistema de cogeneración, indicar que para afrontar el análisis de la planta teniendo unos recursos limitados se hace necesario recurrir a técnicas de simulación computacional para poder trabajar y estudiar el sistema. El software utilizado será Modelica. El potencial de estas técnicas de simulación serán explicadas en secciones posteriores.

1.2. Objetivos

El objetivo principal que persigue este TFG es el de analizar el comportamiento dinámico de una instalación de cogeneración. Para abordar el estudio será necesario identificar los componentes de una planta de cogeneración, conocer y utilizar los servicios que nos ofrecen las herramientas de modelado y simulación del software Modelica, conocer los parámetros que caracterizan los componentes de una planta de cogeneración y la transmisión de información entre ellos, implementación del esquema en un entorno de modelado y por ultimo trabajar sobre el modelo realizando un conjunto de simulaciones para sacar conclusiones sobre el funcionamiento de la planta.

1.3. Estructura

El trabajo se ha estructurado en un total de 6 secciones o capítulos. Para una mayor facilidad de visualización se describe a continuación un breve resumen de lo que contendrá cada capítulo.

- ✓ Capítulo 2: En este capítulo se introducirá las peculiaridades que presenta Modelica para modelar sistemas dinámicos. Se destacaran algunas características del lenguaje, se describirá la librería de modelos Modelica abierta más utilizada, se presentará que librerías especificas nos serán de utilidad para nuestro cometido y se repasara algunas herramientas del entorno de modelado que se utilizará para desarrollar la planta.

- ✓ Capítulo 3: Se hace necesario antes de la implementación de nuestro modelo en Modelica, abordar un análisis de una planta de cogeneración tipo. Se utiliza para su estudio un esquema resultante de una planta de cogeneración real. Se indicará los elementos de la instalación, sus conexiones, comportamiento de la planta en general, balances energéticos y otras consideraciones.

- ✓ Capítulo 4: En esta sección se hará un recorrido por los componentes presentes en las distintas librerías de OpenModelica. Se explicará los modelos que son candidatos para utilizarlos en la instalación, se describirá muchas de las incidencias imprevistas que se han producido durante el desarrollo del trabajo, se propondrán modelos alternativos y ejemplos para la validación y explicación de los componentes. Por último se desarrollará la instalación en OM.

- ✓ Capítulo 5: Se presentarán las conclusiones finales de todas las tareas realizadas. Se justificará algunas decisiones finales y se propondrá líneas de trabajo futuro.

2. Herramienta software Modelica

2.1. Introducción

Una definición muy amplia del término sistema puede ser "el objeto o colección de objetos cuyas propiedades queremos estudiar". Para hacerlo, construimos representaciones de ese sistema que reciben el nombre de modelos. Algunos de los modelos están construidos de tal manera que permiten realizar experimentos sobre ellos. El objetivo es experimentar sobre el modelo en lugar de hacerlo sobre el sistema real. Tales experimentos se denominan simulaciones. El interés en este trabajo es construir un modelo de un sistema dinámico. A partir de modelos matemáticos que caractericen el sistema se pueden obtener modelos de software para realizar simulaciones en ordenador.

Existen muchas técnicas de construcción de modelos matemáticos mediante las cuales se pueden obtener ecuaciones susceptibles de ser simuladas por ordenador pero la elegida para este trabajo es el modelado orientado a objetos.

El enfoque de la programación orientada a objetos (OOP) tiene sus raíces en los problemas de simulación. No obstante, el desarrollo de un paradigma de modelado orientado a objetos (OOM) es posterior al paradigma de programación orientada a objetos. Las características mínimas que presentan son:

- ✓ **Instanciación de objetos**: una clase es una definición genérica de un modelo. Esta definición se vuelve específica cuando se crea un objeto de esa clase. En otras palabras, un objeto es cada instancia que se crea del modelo (de la clase). De esta forma, es posible reutilizar el modelo que reside en la clase en múltiples ocasiones.

- ✓ **Encapsulamiento del conocimiento**: la información de cada objeto está encapsulada, lo que significa que cada objeto guarda su información internamente sin mezclarse con la información de otros objetos. Esta característica protege la información propia de cada objeto, de tal manera que en el diseño de las clases no es necesario preocuparse por los efectos que otros objetos (quizás aún no diseñados) puedan llegar a tener accidentalmente.

- ✓ **Capacidades de interconexión topológicas**: los objetos pueden interconectarse para ensamblar modelos más sofisticados. Los mecanismos de conexión deben ser tales que permitan un diseño tipo plug and play a nivel de software.

- ✓ **Capacidades de construcción de redes generalizadas**: uno de los mecanismos de interconexión disponibles son nodos que permiten conectar un número arbitrario de elementos y de esta forma construir redes complejas de objetos.

- ✓ **Modelado jerárquico**: un modelo sofisticado puede contener modelos más simples en su interior. Esta característica busca que el modelo pueda reflejar las estructuras reales que existen en el sistema a ser modelado.

- ✓ **Herencia de clases**: mediante este mecanismo es posible definir clases especializadas a partir de una o más clases base. La herencia facilita la organización, reutilización y mantenimiento del código, así como la construcción de librerías estructuradas.

- ✓ **Polimorfismo**: esta característica permite la definición de interfaces genéricas para conectar elementos de diferentes tipos, lo que facilita la construcción de clases genéricas, cuyo comportamiento se hace explícito en diferentes clases especializadas.

2.2. Lenguaje Modelica

Modelica es un lenguaje orientado a objetos para el modelado de sistemas. No es un lenguaje de programación, aunque los modelos que se construyen con Modelica se simulan en ordenadores: el lenguaje está diseñado para describir las propiedades de sistemas. El énfasis en el diseño del lenguaje se ha hecho para adecuarlo al modelado de sistemas físicos dinámicos. No obstante, su versatilidad permite usarlo para otro tipo de sistemas, tales como los económicos y sociales. El lenguaje es utilizado por varias herramientas de software cuyo propósito es la simulación del comportamiento de sistemas dinámicos. Estas herramientas son las encargadas, entre otras cosas, de traducir la descripción del sistema en código ejecutable por un ordenador. En otras palabras, aunque Modelica no es un lenguaje de programación, en combinación con las herramientas adecuadas permite la generación automática de código fuente en un lenguaje de programación. Se trata de un lenguaje de uso abierto, no propietario, diseñado y mantenido por la **Modelica Association**.

2.2.1. Modelo de componentes y conectores

Un modelo complejo Modelica se construye conectando modelos más simples, tal como se ilustra en la figura 2.1.

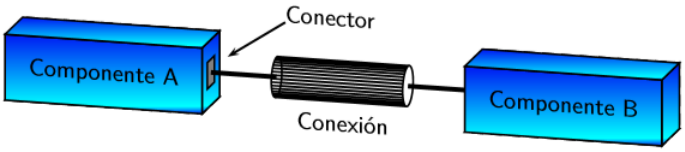


Figura 2.1: Conexión de dos componentes Modelica

Cada componente del modelo está dotado de conectores. La conexión de dos o más componentes se realiza a través de esos conectores, ensamblando así el modelo mayor. El diseño de los conectores es un aspecto crítico del modelado orientado a objetos. Por una

parte, para cada tipo de acople físico entre sistemas es necesario disponer de un tipo de conector. Por otra, el diseño de los conectores debe ser lo suficientemente genérico para permitir acoples de componentes diversos. La figura 2.2 muestra un diagrama de conexión de varios componentes, en el que se ilustra que más de dos componentes pueden estar conectados a un mismo conector.

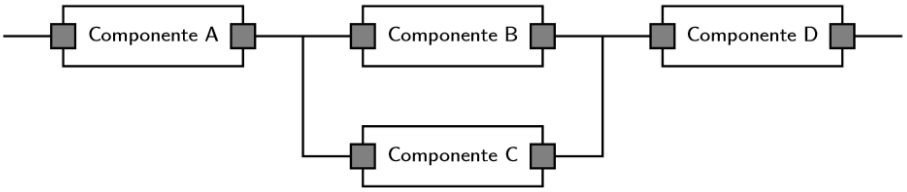


Figura 2.2: Esquema de conexión de varios componentes Modelica

En un conector hay dos tipos de variables:

- **Variables de flujo**: típicamente representan algún tipo de variable que fluye.
- **Variables que no son de flujo**: típicamente representan algún tipo de variable semejante a un potencial o nivel de energía.

La diferencia real entre los dos tipos de variables se hace evidente cuando dos o más componentes se conectan a través de un conector:

Con las variables de flujo se construye una ecuación del tipo suma de variables igual a cero. Con las variables que no son de flujo se construyen ecuaciones de igualdad entre todas las variables.

Las conexiones en un modelo Modelica sólo pueden realizarse entre conectores del mismo tipo. No es correcto conectar, por ejemplo, un conector eléctrico con uno térmico. La razón fundamental para esta restricción es que la información asociada a cada tipo de conector es diferente. En los conectores de la figura 2.3, el tipo de información asociada es la siguiente:

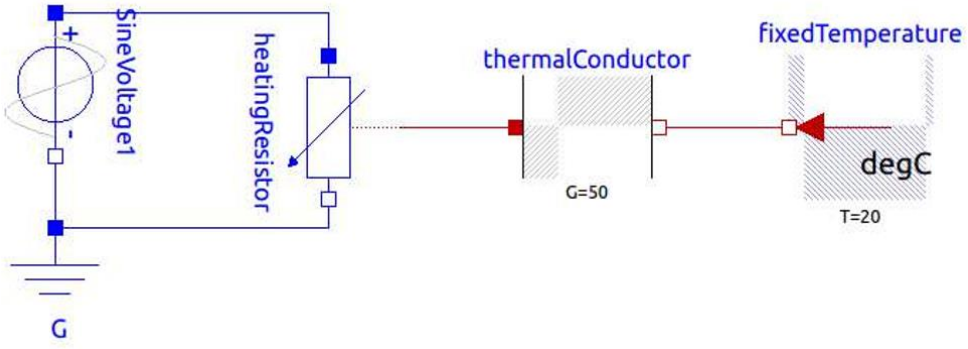


Figura 2.3: Diagrama que representa el calentamiento de un cuerpo a través de una resistencia eléctrica

En los conectores eléctricos:

- Corriente eléctrica (variable de flujo).
- Diferencia de potencial eléctrico (variable que no es de flujo).

En los conectores térmicos:

- Flujo de calor (variable de flujo).
- Temperatura (variable que no es de flujo).

Un mismo modelo puede tener conectores de diferente tipo.

2.2.2. Orientación

Modelica es un lenguaje orientado a objetos y por tanto comparte algunas características comunes con los lenguajes de programación orientados a objetos tales como encapsulamiento, herencia, polimorfismo, etc. En la práctica, esto significa que es posible definir clases y luego crear diferentes instancias de una misma clase, cada una de ellas con propiedades específicas. También significa que es posible crear una jerarquía de clases. Como elemento particular del lenguaje, se definen **clases especializadas**, que tienen alguna especificación adicional para facilitar el modelado de algunos componentes. El listado de clases especializadas es el siguiente:

- **model**: es el tipo de clase más general. La única restricción consiste en que las instancias de estas clases no pueden usarse en conexiones.
- **record**: esta es una clase para encapsular propiedades sin comportamientos. Por tanto, no pueden definirse ecuaciones.
- **type**: se utiliza para definir nombres de clases como tipos de dato.
- **connector**: esta clase es usada específicamente para definir los conectores de los modelos. Se utilizan para almacenar información que se comparte entre dos componentes que se conectan. No pueden contener ecuaciones.
- **block:** se trata de una clase en la que la causalidad está predefinida. Las variables declaradas deben acompañarse del prefijo input u output, para declarar la causalidad.
- **function:** son clases definidas para modelar el concepto de funciones matemáticas. Son semejantes a los bloques, pero no pueden definirse ecuaciones; en su lugar se utilizan algoritmos explícitos o llamados a funciones externas.
- **package:** se utiliza para organizar espacios de nombres y librerías. Un paquete sólo contiene declaraciones de clases.

2.2.3. Multidominio

Modelica es un lenguaje de propósito general, en el sentido que permite modelar sistemas de dominios físicos diversos. Por 'dominio' entendemos aquí las distintas ramas de la ciencia y la tecnología tales como biología, química, electricidad, mecánica, hidráulica, redes, etc. Existen herramientas de software para simular fenómenos de cada uno de esos dominios que se apoyan en lenguajes diseñados para describir dichos fenómenos. Estos lenguajes, no obstante, difícilmente pueden describir fenómenos de otros dominios, y por tanto las capacidades de simulación quedan necesariamente circunscritas a su dominio específico. En su lugar, Modelica ha sido concebido para permitir el modelado y simulación de fenómenos multidominio. Un mismo modelo Modelica puede, por ejemplo, integrar fenómenos eléctricos, mecánicos y térmicos. La orientación a objetos permite construir jerarquías de clases partir de definiciones muy generales a otras más específicas. Por esta razón, los modelos Modelica pueden a la vez:

- Ser tan fáciles de emplear como los modelos diseñados para herramientas de dominio específico.
- Tener la versatilidad necesaria para construir modelos multidominio.

2.2.4. Acausalidad

Un modelo es causal si es posible clasificar de antemano las variables que definen su comportamiento como variables de entrada (conocidas) y de salida (por conocer).

Modelica permite modelos con relaciones causales y acausales. Para diferenciar se utiliza dos operadores diferentes:

- Operador de igualdad (=) → relaciones acausales
- Operador de asignación (:=) → relaciones causales

En las relaciones acausales es fundamental implementar algún algoritmo de ordenamiento.

2.2.5. Modelo del tiempo

El tiempo se representa como una variable continua. El nombre de la variable tiempo es time, y la función der() representa la derivada respecto al tiempo.

No obstante, también es posible construir modelos discretos. Para ello, Modelica utiliza el concepto de evento. Un evento es algo que ocurre y presenta las siguientes propiedades:

- Es instantáneo, su duración en el tiempo es cero.
- Tiene asociado una condición del evento que debe pasar de falsa a verdadera para que el evento ocurra.

- Tiene un conjunto de variables asociadas al evento, cuyo valor cambia cuando el evento ocurre.

- Tiene algún comportamiento asociado a través de ecuaciones condicionales.

La función sample(to,dt) está diseñada para facilitar el modelado de sistemas discretos de periodo constante. El periodo es dt, y el primer cambio sucede en to.

2.2.6. Documentación de los modelos

Cada modelo Modelica puede contener su propia documentación interna a través de la palabra clave **annotation**. Los dos usos principales de esta capacidad son:

- Incorporar una descripción textual del modelo, sus características y forma de uso. Esta descripción se guarda en formato HTML.

- Incorporar una descripción gráfica del modelo, en forma de ícono. Esta descripción se hace a través de primitivas gráficas propias del lenguaje.

2.2.7. Reusabilidad

Modelica ha sido diseñado para potenciar la reutilización de componentes. La orientación a objetos, la acausalidad y el modelo de componentes-conectores son la columna vertebral del lenguaje que lo permite. En la práctica, esta característica se manifiesta en la existencia de librerías de componentes. La colección de librerías se tratará más adelante.

2.2.8. Algunos elementos del lenguaje

Estos elementos que no han sido presentados con anterioridad y que reflejan el potencial del lenguaje son:

Tipos de datos: La clase especializada type permite definir nuevos tipos de datos. Por ejemplo, el tipo de dato ***Real*** es una clase que tiene los siguientes atributos:

- *value*: es el valor en punto flotante de doble precisión de la variable.

- *quantity*: nombre de la cantidad física que representa; por ejemplo "Longitud".

- *unit*: unidades en que se mide la cantidad; por ejemplo "m".

- *displayUnit*: unidades utilizadas para desplegar información; por ejemplo "km".

- *min*: valor mínimo que puede tomar la variable.

- *max*: valor máximo que puede tomar la variable.

- *start*: valor inicial fijo o candidato.

- *fixed*: valor booleano que especifica si el parámetro start es un valor inicial fijo o candidato.
- *nominal*: valor nominal de la variable.
- *stateSelect*: variable auxiliar para diferenciar estados.

De forma semejante, los tipos de datos ***Integer, Boolean y String*** son clases con atributos.

Arreglos de datos: Se utilizan cuatro designaciones diferentes para referirse al concepto de 'arreglo de datos':

- Escalares: datos simples.
- Vectores: matrices unidimensionales
- Matrices: matrices bidimensionales.
- Arreglos: matrices con k dimensiones (k > 0).

El lenguaje es rico en instrucciones para manipulación de arreglos. Cuenta con funciones para:

- Declarar, dimensionar y redimensionar arreglos.
- Gestionar los datos en los arreglos.
- Aplicar operaciones sobre los datos de los arreglos.
- Operar arreglos entre sí.
- Aplicar operaciones algebraicas sobre los arreglos.

Ecuaciones condicionales e iterativas: Las ecuaciones definen las relaciones entre las variables, por ejemplo a través de la instrucción **connect()**. En Modelica es posible modelar ecuaciones, es decir conexiones de componentes, apoyadas en estructuras tipo for, if-then-else, while, when.

2.3. OpenModelica

La simulación de modelos preparados en lenguaje Modelica requiere el uso intensivo de herramientas de software. Existen varias herramientas de compilación, simulación y visualización, algunas de las cuales son propietarias y otras son libres. Dentro de la oferta de herramientas libres destaca OM por su grado de maduración, soporte y desarrollo permanente. Dadas estas características, será el entorno de modelado utilizado en el trabajo.

OM es un entorno de modelado y simulación basado en Modelica. Cabe destacar que es de código abierto y gratuito, disponiendo de un desarrollo y soporte continuo del

conjunto de herramientas que conforman la suite por la organización sin ánimo de lucro Open Source Modelica Consortium (OSMC).

La figura 2.4 muestra el esquema de comunicación entre los diferentes módulos de OpenModelica que operan bajo el esquema cliente/servidor.

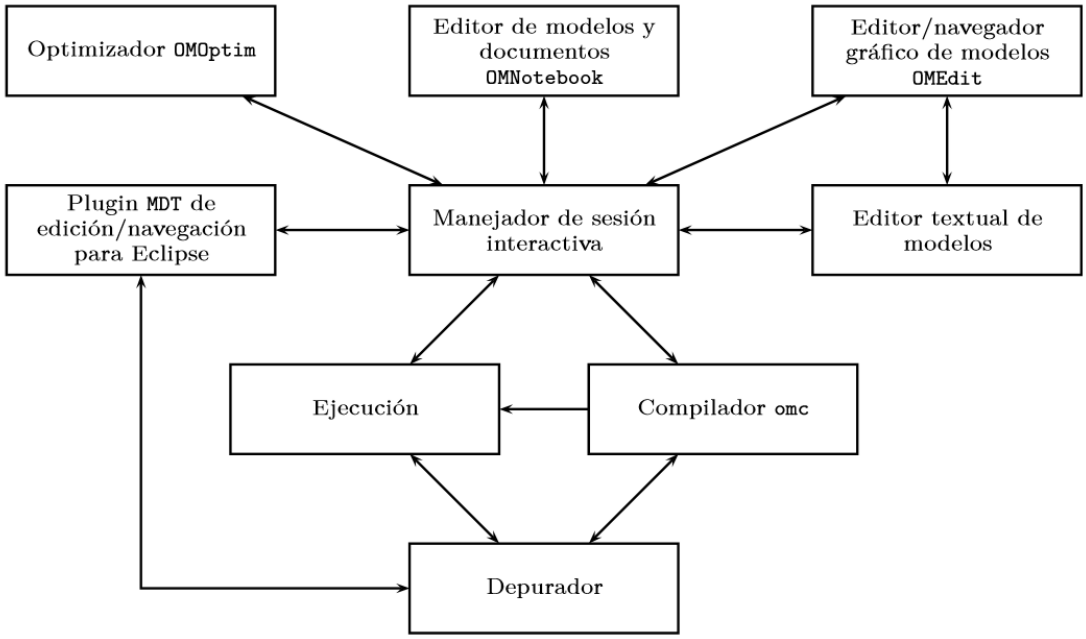


Figura 2.4: Interacción de los módulos de la suite OpenModelica

La función de cada uno de ellos es la siguiente:

- **Manejador de sesión interactiva:** interpreta y traslada comandos del sistema e instrucciones Modelica que permiten la evaluación y simulación de modelos, así como la visualización de resultados.
- **Compilador omc:** es el módulo encargado de la generación del código binario ejecutable a partir del código fuente en lenguaje Modelica.
- **Ejecución:** la ejecución del archivo compilado realmente es responsabilidad del sistema operativo.
- **Editor textual de modelos**: la construcción del código fuente de forma textual puede realizarse a través de cualquier editor de textos convencional. La suite no provee ningún editor de textos especial.
- **Editor de modelos y documentos OMNotebook**: Este módulo permite la elaboración de textos simples en los que se pueden insertar bloques de simulación y graficación de resultados.
- **Editor/navegador gráfico de modelos OMEdit:** se trata de una herramienta gráfica que permite la edición de modelos Modelica, la

navegación a través de librerías y modelos, y la visualización de resultados de simulación.

- **Plugin MDT de edición/navegación para Eclipse**: Eclipse es un ambiente de desarrollo de software ampliamente utilizado. Este plugin permite el desarrollo de modelos Modelica en dicho ambiente, la compilación de los modelos y la visualización de los resultados.

- **Depurador:** el depurador de modelos se utiliza también a través de Eclipse. Para ello se cuenta con un plugin adicional.

- **Optimizador OMOptim:** esta herramienta utiliza heurísticas para la búsqueda de los parámetros de un modelo Modelica que minimicen una función objetiva incluida en dicho modelo.

- **Otros módulos:** existen otros módulos y desarrollos (aún en consolidación) que no se han incorporado en la figura 2.4.

2.4. Librerías de sistemas térmicos

Existe multitud de librerías Modelica, tanto colecciones de uso libre y/o gratuito como comerciales. Estas librerías han sido desarrolladas por diferentes instituciones a lo largo de los últimos años y puestas a disposición de la comunidad a través de la web de la Modelica Association. Existen librerías específicas de componentes eléctricos, electrónicos y magnéticos, de componentes mecánicos, de fluidos, elementos de control, funciones, plantas de potencia, climatización, etc. Hay que destacar la librería estándar MSL, cuyo desarrollo y mantenimiento está al cargo de la Modelica Association. La MSL presenta gran cantidad de modelos de uso común en los diversos campos de la ingeniería. La MSL permite reutilizar algunos de sus elementos y paquetes para explorar la capacidad de reutilización de modelos. Aunque no se han utilizado para este trabajo, las siguientes librerías comerciales pueden ser de gran utilidad para modelar la instalación:

- Air Conditioning library
- Heat Exchanger library
- Thermal Power library

Entre las librerías de libre disposición que pueden ser de utilidad para el modelado y la simulación de sistemas térmicos se encuentran:

- ✓ **MSL**

Paquete Modelica® es un paquete estandarizado y libre que se desarrolla junto con el lenguaje Modelica® de la Modelica Association, ver https://www.Modelica.org. También se llama la Modelica Standard Library. Proporciona componentes de modelos en muchos dominios que se basan en definiciones de interfaz estandarizado.

Esta versión de la Modelica Standard Library consiste en:

- **1600** modelos y bloques y
- **1350** funciones

que son directamente utilizables (=número de clases públicas, no parcial). Es totalmente compatible con Modelica especificación versión 3.2 revisión 2 y ha sido probado con herramientas Modelica de diferentes proveedores.

Tabla 2.1: Contenido de la librería Modelica Standard Library

Nombre	Descripción
UsersGuide	Guía de usuario
Blocks	Librería de bloques de control básicos de entrada/salida (continua, discreta, lógica, bloques gráficos)
ComplexBlocks	Librería de bloques de control básicos de entrada/salida con señales complejas
StateGraph	Librería de componentes de la máquina del estado jerárquico para modelar eventos discretos y sistemas reactivos
Electrical	Librería de modelos eléctricos (analógico, digital, maquinas, multi-fase)
Magnetic	Librería de modelos magnéticos
Mechanics	Librería de componentes mecánicos de 1 dim. y 3-dim. (multi-cuerpo, rotación, traslación)
Fluid	Librería de modelos de 1 dim. de flujos termo-fluidos mediante la descripción de los medios de Modelica.Media
Media	Librería de modelos de propiedades de los medios

Thermal	Librería de componentes de sistemas térmicos, modelos de transferencia de calor y flujos termo-fluidos en tuberías
Math	Librería de funciones matemáticas (p.ej., sin, cos) y de funciones que operan sobre vectores y matrices
ComplexMath	Librería de funciones matemáticas complejas (p.ej., sin, cos) y de funciones que operan sobre matrices y vectores complejos
Utilities	Librería de funciones de utilidad dedicada a secuencias de comandos (operación en archivos, secuencias, sistema)
Constants	Librería de constantes matemáticas y constantes de la naturaleza (p.ej, pi, eps, R, sigma)
Icons	Librería de iconos
SIunits	Librería de definiciones de tipo y unidad basada en unidades de SI según la norma ISO 31-1992

- ✓ **ThermoPower**

La librería ThermoPower es una libraría Modelica de código abierto para el modelado dinámico de centrales térmicas y sistemas de conversión de energía. Proporciona componentes básicos para el modelado de nivel de sistema, en particular para el estudio de sistemas de control en plantas de energía tradicionales e innovadores y sistemas de conversión de energía.

La librería ha estado bajo desarrollo continuo en el Politecnico di Milano desde el año 2002. Se ha aplicado a la modelación dinámica de generadores de vapor, centrales de ciclo combinado, plantas de energía nuclear de III y IV generación, plantas solares de generación directa de vapor, centrales de ciclo Rankine orgánicas y circuitos criogénicos para aplicaciones de fusión nuclear. El principal autor es Francesco Casella, con contribuciones de Leva Alberto Matilde Ratti, Luca Savoldelli, Roberto Bonifetto, Stefano Boni, Leonardo Pierobon y muchos otros. La librería está licenciada bajo la Modelica licencia 2.

La última versión publicada es Beta 3.1 0, que utiliza la Modelica Standard Library 3.2.1 y Modelica 3.2 revisión 2.

Acuerdo de licencia

El paquete ThermoPower está licenciado por Politecnico di Milano en Modelica licencia 2.

Tabla 2.2: Contenido de la librería ThermoPower

Nombre	Descripción
System	Propiedades y valores predeterminados del sistema
Media	Modelos de medios para la librería ThermoPower
Electrical	Modelos simplificados de componentes eléctricos
Icons	Iconos para la librería ThermoPower
Choices	Enumeración de opciones para modelos de ThermoPower
Examples	Ejemplos de aplicación
Gas	Modelos de componentes con gases ideales como fluido de trabajo
Functions	Funciones de Miscelánea
PowerPlants	Modelos de componentes de centrales termoeléctricas
Test	Casos de prueba para los modelos de ThermoPower
Thermal	Modelos termales de transferencia de calor
Water	Modelos de los componentes con agua/vapor como fluido de trabajo
Units	Types con unidades personalizadas

✓ Buildings

La librería Buildings es una librería gratuita para el modelado de sistemas de energía y control. Muchos modelos se basan en modelos del paquete Modelica.Fluid y usan los mismos puertos para garantizar la compatibilidad con la Modelica Standard Library.

La página web de esta librería es http://simulationresearch.lbl.gov/modelica, y el desarrollo de la página es https://github.com/lbl-srg/modelica-buildings.

Tabla 2.3: Contenido de la librería Buildings

Nombre	Descripción
UsersGuide	Guía de usuario
Airflow	Paquete para calcular el transporte de flujo de aire y contaminantes entre habitaciones
BoundaryConditions	Paquete con modelos para las condiciones de contorno
Controls	Paquete con modelos para controles
Electrical	Paquete con modelos de sistemas eléctricos
Fluid	Paquete con modelos de sistemas de flujo de fluidos
HeatTransfer	Paquete con modelos de transferencia de calor
Media	Paquete con modelos del medio
Rooms	Paquete con modelos de habitaciones
Utilities	Paquete con funciones de utilidad como I/O
Types	Paquete con definiciones de type
Examples	Colección de modelos que ilustran el uso del modelo y prueba de modelos
BaseClasses	Paquete con clases base para la librería Buildings

✓ **BuildSysPro**

La librería BuildSysPro es una librería gratuita de Modelica de código abierto para modelado de sistemas de construcción y energía.

Esta biblioteca está diseñada para ser utilizado en varios contextos, incluyendo la investigación de la física de edificios, evaluación del desempeño global, desarrollo tecnológico y evaluación de impacto. Es también una base para la simulación de acciones urbana y de construcción. BuildSysPro está destinado a un público relativamente grande desde científicos de I + D hasta servicios de ingenieros de edificios.

BuildSysPro contiene clases para describir todo el edificio y sus sistemas de energía incluyendo componentes de la envolvente, climatización y otros dispositivos de conversión de energía (ACS, paneles fotovoltaicos y térmicos...) y modelos de condiciones límite.

Autorizado por la EDF bajo la Modelica licencia 2

Tabla 2.4: Contenido de la librería BuildSysPro

Nombre	Descripción
UsersGuide	Guía de usuario
Building	Paquete con modelos de flujo de aire y envolvente del edificio
Systems	Paquete de modelos de sistemas de energía y control
BoundaryConditions	Paquete con los modelos de condiciones de contorno
BuildingStock	Paquete con construcciónes
Utilities	Paquete de clases de utilidad
BaseClasses	Clases base de la librería BuildSysPro

✓ **ThermoSysPro**

ThermoSysPro proporciona modelos de los componentes más utilizados para el modelado estático y dinámico de sistemas termodinámicos de procesos industriales, como

centrales nucleares, térmicas o solares, sistemas de conversión de energía, etc. Implica disciplinas como la termohidráulica, combustión o radiación solar.

Los modelos de la biblioteca están validados con situaciones de prueba de referencia de los campos nuclear, térmico, solar y de biomasa. Se puede utilizar para el dimensionamiento del sistema, verificación y validación de sistemas de control, diagnostico de sistemas, monitoreo de plantas…

ThermoSysPro Version 3.1

Tabla 2.5: Contenido de la librería ThermoSysPro

Nombre	Descripción
AAAUsersGuide	Licencia ThermoSysPro y Guía de usuario
Combustion	Librería de combustión
Correlations	Librería de correlación de fluidos
ElectroMechanics	Librería de componentes electro-mecánicos
FlueGases	Librería de componentes de gases de combustión
Functions	Funciones de propósito general
HeatNetworksCooling	Librería de refrigeración redes de calor
InstrumentationAndControl	Librería de instrumentación y control
MultiFluids	Librería de componentes multi-fluidos
Properties	Librería de propiedades de fluidos
Solar	Librería de componentes solares
Thermal	Librería de componentes térmicos
Units	Adicional SI y no unidades-SI

WaterSolution	Librería de componentes de solución de agua
WaterSteam	Librería de componentes de agua/vapor
Examples	

Aunque se hayan presentado estas cuatro librerías, el devenir del trabajo ha llevado solo a seleccionar componentes de solo dos librerías (ThermoSysPro y Buildings). Esto no quiere decir que no haya componentes validos en las otras librerías, ya que se ha trabajado con muchos de esos modelos, pero eran menos adecuados para nuestra instalación de cogeneración, ya que no se encontraban gran parte de los modelos necesarios para crear la planta.

3. Análisis instalación de cogeneración

3.1. Introducción

En este capítulo se aborda el análisis de la instalación de cogeneración a modelar. Se realiza una breve descripción de los componentes principales de la planta. A partir de los datos suministrados de la instalación de cogeneración que CIAT tiene en Montilla (Córdoba), se generará un esquema resultante que servirá de apoyo para estudiar los componentes de la instalación, sus conexiones y el comportamiento de la planta.

Es importante resaltar que el éxito de una planta de cogeneración dependerá del diseño de su balance, y para ello es necesario tener en cuenta los siguientes criterios:

- Si interesa cubrir todas las necesidades térmicas o si se van a mantener sistemas complementarios.
- Cubrir exactamente las necesidades eléctricas, generar por debajo del consumo y continuar importando, o generar por encima del consumo y exportar a la red.
- Disponibilidad, calidad y precio de combustibles.
- Temperatura a la que se debe suministrar la energía térmica.
- Proporción entre la energía térmica y la energía eléctrica a producir y nivel de eficiencia buscado.

En el siguiente apartado se explicará brevemente el funcionamiento de los principales componentes de la instalación de cogeneración.

3.2. Componentes

Los componentes principales que constituyen la instalación son los siguientes:

- Motor a gas con refrigeración por agua
- Colector de varias entradas/salidas de agua
- Bomba de agua (circulación)
- Intercambiador de calor
- Máquina de absorción
- Torre de refrigeración

- Aerotermo (bomba de calor aire-agua)
- Bomba de calor agua-agua

Motor a gas con refrigeración por agua

Los motores alternativos de combustión interna (MACI) son máquinas volumétricas consistentes en un dispositivo de cilindro-émbolo en que se produce una reacción de combustión y se transforma la energía liberada en un efecto motor útil mediante un mecanismo de biela-manivela, y también en forma de calor. Básicamente, los MACI se basan en dos tipos de ciclos termodinámicos: ciclo diesel y ciclo de Otto.

Los MACI se caracterizan por un rango de aplicación cuanto a potencia eléctrica que va de unos 5 kWe hasta unos 15.000 kWe, aunque para aplicaciones superiores a los 8-10 MWe se prefiere el uso de otros sistemas, principalmente turbinas de gas.

Con los motores alternativos se obtienen rendimientos eléctricos más elevados pero, por otra parte, con una mayor limitación en lo referente al aprovechamiento de la energía térmica. Esta energía térmica posee un nivel térmico inferior y se encuentra repartida entre diferentes subsistemas (gases de escape y circuitos de refrigeración de aceite, camisas y aire comburente del motor).

Los sistemas con motor alternativo presentan una mayor flexibilidad de funcionamiento, lo que permite responder de manera casi inmediata a las variaciones de potencia, sin que ello conlleve un gran incremento en el consumo específico del motor.

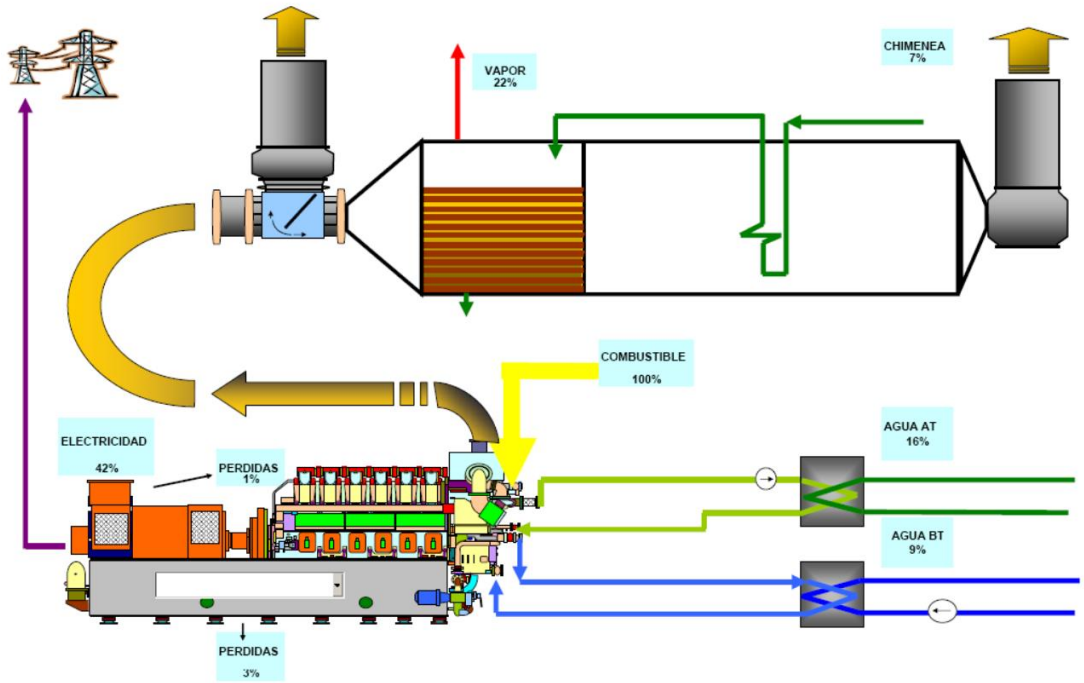


Figura 3.1: Planta de cogeneración con motor a gas

Colector de varias entradas/salidas de agua

La función de los colectores es la de obtener varios circuitos derivados partiendo de un único circuito principal. Siempre deberá de existir un colector de impulsión y un colector de retorno por cada zona a calentar o enfriar. Dentro de una instalación podrán existir varias zonas, incorporando cada una de ellas, un conjunto colector de impulsión y colector de retorno.

Un conjunto de colectores, incluye colector de impulsión y de retorno, además de los siguientes componentes, válvulas de paso a la entrada de cada colector, termómetros, purgadores automáticos y válvulas de llenado y purga.

Bomba de agua (circulación)

Hasta hace pocos años, era usual que las instalaciones hidráulicas funcionaran por gravedad o termosifón, circulando el fluido, agua normalmente, por diferencia de densidades generada por las diferencias de temperatura entre unos puntos y otros de la red de tuberías.

En la actualidad, para disminuir los costos de las redes hidráulicas, reduciendo las secciones de tubería, se utilizan bombas que, al aumentar la disponibilidad de presión, permiten velocidades más elevadas de circulación.

En las instalaciones hidráulicas se emplean casi exclusivamente bombas centrifugas para bombear líquidos en general, debido a su funcionamiento prácticamente silencioso. Transforman la energía mecánica de un impulsor en energía cinética o de presión de un fluido incomprensible. El fluido entra por el centro del rodete o impulsor, que dispone de unos álabes para conducir el fluido, y por efecto de la fuerza centrífuga es impulsado hacia el exterior, donde es recogido por la carcasa o cuerpo de la bomba, dispuesta sobre un eje al que se acopla el motor de arrastre, generalmente eléctrico. Debido a la geometría del cuerpo, el fluido es conducido hacia las tuberías de salida o hacia el siguiente impulsor.

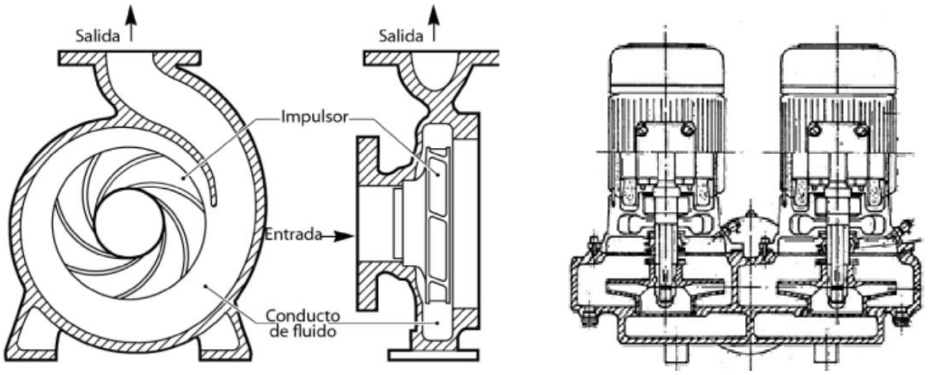


Figura 3.2. Diferentes bombas circuladoras (bancada y en línea)

Intercambiador de calor

En los sistemas mecánicos, químicos, nucleares y otros, ocurre que el calor debe ser transferido de un lugar a otro, o bien, de un fluido a otro. Los intercambiadores de calor son los dispositivos que permiten realizar dicha tarea. Entre las principales razones por las que se utilizan los intercambiadores de calor se encuentran las siguientes:

- Calentar un fluido frío mediante un fluido con mayor temperatura.
- Reducir la temperatura de un fluido mediante un fluido con menor temperatura.
- Llevar al punto de ebullición a un fluido mediante un fluido con mayor temperatura.
- Condensar un fluido en estado gaseoso por medio de un fluido frío
- Llevar al punto de ebullición a un fluido mientras se condensa un fluido gaseoso con mayor temperatura.

Se presentan los intercambiadores de calor en función a su aspecto constructivo:

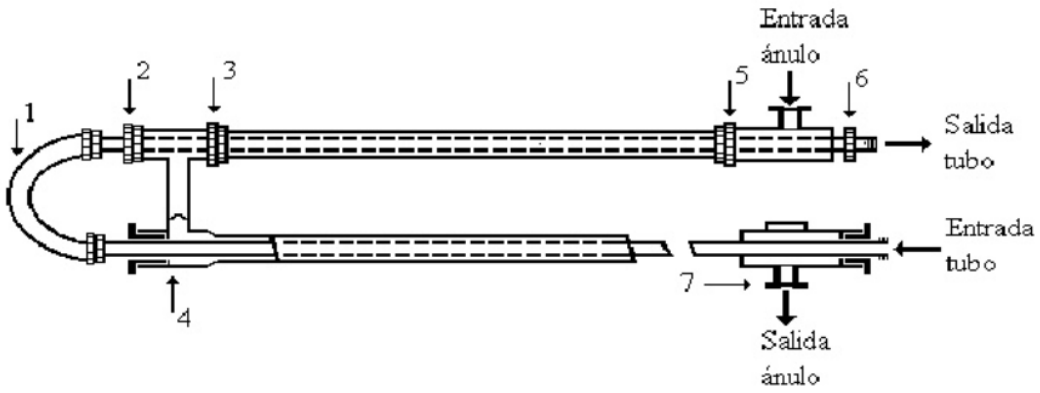


Figura 3.3: Intercambiador de calor de doble tubo.

1-Codo. 2, 3, 5, 6-Prensa estopa. 4-Cabezal de retorno. 7-Tee

Doble tubo (tubos concéntricos): Consiste en 2 tubos concéntricos en donde una corriente circula por dentro del tubo interior mientras que la otra corriente circula por el ánulo formado entre los tubos. Son fáciles de fabricar (estandarización y construcción modular) obteniendo superficies de transferencia de calor a un coste muy bajo. La principal desventaja radica en la pequeña superficie de transferencia, por lo que si se emplea en procesos industriales, generalmente se va a requerir un gran número de éstos conectados en serie, lo que involucra gran espacio físico en la planta. Los gastos de mantenimiento y desmantelación son muy altos. Se usan cuando la superficie total de transferencia requerida es pequeña (100 a 200 ft2). Operan con altas presiones.

Coraza y tubo: Es el más utilizado en refinerías y plantas químicas.

a) Proporciona flujos de calor elevados en relación con su peso y volumen.

b) Es relativamente fácil de construir en una gran variedad de tamaños.

c) Es bastante fácil de limpiar y de reparar.

d) Es versátil y puede ser diseñado para cualquier aplicación.

Consiste en una coraza cilíndrica que contiene un arreglo de tubos paralelo al eje longitudinal de la coraza. El flujo de fluido dentro de los tubos se llama flujo interno y el que fluye en el interior de la coraza se denomina flujo externo. Los tubos pueden o no tener aletas y están sujetos en cada extremo por laminas perforadas. Estos atraviesan a su vez a una serie de láminas denominadas deflectores (baffles) que al ser distribuidas a lo largo de toda la coraza, sirven para soportar los tubos y dirigir el flujo que circula por la misma, de tal forma que la dirección del flujo sea siempre perpendicular a los tubos.

El fluido que va por dentro de los tubos es dirigido por unos ductos especiales conocidas como cabezales o canales.

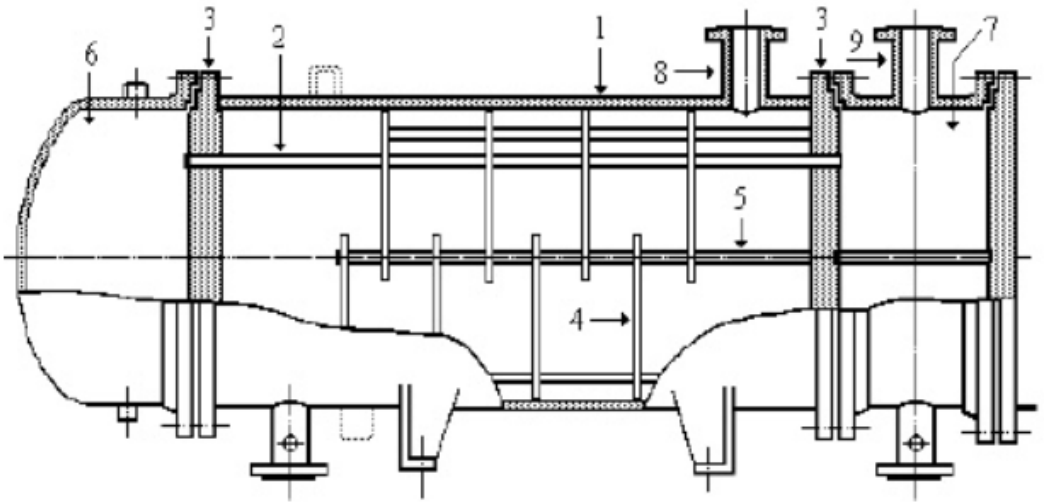


Figura 3.4: Intercambiador de tubo y coraza.

1-Coraza. 2-Tubos. 3-Placa de tubos. 4-Deflectores. 5-Deflector longitudinal. 6-Cabezal posterior. 7-Cabezal fijo. 8-Boquilla de la coraza. 9-Boquillas para los tubos

Placas: Se disponen de dos tipos: Placas empacas (Plate Heat Exchanger, PHE) y en Espiral (Spiral Heat Exchanger, SHE).

Consiste en una sucesión de láminas de metal armadas en un bastidor y conectadas de modo que entre la primera y segunda placa circule un fluido, entre la segunda y la tercera, otro, y así sucesivamente. Estas placas están separadas por juntas, fijadas en una coraza de acero. La circulación de estos fluidos puede tener diferentes configuraciones, en paralelo y contracorriente. Si el fluido frío circula por la parte de delante de la placa, el fluido caliente lo hace por la parte de atrás.

En los intercambiadores SHE, los flujos trabajan en espiral. Estos equipos son apropiados para trabajar con fluidos de alta viscosidad. Son fácilmente desmontables para labores de mantenimiento. Las condiciones de operación se encuentra limitadas por las empacaduras (T=250 ºC, p=30 bar).

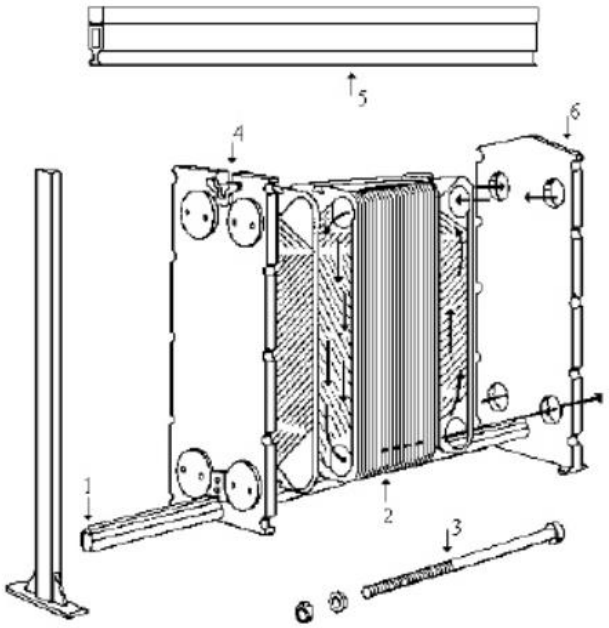


Figura 3.5: Intercambiador de placas empacadas (PHE).

1-Barra de soporte. 2-Conjunto de placas y empacaduras. 3-Perno para compresión. 4-Cubierta móvil. 5-Barra de soporte. 6-Cubierta fija

Máquina de absorción

La refrigeración por absorción es un medio de producir frío que aprovecha que las sustancias absorben calor al cambiar de estado, de líquido a gaseoso. El ciclo se basa físicamente en la capacidad que tienen algunas sustancias, como el bromuro de litio, de absorber otra sustancia, tal como el agua, en fase de vapor. Otra posibilidad es emplear el agua como substancia absorbente (disolvente) y amoníaco como substancia absorbida (soluto). Según la temperatura del foco caliente las clasificamos en dos tipos:

- ✓ Absorción simple efecto: La fuente térmica es agua caliente a una temperatura entre 80 y 95ºC, obteniéndose un COP del orden del 0,7.

- ✓ Absorción doble efecto: La fuente térmica puede ser vapor de agua, gases de escape de motores o turbinas o, incluso, fuego directo, obteniéndose un COP hasta de 1,4

Dicho esto, solo se explicará la de simple efecto porque es la que se utilizará para modelar la instalación.

Máquina de ciclo de efecto simple con par bromuro de litio/agua: En el ciclo que se describe a continuación se utiliza como fuente de energía el calor contenido en un circuito de agua caliente procedente de la refrigeración de un motor alternativo de una planta de cogeneración pero que es válido para cualquier sistema de recuperación de calor. El fluido utilizado en el ciclo de refrigeración, es una solución de agua y Bromuro de litio (LiBr), siendo el agua el refrigerante y el LiBr el absorbente. El LiBr es una sal que tiene una gran afinidad con el agua, absorbiéndola fácilmente.

El otro aspecto importante para entender cómo puede utilizarse el agua como refrigerante, es saber que ésta, cuando se encuentra en un espacio en el que la presión absoluta está muy por debajo de la atmosférica y que en este caso es de únicamente de 0,9 kPa, el agua se evapora (hierve) a tan solo 3ºC.

Para explicar el funcionamiento se seguirá la Figura 3.5. Se pondrá unos valores orientativos en algunas variables para entender lo que pasa en el equipo.

Empezamos en el generador, que está situado en la parte superior izquierda de la figura, donde la solución acuosa (denominada solución diluida) contiene un 52% de LiBr. Por el circuito primario del generador circula el agua caliente que aporta la energía necesaria para hacer funcionar el sistema. Esta agua caliente entra nominalmente a una temperatura de 88ºC en el circuito primario del generador saliendo de él a 83ºC. Mientras, en el circuito secundario del generador, o sea en el circuito de refrigeración, la presión absoluta es de 13 kPa. Por efecto del calor aportado por el circuito primario de agua caliente, el agua de la solución diluida entra en ebullición y el vapor formado se encamina hacia el recipiente contiguo que es el condensador. Debido a esta separación de vapor, la solución restante, denominada solución concentrada, se concentra hasta un 56% de LiBr, dirigiéndose en estas condiciones hacia el intercambiador de calor situado en la parte inferior del esquema.

Mientras, en el condensador, el vapor de agua es enfriado hasta 36ºC gracias al circuito de agua procedente, por ejemplo, de una torre de enfriamiento (equipo que se utilizará en nuestra modelo) y que entra a la máquina a una temperatura de 31ºC, condensando el vapor de agua y convirtiéndolo en líquido. Este líquido refrigerante, es introducido por diferencia de presión en el evaporador donde se mantiene una presión absoluta de 0,9 kPa, por lo que se evapora a 3ºC, adquiriendo el calor necesario para ello del circuito de agua a refrigerar, rebajando su temperatura a 7ºC suponiendo que ha entrado a la instalación a 12ºC.

Al mismo tiempo, la solución concentrada al 56% de LiBr procedente del generador fluye hacia el absorbedor, que comparte espacio y presión con el evaporador, siendo el vapor de agua contenido en este espacio absorbido por el LiBr debido a su afinidad con el agua, diluyendo la concentración de LiBr de nuevo al 52%. Ello permite eliminar el vapor a medida que se produce y continuar manteniendo la presión de 0,9 kPa en el espacio compartido por el evaporador y el absorbedor.

El fenómeno de la absorción produce calor que a su vez es eliminado por el mismo circuito de enfriamiento antes de dirigirse al condensador.

Finalmente, la solución diluida al 52% de LiBr por la absorción del vapor, es aspirada por la bomba de solución (SP) para enviarla de nuevo al generador donde se reinicia el proceso, pasando previamente por el mismo intercambiador de calor de antes por dónde paso la solución concentrada procedente del generador, aumentando el rendimiento del ciclo.

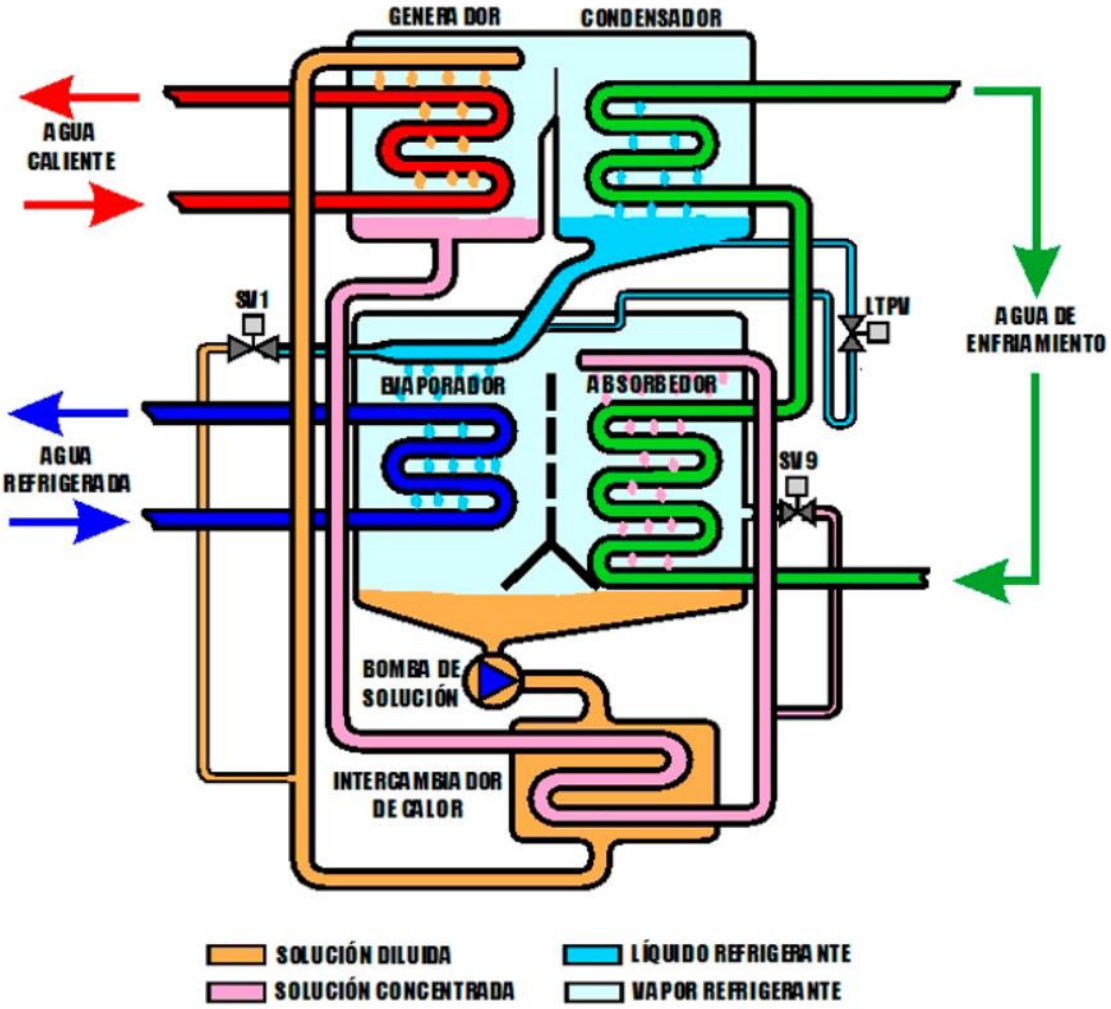


Figura 3.6: Esquema del ciclo de absorción LiBr-H_2O de simple efecto

Torre de refrigeración

El funcionamiento de una torre de refrigeración es el siguiente: Un flujo de agua caliente está en contacto directo con un flujo de aire de forma que el primero es refrigerado como resultado de una transferencia de calor sensible debido a la diferencia de temperatura entre ambos y de una transferencia de masa hacia el aire como resultado de la evaporación. La torre de refrigeración puede ser de flujo contracorriente o de flujo cruzado, dependiendo de la configuración de los flujos de agua y de aire. El flujo de aire ambiente que entra en la torre de refrigeración asciende a lo largo de la misma mientras que el flujo de agua caliente asciende. Generalmente, las torres de refrigeración están compuestas de varias celdas en paralelo que comparten un sumidero común. Las pérdidas de agua que se producen en las celdas de la torre son compensadas con agua de red al sumidero.

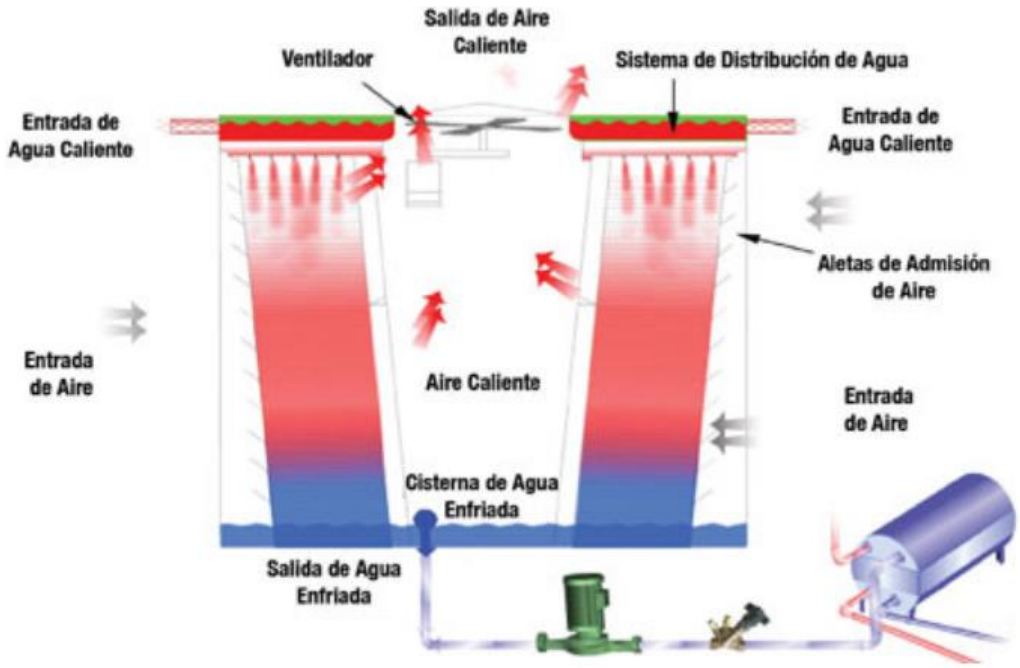


Figura 3.7: Esquema funcionamiento torre de refrigeración

Aerotermo (bomba de calor aire-agua)

Los dos equipos que se describen ahora, son los utilizados como sistemas auxiliares de apoyo para cubrir las demandas de calefacción o refrigeración. En términos generales, las bombas de calor usan energía procedente del aire, del suelo o del agua subterránea. Constan de un circuito compuesto por un evaporador, un compresor, un condensador y una válvula de expansión. El fluido refrigerante que circula por ese circuito, y que es la base de la bomba, está a baja temperatura y a baja presión, y por tanto está en estado líquido. Al conectar la bomba, el medio (agua, aire...) pasa a través del evaporador rodeando el punto donde está el refrigerante y absorbe el calor presente en medio, evaporándose.

El refrigerante en forma de gas a baja presión, entra en el compresor. El compresor se encarga de aumentar la presión y la temperatura. Posteriormente entra en el condensador y cede su calor al agua o aire que lo rodea, calentándolo para enviarlo dentro de una habitación por ejemplo. El refrigerante ha bajado su temperatura y vuelve a estado líquido. El refrigerante aún está demasiado caliente y a mucha presión para que se vuelva a repetir el ciclo. Por lo tanto, se pasa por una válvula de expansión para reducir su presión, y su temperatura volviéndose a repetir el ciclo.

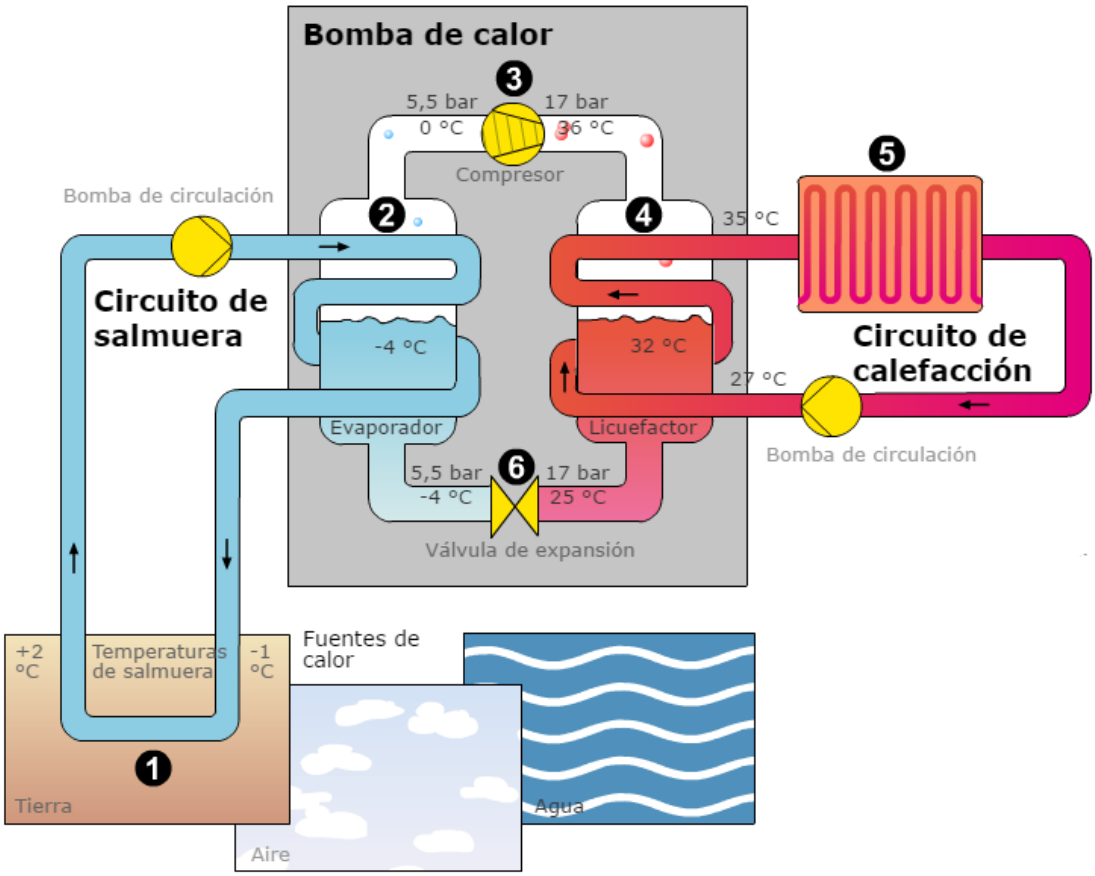


Figura 3.8: Esquema básico funcionamiento de una bomba de calor

Un aerotermo es la combinación de una bomba de calor con un acumulador que utiliza o extrae energía del aire circundante y lo transforma en energía térmica para entregársela al agua. Dicha energía puede ser distribuida a otros sistemas como radiadores, unidades de tratamiento de aire, suelo radiante o fancoils…que a su vez ceden el calor de ese agua al ambiente.

También pueden funcionar en modo refrigeración. En el intercambiador exterior se cede el calor del agua al aire, y en el intercambiador interior se absorbe el calor del ambiente calentando el agua.

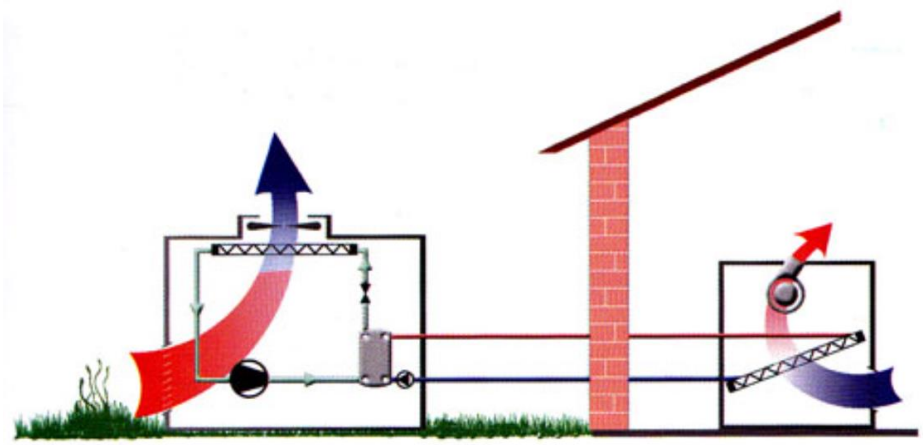


Figura 3.9: Esquema simplificado de una bomba de calor aire-agua

Como se deduce, este sistema se compone de dos unidades. La unidad exterior consta de un compresor hermético modulante por frecuencia con tecnología invertir DC, válvulas de expansión, válvulas de cuatro vías para configurar su funcionamiento reversible, un intercambiador de aletas de alto rendimiento aire-agua que funciona como condensador o como evaporador, dependiendo del modo de operación de la bomba (calefacción o refrigeración), por donde circula el refrigerante absorbiendo o cediendo temperatura. La unidad interior consta de otro intercambiador completamente aislado del exterior donde el refrigerante que circula por este circuito hermético cede o absorbe calor del ambiente del circuito primario del interior de las estancias a climatizar.

Las Bombas de Calor aire/agua, con mayores índices de rendimiento y compatibles con otros sistemas de calentamiento ya existentes, se caracterizan por su fácil instalación y mantenimiento.

Las Bombas de Calor aire-agua que permiten un abastecimiento térmico libre de emisiones de CO2 en el punto de consumo, tampoco utilizan combustibles líquidos o gaseosos, por lo que no requieren adaptarse a las condiciones limitadoras de otros generadores que utilizan estos combustibles convencionales ni seguir pautas en la evacuación de gases de la combustión facilitando su instalación e integración.

Bomba de calor agua-agua

La bomba de calor agua-agua utiliza como fuente de extracción el agua subterránea de la capa freática, lago, río o subsuelo e intercambian calor tanto con el exterior como con el interior mediante un circuito de agua. Incluso en invierno, la temperatura de las aguas subterráneas se sitúa entre +7°C y +12°C, temperatura suficiente para abastecer una bomba de calor para la calefacción de una casa o de un inmueble. Numerosos edificios situados en las zonas más frías de Europa, dónde la temperatura exterior del aire alcanza

los – 15°C, utilizan desde hace muchos años esta técnica de captación y deposito en la capa freática obteniendo coeficientes de rendimiento muy interesantes (COP sup. a 3).

En modo calefacción, toman el calor del circuito exterior y producen agua caliente en el circuito interior para calentar de forma indirecta, mediante fancoils, radiadores, suelo radiante, etc.

El funcionamiento en modo refrigeración (bomba de calor reversible) utiliza también el agua de la capa freática, que con un ciclo inverso de la bomba de calor causa un ligero recalentamiento del agua de la capa freática en verano y consigue rendimientos excelentes, bastante superiores a los rendimientos obtenidos con una tecnología aire-agua o aire-aire. Las bombas de calor agua-agua montan dos intercambiadores refrigerante-agua.

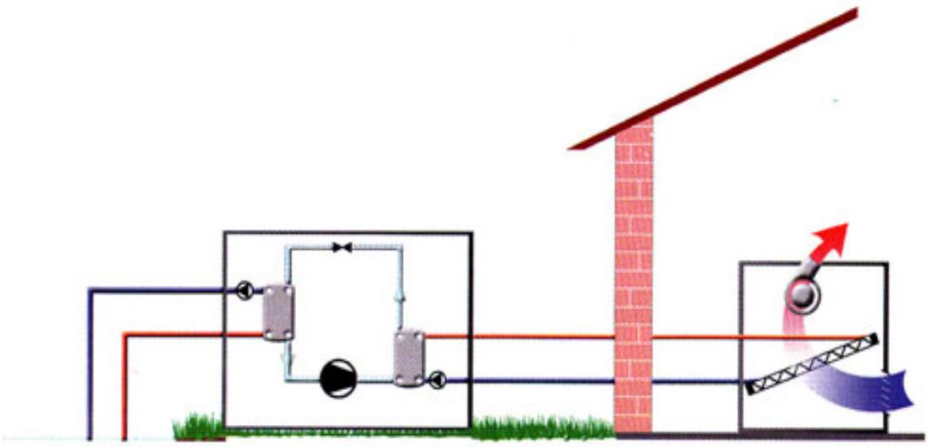


Figura 3.10: Esquema simplificado de una bomba de calor agua-agua

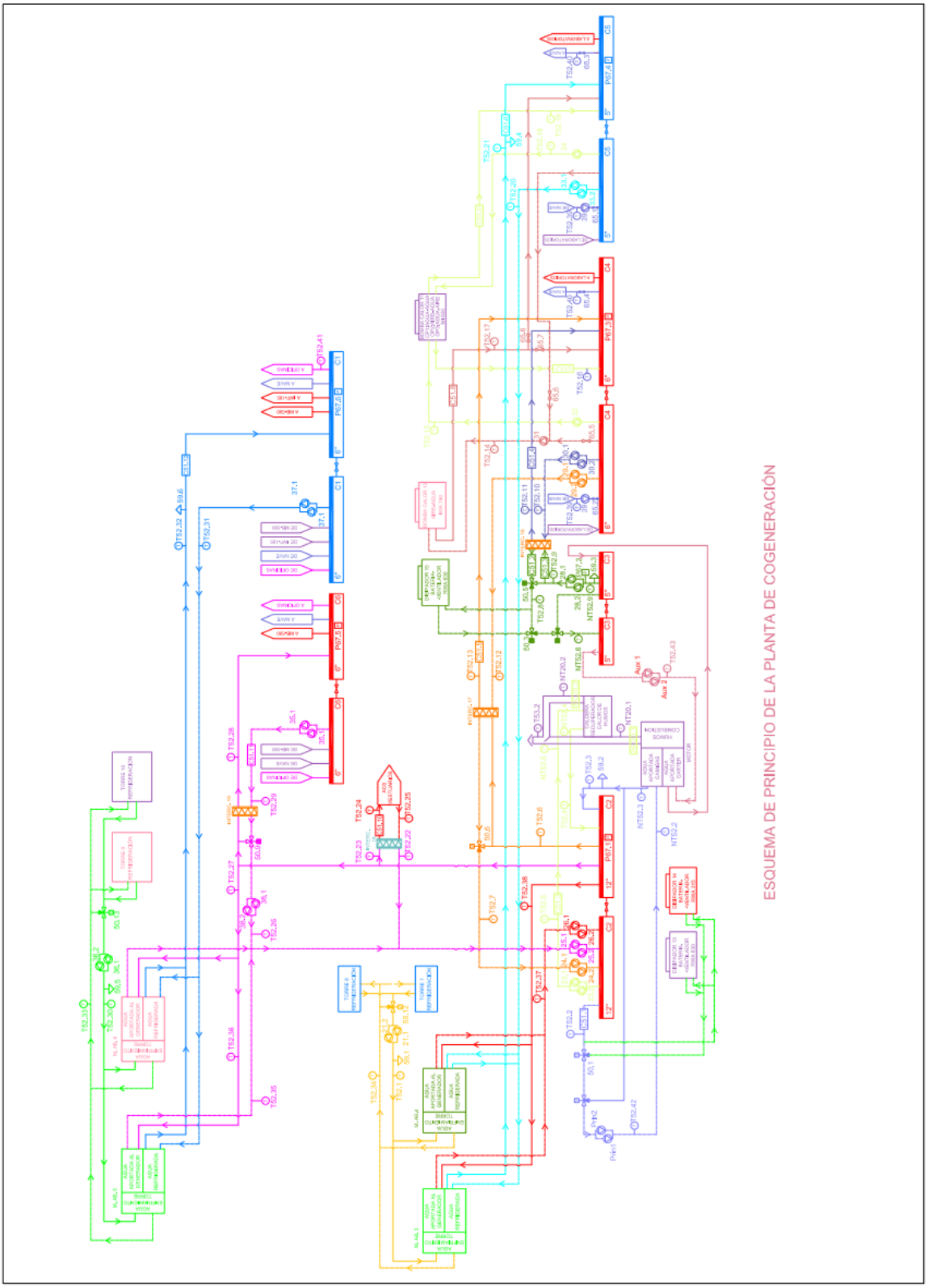
La instalación de la bomba de calor agua-agua se efectúa en un local técnico, (bodega, sótano por ejemplo). Es necesario cavar dos pozos, uno de captación y otro de restitución, o un pozo agrupando la captación y la restitución del agua. En el primer pozo, el pozo de aspiración, la bomba bombea el agua para absorber sus calorías y en el segundo, el pozo de restitución, retorna el agua enfriada (y recalentada en verano) en la capa freática.

3.3. Planta de cogeneración CIATESA

Como ya se ha dicho, la instalación de cogeneración a modelar es de una planta que se encuentra en CIATESA. En el Anexo A: Datos Planta Cogeneración aparecen los datos aportados por CIAT. En dicha documentación gráfica se observa los componentes de los que consta la instalación, como están conectados entre ellos y valores de algunas variables que son de suma importancia para el estudio de la planta. Se recogerá toda la información posible de las distintas líneas de la instalación que sea de utilidad para el objetivo de este TFG.

3.4. Análisis de la instalación

3.4.1. Esquema resultante



3.4.2. Descripción del esquema

Un primer acercamiento para entender el funcionamiento de la planta es el siguiente: La instalación consta de un motogenerador de gas natural del cual se recupera calor a través de la refrigeración de su circuito principal (agua aportada a camisas) como de la caldera de recuperación de los gases de escape, y de su circuito auxiliar (agua aportada a cárter). Al circuito de refrigeración de camisas como al de la caldera de recuperación lo llamaremos **Circuito de Alta Temperatura**. Al circuito auxiliar se le llamará **Circuito de Baja Temperatura.** Dichos nombres son dados porque la recuperación de calor ocurre a más baja temperatura en el circuito auxiliar que en el otro.

El Circuito de Alta Temperatura llega a un colector común dónde se dirige a un subsistema de intercambio que cubre cargas de refrigeración (a través de 4 máquinas de absorción de simple efecto con torres de refrigeración), como de calefacción (mediante 3 intercambiadores de calor). El Circuito de Baja Temperatura es conducido a un subsistema de intercambio que solo cubre cargas de calefacción (a través de 1 intercambiador de calor).

La instalación cuenta además con un subsistema de apoyo convencional basado en bombas de calor (BdC) para cubrir las cargas de calefacción o refrigeración cuando la instalación de cogeneración no sea suficiente para cubrir dichas cargas. En concreto, existen dos bombas de calor: una BdC aire-agua y una BdC que puede trabajar tanto en modo agua-agua como en modo aire-agua.

Para intentar facilitar un estudio de la planta más detallado es conveniente separar los circuitos en:

- ✓ **Circuito de Generación de Alta y Baja Temperatura**
- ✓ **Circuito Primario**
- ✓ **Circuito Secundario y de Consumo de Alta y Baja Temperatura**

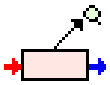
Para intentar separar el esquema en los respectivos circuitos se ha utilizado el conocido software de simulación TRNSYS, pero con fines de facilitar la separación y visualización de los circuitos para su explicación y no con intención de simular la planta, ya que la modelación de la planta se hará con la suite OpenModelica. Antes de proceder a la descripción de los diferentes circuitos se nombran los componentes del sistema.

Los componentes de los que consta la instalación de cogeneración quedan recogidos en la siguiente tabla:

Tabla 3.1: Componentes instalación de cogeneración con ilustraciones de TRNSYS

Motogenerador de gas con caldera de recuperación (refrigeración por agua)	Camisas y Gases de escape	
	Cárter	

Colectores	C1 ida/retorno	
	C2 ida/retorno	
	C3 ida/retorno	
	C4 ida/retorno	
	C5 ida/retorno	
	C6 ida/retorno	
Intercambiadores de calor	IC 16	
	IC 17	
	IC 18	
	IC 19	
Máquinas de absorción	MA 3	
	MA 4	
	MA 5	
	MA 6	
Torres de refrigeración	TR 7	
	TR 8	
	TR 9	
	TR 10	

Disipadores	Dis 13 (RWA 630)	
	Dis 14 (RWA 315)	
	Dis 15 (RWA 630)	
Bombas de calor	BOMBA CALOR 11 AGUA-AGUA AIRE-AGUA (MI 630)	
	BOMBA CALOR 12 AIRE-AGUA (IWA 740)	

Las bombas de calor no se han implementado en la separación de los circuitos para que la visualización sea lo más simplificada posible y porque son sistemas de apoyo convencionales cuya función es conocida. En la separación de los circuitos se ha prescindido del colector C3 por motivos de simplificación.

Existen algunos elementos que no se han nombrado ya que no son de suma importancia para explicar el funcionamiento de la planta. Aunque no se citarán para describir la instalación se pondrán en la siguiente tabla para que se identifiquen claramente.

Tabla 3.2: Componentes hidráulicos de la instalación de cogeneración

Bombas de circulación	Bombas de agua cuyo caudal es constante	
Uniones (en líneas de tuberías)	Dos flujos de entrada son mezclados en único flujo de salida	
Desviadores (en líneas de tuberías)	Un flujo de entrada es proporcionalmente dividido en dos flujos de salida	
Válvulas 3-vías (atemperamiento)	Función de una válvula de 3 vías con control de temperatura	

Líneas de tuberías	Modela el comportamiento térmico de un fluido en una tubería	

Los puntos de consumo son los siguientes:

- Laboratorios
- Nave
- ACS/Vestuarios (No existe actualmente)
- Oficinas
- MB-590
- IWP-185

En las imágenes posteriores no aparecerán los puntos de consumo ya que simplemente el circuito iría desde los colectores finales de ida hasta el sitio de demanda energética y volvería hasta los colectores de retorno.

Antes de adentrarnos con los diferentes circuitos, mencionar que las líneas que aparecen para conectar los diferentes elementos aparecen de dos formas: en continua y en discontinua, para reflejar la dirección del fluido de trabajo (ida y retorno).

Circuito de Generación de Alta y Baja Temperatura

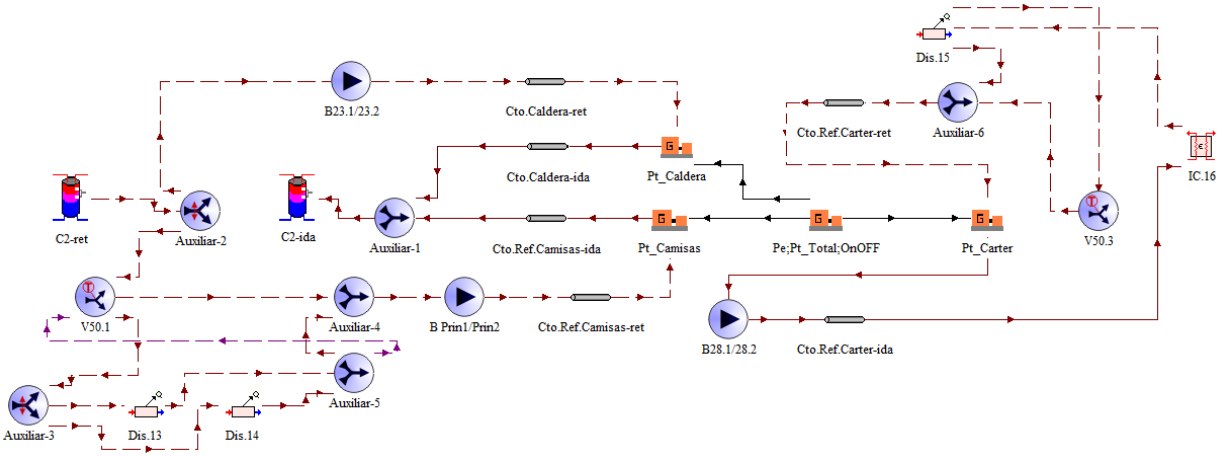


Figura 3.11: Esquema circuito de generación de alta y baja temperatura elaborado con TRNSYS

Como ya se ha comentado, se parte de un motogenerador que utiliza como combustible gas natural capaz de producir energía eléctrica. La energía eléctrica puede servir para consumo propio o para venderse a la red eléctrica. El calor inherente producido en el proceso se intenta aprovechar. Dicha energía térmica procede de las camisas del motor, del cárter y de los gases procedentes de la combustión que son recuperados por una caldera. Una parte del calor no se puede aprovechar y es vertido al ambiente. La energía térmica es recuperada a través de unos circuitos refrigeradores en los que se produce una transferencia de calor de dichos elementos hasta el fluido de trabajo, que en este caso es agua. Al circuito de refrigeración de camisas como al de la caldera de recuperación lo llamamos Circuito de Alta Temperatura. Al circuito auxiliar (cárter), Circuito de Baja Temperatura. Dichos nombres son dados porque la recuperación de calor ocurre a más baja temperatura en el circuito auxiliar que en el otro.

Por lo tanto, del motor salen 3 conductos: Cto.Caldera-ida, Cto.Ref.Camisas-ida y Cto.Ref.Cárter-ida. En las condiciones de operación dadas por CIAT en sus fotografías, las temperaturas del fluido caloportador son de 90.4ºC, 89.7ºC y 59.9ºC respectivamente.

El fluido procedente de la caldera como el de las camisas va directamente hacia el colector (C2-ida). El circuito de ida del Cárter se conecta con un intercambiador de calor (IC.16). Una vez producido, en dicho intercambiador, una trasferencia de energía térmica con un circuito primario que se describirá posteriormente, el fluido regresa hacia un disipador (Dis.15). La función del disipador es la de reducir la temperatura del caudal de retorno si el valor de ésta es superior a uno de consigna. Desde el disipador, el fluido regresa hacia el circuito refrigerador del Cárter (previamente existe una válvula de 3 vías pero como se mencionó en apartados anteriores no se citarán elementos como bombas de circulación, válvulas, etc, por la insignificancia que supone para el entendimiento del esquema).

Para explicar el Cto.Caldera-ret y Cto.Ref.Camisas-ret, se partirá del colector de retorno C2-ret. Del C2-ret se separan dos líneas de circuito: uno va directamente hacia el circuito de refrigeración de la caldera de recuperación y el de camisas pasa previamente por dos disipadores (Dis.13, Dis14). Al igual que con el disipador 15, su función es la de reducir la temperatura del caudal de retorno si el valor de ésta supera a uno de consigna.

Una vez pasada el fluido por ambos disipadores vuelven hacia el circuito refrigerador de las camisas del motor.

Para concluir con la explicación del circuito de generación, indicar que las temperaturas aproximadas del fluido de trabajo por los circuitos de retorno según las condiciones de operación de las fotografías son de: 85.8ºC para Cto.Caldera-ret, 81.9ºC para Cto.Ref.Camisas-ret y 54.9ºC para Cto.Ref.Cárter-ret.

Circuito Primario

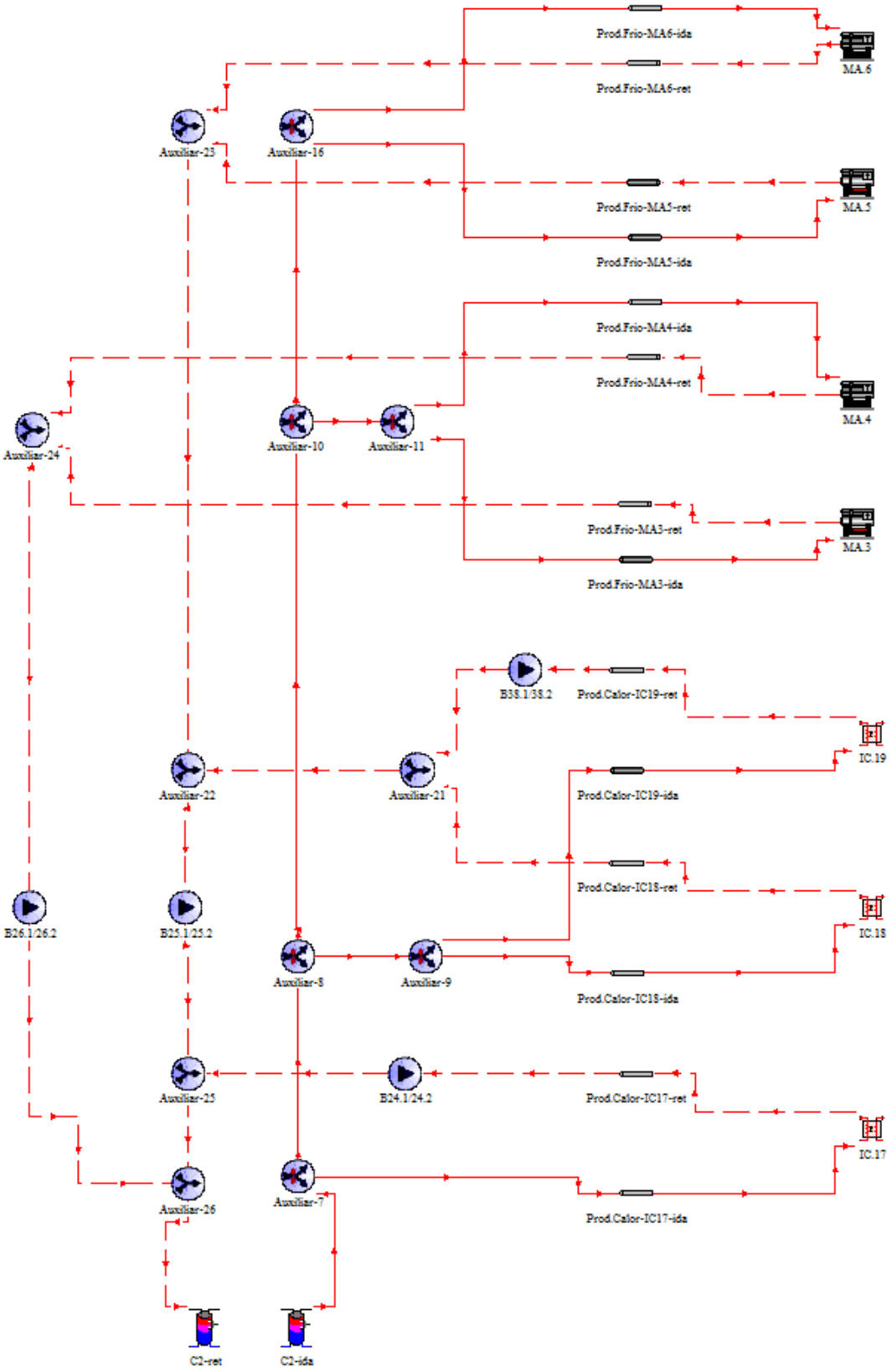


Figura 3.12: Esquema circuito primario elaborado con TRNSYS

Como se puede observar en la fotografía el circuito primario solo corresponde con el circuito de alta temperatura, ya que el circuito de baja temperatura (cárter), se conectaba directamente al intercambiador de calor 16 (IC.16), por lo que según las fotografías otorgadas por CIAT, el circuito de baja temperatura solo cubrirá cargas de calefacción.

Partimos del colector de ida (C2-ida). Dicho colector va a nutrir por un lado, a 3 intercambiadores de calor (IC.17, IC.18, IC.19), y por otro lado, a 4 máquinas de absorción de simple efecto (MA.3, MA.4, MA.5, MA.6). Dado el comportamiento de estos equipos, los intercambiadores de calor están destinados para cubrir cargas de calefacción, y las máquinas de absorción a cubrir cargas de refrigeración. No se va a esclarecer todas las conexiones de este circuido de "ida" hacia estos equipos, ya que en la figura 3.12 se ven con total claridad. Al igual que con el intercambiador 16 del circuito de generación, en los intercambiadores 17,18 y 19 se producirá una transferencia de calor con el fluido procedente de otro circuito, al otro lado del intercambiador. Una vez producido el intercambio de calor, el fluido vuelve de los intercambiadores hacia el colector de retorno (C2-ret).

En el caso de las máquinas de absorción el proceso es semejante. Lo que pasa que el funcionamiento de estas máquinas es algo más complejo que la de los intercambiadores. De todas formas, el funcionamiento de una máquina de absorción de simple efecto ya se ha explicado con anterioridad en una sección de este trabajo y se tratará como una "caja negra". Por lo tanto, decir, que el agua caliente procedente del colector de ida va hacia el generador de dichas máquinas. Al igual que con los intercambiadores no se van a describir todas las interconexiones del circuito de ida porque se ven claramente en la figura 3.12. El agua procedente del generador de las máquinas de absorción regresa hasta el colector de retorno (C2-ret), al igual que pasaba con los circuitos de retorno de los intercambiadores.

A nivel orientativo, para observar lo que le pasa al fluido caloportador se expondrán las temperaturas en diferentes puntos del circuito primario. Las temperaturas del fluido caloportador por dichos conductos son aproximadamente los siguientes según las condiciones de operación dadas por CIAT:

- Prod.Calor-IC17-ida: 88.8ºC Prod.Calor-IC17-ret: 86.7ºC
- Prod.Calor-IC18-ida: -10ºC Prod.Calor-IC18-ret: 0ºC
- Prod.Calor-IC19-ida: 89.7ºC Prod.Calor-IC19-ret: 88.5ºC
- Prod.Frío-MA3-ida: 88.6ºC Prod.Frío-MA3-ret: 86.3ºC
- Prod.Frío-MA4-ida: 88.6ºC Prod.Frío-MA4-ret: 86.3ºC
- Prod.Frío-MA5-ida: 76.1ºC Prod.Frío-MA5-ret: 71.3ºC
- Prod.Frío-MA6-ida: 76.1ºC Prod.Frío-MA6-ret: 71.3ºC

Mencionar que con IC18 pasa una peculiaridad que se visualizará en el circuito secundario, y por eso existen esas anomalías en las temperaturas.

Circuito Secundario y de Consumo de Alta y Baja temperatura

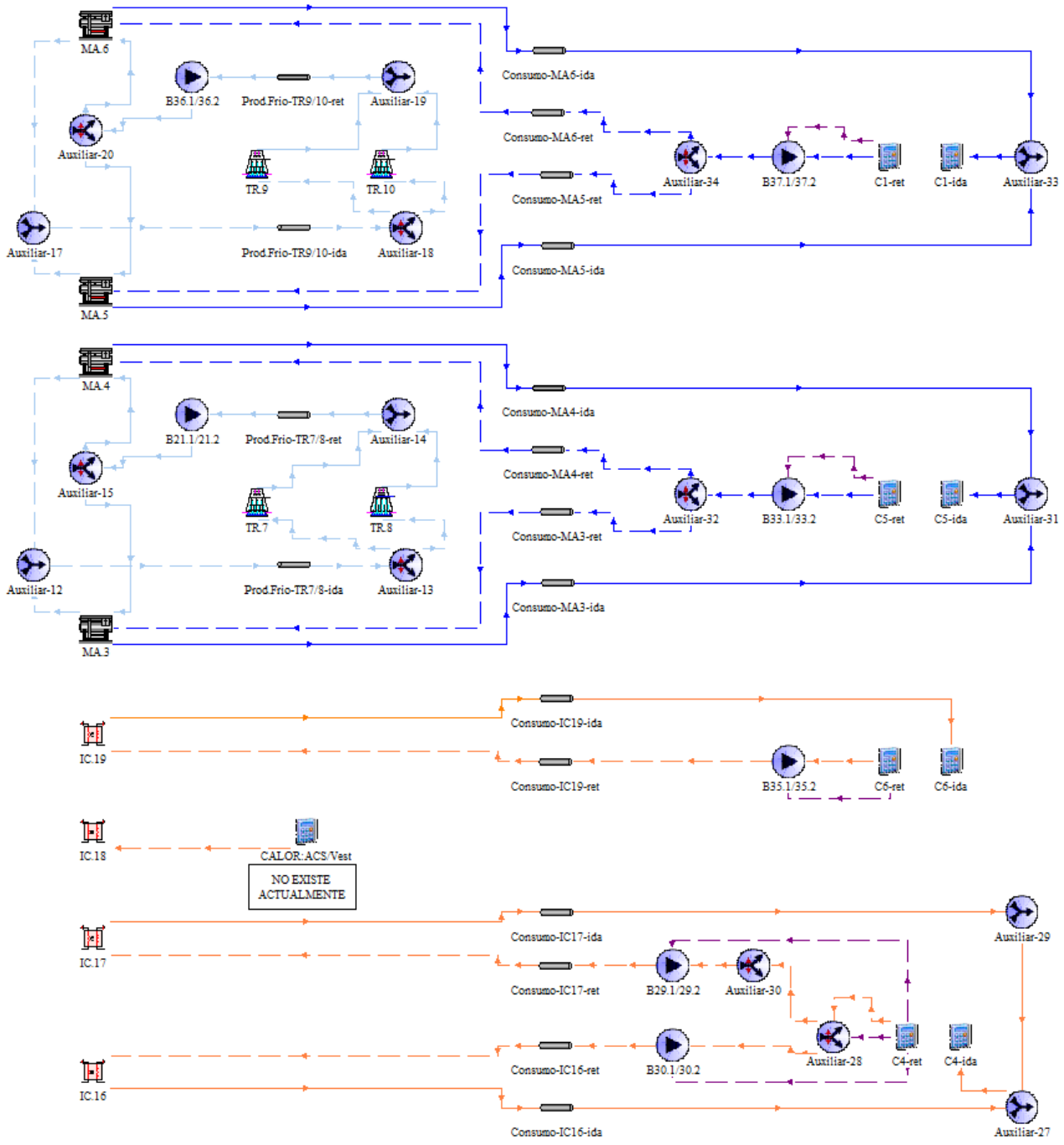


Figura 3.13: Esquema circuito secundario y de consumo de alta y baja temperatura elaborado con TRNSYS

En primer lugar, como ya se mencionó anteriormente, en el intercambiador 18 aparece un punto de consumo que corresponde a ACS/Vestuarios. En el esquema aparece porque en los datos entregados por CIAT está presente por intenciones futuras, pero actualmente no existe y por lo tanto no hay datos reales.

Continuando con los intercambiadores, veamos el circuito al otro lado del intercambiador, que es por dónde pasa el fluido que intercambia energía térmica con el circuito primario (parte de calefacción). Como ya se dijo, el intercambiador 16 corresponde al circuito de baja temperatura y el 17,18, y 19 al de alta temperatura. Las conexiones son muy sencillas. El fluido parte desde el IC.16 y 17 para entrar en el colector de ida (C4-ida). De aquí se dirige para los respectivos puntos de consumo. El colector de retorno C4-ret recoge el agua procedente de los puntos de consumo y los entrega de vuelta a los intercambiadores respectivos para que se siga repitiendo el ciclo. Con el IC.17 ocurre exactamente los mismo utilizando el colector de ida y retorno C6.

Ahora, el funcionamiento de la parte del circuito secundario que se encarga de las cargas de refrigeración (recordar que solo corresponde al circuito de alta temperatura), es un poco más complejo debido al funcionamiento de las máquinas de absorción. Como se puede ver en la figura 3.13 el circuito presenta la misma morfología para todas las máquinas de absorción. Empecemos por el circuito de enfriamiento, que es el que se suministra al condensador y absorbedor de dichas máquinas. Como ya se mencionó anteriormente el funcionamiento de una máquina de absorción ya se explicó en apartados anteriores de este trabajo, solo decir que este circuito de agua procede de una torre de refrigeración (cuyo funcionamiento también esta explicado en secciones anteriores). En la figura 3.13 se ve claramente este circuito. Está de color celeste. Para resumirlo brevemente: el agua enfriada procedente de las torres de refrigeración TR.7 y TR.8, TR.9 y TR.10 entra en el absorbedor de sus máquinas correspondientes, MA.3 y MA.4, MA.5 y MA.6, y sale del condensador de dichas máquinas para entrar en las torres para enfriar el agua y repetir el ciclo.

Por último, está el circuito de agua refrigerada (color azul). El agua procedente de los colectores de retorno C5-ret y C1-ret es dirigida hacia el evaporador de las máquinas de absorción. Allí el agua es enfriada y se dirige para el colector de ida C5-ida y C1-ida para cubrir las cargas de refrigeración. Para entender de una forma más clara las transformaciones que experimenta el fluido caloportador, se exponen a continuación temperaturas en puntos de interés (aunque algunos datos no tengan mucho sentido):

- Consumo-IC16-ida: 51.6ºC Consumo-IC16-ret: 49.6ºC
- Consumo-IC17-ida: 52.7ºC Consumo-IC17-ret: 46.8ºC
- Consumo-IC19-ida: 58ºC Consumo-IC19-ret: 55.2ºC
- Prod.Frio-TR7/8-ida: 28.5ºC Prod.Frio-TR7/8-ret: 25.8ºC
- Prod.Frio-TR9/10-ida: 21.5ºC Prod.Frio-TR9/10-ret: 21.8ºC
- Consumo-MA3-ida: 15.4ºC Consumo-MA3-ret: 13.2ºC
- Consumo-MA4-ida: 15.4ºC Consumo-MA4-ret: 13.2ºC
- Consumo-MA5-ida: 22.7ºC Consumo-MA5-ret: 21.6ºC
- Consumo-MA6-ida: 22.7ºC Consumo-MA6-ret: 21.6ºC

Por el circuito de consumo, se entiende que es el recorrido que hace el fluido desde los respectivos colectores finales de ida hasta los puntos de consumo para cubrir las cargas térmicas y volver hasta los colectores finales de retorno como ya se había explicado anteriormente. En cercanías a los puntos de consumo se conectarían las bombas de calor, para servir como sistemas de apoyo para cubrir la demanda energética.

3.4.3. Balance energético y rendimiento

El balance energético global de la instalación puede estudiarse a través de las entradas y salidas de energía en su conjunto vistas externamente. En este caso,

- ✓ la entrada corresponde al calor liberado por el combustible en el motor
- ✓ las salidas, por su parte, se dividen en:
  - energía útil
    - ➢ potencia eléctrica
    - ➢ calor útil para usos de calefacción o refrigeración
  - pérdidas de calor

La representación en forma de diagrama de Sankey es la siguiente:

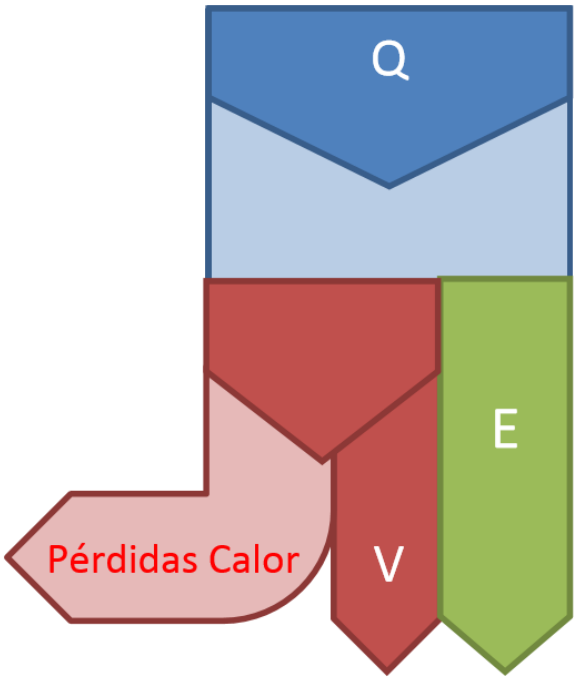


Figura 3.14: Diagrama de Sankey del balance energético global de la instalación

Dónde:

- **Q** es el calor liberado por el combustible
- **E** es la potencia eléctrica generada por el motor
- **V** es el calor útil para calefacción, refrigeración u otros usos
- **Pérdidas de calor** es el calor liberado por el motor y que no se utiliza

Para poder analizar los resultados de supuestas simulaciones térmicas que se quisieran hacer sobre la instalación de cogeneración, conviene definir rendimientos parciales y globales de la instalación.

En primer lugar se define el Rendimiento Eléctrico Equivalente como:

$$REE = E / (Q - V/0.9)$$

Además es definible un rendimiento nuevo, éste sería el Rendimiento Global de la Instalación (**RG**). Expresa el uso energético global, tanto térmico como eléctrico, y tanto por aprovechamiento del calor liberado en la cogeneración como por uso convencional, dividido por el consumo energético.

Así, el uso energético total sería:

- ✓ Eléctrico: **E' = E – Econsumida**, es decir, la energía eléctrica producida neta es igual a la energía producida menos la energía consumida.
- ✓ Térmico: **V + V'**, es decir, la energía térmica utilizada total, tanto la que se aprovecha de la instalación de cogeneración, como la que se produzca por fuentes convencionales.

$$RG = (E' + V + V') / Q$$

Este rendimiento global se puede expresar en función de:

- Fracción de aprovechamiento eléctrico: fE' = E' / Q

- Fracción de aprovechamiento térmico de cogeneración: fV = V / Q

- Fracción de aprovechamiento térmico del sistema de apoyo: fV' = V' / Q

Quedando:

$$RG = fE' + fV + fV'$$

Siendo la relación entre los dos primeros**: REE ≈ fE' / (1 – fV)**

y pudiéndose aproximar el **fV' ≈ fE' * (1 – fE') * COP**

Según el tipo de sistema de apoyo en funcionamiento se definen otros rendimientos, como son el COP de bombas de calor para calefacción, o el EER para refrigeración.

Sistema de apoyo para calefacción: Por ejemplo un sistema de apoyo para calefacción basado en una bomba de calor. La representación en forma de diagrama de Sankey sería:

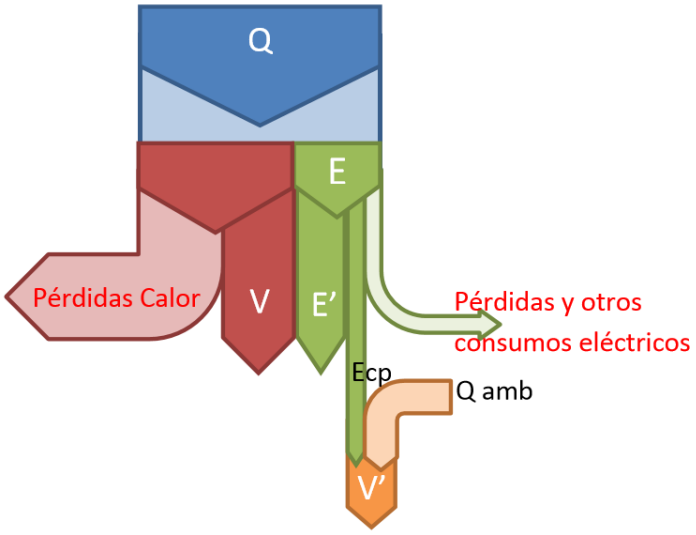


Figura 3.15: Diagrama de Sankey de un sistema de apoyo basado en una bomba de calor (calefacción)

Siendo el rendimiento de la bomba de calor de apoyo: **COP = V' / Ecp**

Sistema de apoyo para refrigeración: Por ejemplo un sistema de apoyo para refrigeración basado en una bomba de calor. La representación en forma de diagrama de Sankey sería:

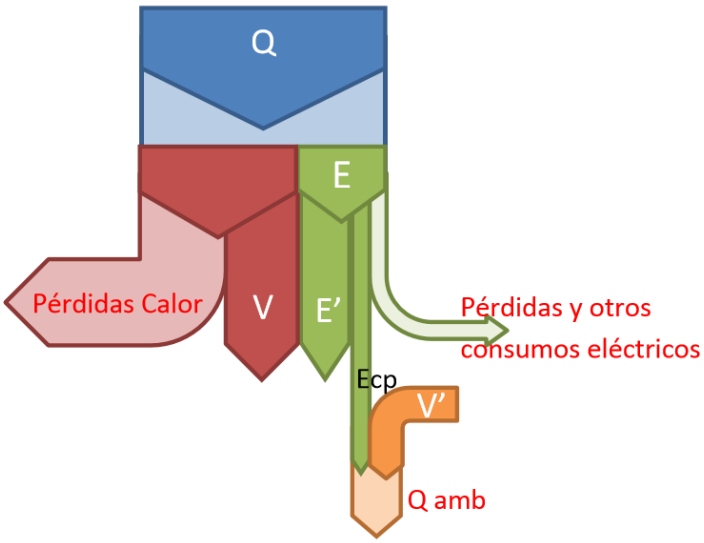


Figura 3.16: Diagrama de Sankey de un sistema de apoyo basado en una bomba de calor (refrigeración)

Siendo el rendimiento de la bomba de calor de apoyo: **EER = V' / Ecp**

3.4.4. Cálculo para un día tipo invierno

Aunque realizar simulaciones sobre el modelo en régimen transitorio con objetivo de entender su comportamiento dinámico y optimizar la planta con estrategias de control no está recogido en el alcance de dicho trabajo, es conveniente entender la curva de potencia demandada prevista para un día tipo de invierno como de verano.

Se puede decir, de manera general, que a lo largo de un día de invierno, las necesidades de calefacción van de más a menos. Alcanzándose la punta de calefacción a primera hora de la mañana, dentro del horario laboral.

La situación típica será la representada en la gráfica, en la que:

- ✓ En las primeras horas la potencia calorífica necesaria de calefacción sea superior a la que puede cubrirse con el sistema de cogeneración. Durante estas horas todo el calor aprovechable de la instalación está siendo útil, pero es necesario utilizar simultáneamente un sistema de apoyo.
- ✓ En un cierto instante, que convendrá alargarlo lo máximo que las exigencias de confort térmico lo permitan, el calor máximo aprovechable del sistema de cogeneración, coincidirá con el calor necesario para calefacción.
- ✓ Durante el resto del día, típicamente a partir de un cierto instante hasta final de la jornada laboral, las necesidades de calor para calefacción son menores que el calor que en principio sería aprovechable del sistema de cogeneración.
- ✓ Como no todas las horas de funcionamiento de la instalación se está utilizando la máxima potencia calorífica disponible, es necesario calcular el área por debajo de la curva (en la gráfica), que es el calor que realmente está siendo útil para calefacción.

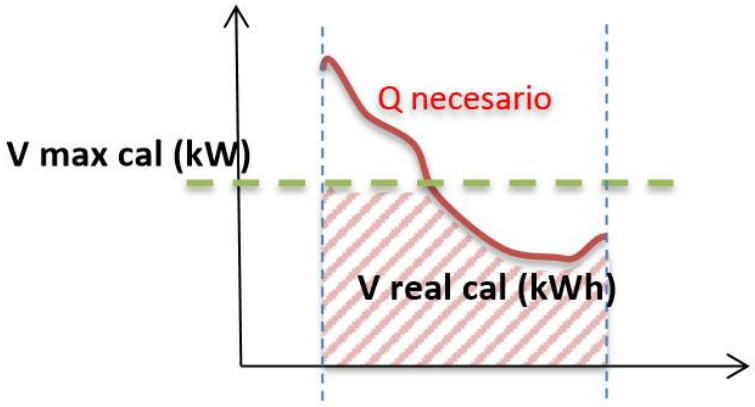


Figura 3.17: Curva de potencia demandada para un día tipo invierno

3.4.5. Cálculo para un día tipo verano

Al igual que en calefacción, en refrigeración pueden darse situaciones diferentes a lo largo de un día, es decir, momentos en los que el frío producido en las máquinas de absorción sea suficiente para cubrir las necesidades, y momentos en que no sea suficiente y sea necesario utilizar un sistema de producción de frío convencional de apoyo.

La particularidad de la demanda de refrigeración en comparación con la de calefacción es que el máximo se puede dar a lo largo del día.

Por otra parte, en la instalación se cuenta con 4 máquinas de absorción de 100kW cada una, es decir, que en el mejor de los casos en refrigeración se llegaría a un V = 400kW. Por lo tanto, el **REE refrigeración** = 745 / (1910-400) = **49%**

Como consecuencia, si en el valor acumulado anual hay que llegar a un REE = 55%, y en verano como máximo se llega a un 49%, deberá compensarse con un valor superior al 55% en invierno como uso de calefacción.

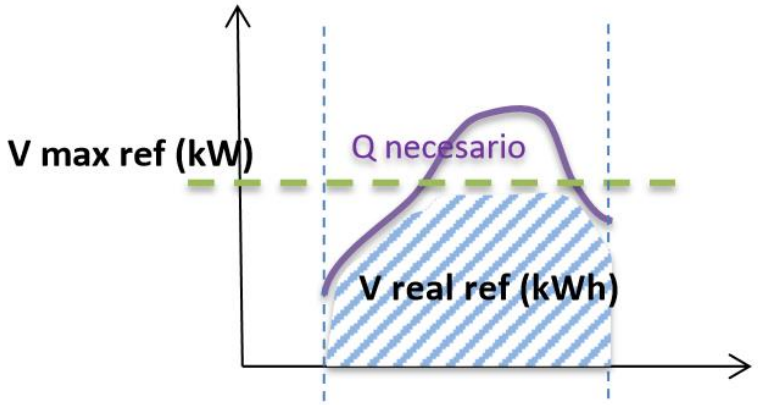


Figura 3.18: Curva de potencia demandada para un día tipo verano

El estudio del comportamiento térmico con variaciones a lo largo de un día de necesidades de calor y/o frío se puede hacer gracias al uso de tablas paramétricas.

4. Modelado de la instalación en OpenModelica

4.1. Introducción

En este capítulo se describirá todos los aspectos fundamentales para montar el modelo de una planta de cogeneración utilizando la suite OpenModelica. Es necesario en primer lugar centrarnos en los componentes que conforman una planta de cogeneración. Para ello es necesario identificar cuáles son los componentes potenciales que se pueden implementar en la planta. Se revisan los parámetros más generales que caracterizan los modelos, se proponen modelos alternativos a los que se proporcionan en las diferentes librerías debido a problemas de validación o funcionamiento de dichos componentes, se propondrá ejemplos de funcionamiento de los modelos finalmente seleccionados, habrá un apartado sobre algunas incidencias y peculiaridades que hay que tener en cuenta para poder hacer funcionar los modelos, y por último, se desarrollará la planta.

La versión de OpenModelica con la que se ha trabajado principalmente es OpenModelica1.11.0-dev.beta3-64bit. Aunque ésta es una versión beta no destinada a un uso productivo, se ha utilizado por la rapidez de compilación y los pocos problemas que ha presentado el programa en general. De todas formas, la última versión para 64 bits disponible en su web: https://www.openmodelica.org/download/download-windows, es la que se recomienda para un uso productivo, pero la lentitud de compilación y diversos problemas de cuelgue del software ha obligado la utilización de la versión beta.

4.2. Models OpenModelica

En este apartado se pondrá, en cada uno de los componentes, los models de las librerías gratuitas disponibles en OpenModelica que podrían cumplir adecuadamente con la función de los componentes reales de la instalación de cogeneración. Cada uno de estos models servirían para desarrollar la planta, aunque como se verá en otro apartado posterior, se han seleccionado unos y no otros, atendiendo a criterios de comodidad conforme a la visualización de sus parámetros, menor tiempo de compilación, errores de simulación, y por intentar utilizar los máximos componentes posibles de la misma librería, debido a problemas de conexionado entre modelos de distintas librerías. Cuando se presentaron las librerías en este TFG, ya se justificó la utilización principalmente de las librerías ThermoSysPro y Buildings, aunque componentes de las otras librerías han sido también testeados, pero por los motivos anteriores, no se seleccionarán para el montaje de la planta. Aunque en esta sección si se presentan algunos.

Como ya se ha dicho, se expondrán solo los parámetros generales de entrada que el usuario puede introducir manualmente. Generalmente, los modelos tienen para definir otros tipos de parámetros más especiales que aparecen en otras pestañas (Advanced,

Assumptions, etc) y que permite modificar las ecuaciones de balance de masa, energía o momentum. La existencia o no de flujo inverso, cambiar valores de inicialización, por nombrar algunas características. La descripción de los parámetros se dejará en inglés porque aparecen nombres que son declarados en el código (text view) de esa forma, y su traducción podría perder al lector. De todas formas, aunque se presenten todos los parámetros, no se explicará cada uno de ellos, ni las implicaciones causales que tienen sobre otras variables, ya que solo he trabajado con algunos de ellos, debido a mis necesidades como modelador. Todos son importantes, pero los indicadores que son más interesantes para entender y controlar lo que pasa en esta instalación, son las temperaturas, las potencias y los rendimientos.

Por último, a modo informativo, se indicará si los modelos presentados funcionan o no sin ningún tipo de problemas para la versión de OpenModelica utilizada.

4.2.1. Motor a gas con refrigeración por agua

Path: ThermoSysPro.MultiFluids.Machines.AlternatingEngine

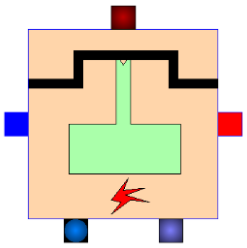


Figura 4.1: Icon View del modelo motor a gas de ThermoSysPro

Este modelo representa un motor de combustión interna con salida eléctrica. El modelo está caracterizado por las siguientes expresiones:

Potencia térmica disponible del combustible:

$$PCS_{comb} = PCI_{comb} + 224.3e5 * XH_{comb} + 25.1e5 * XEAU_{comb}$$

$$W_{comb} = Q_{comb} * PCI_{comb}$$

Dónde: PCS_{comb} es el poder calorífico superior del combustible, PCI_{comb} es el poder calorífico inferior, XH_{comb} es la fracción de hidrógeno en el combustible, $XEAU_{comb}$ es la fracción de H_2O, W_{comb} es la potencia disponible del combustible y Q_{comb} es el caudal de combustible.

Pérdidas térmicas:

$$W_{pth\_ref} = Q_{comb} * PCI_{comb} * (X_{pth} + X_{ref})$$

$Q_e * (H_{sv} - H_{ev}) = Q_{comb} * PCI_{comb} * X_{ref}$

Dónde: W_{pth_ref} potencia referida a perdidas térmicas más refrigeración del motor, X_{pth} es la fracción de perdidas térmicas/refrigeración (0-1 sobre Q.PCI), X_{ref} es la fracción de potencia de refrigeración (0-1 sobre Q.PCI), Q_e es el caudal de agua, H_{sv} y H_{ev} es la entalpía específica del agua de salida y de entrada respectivamente.

Potencia mecánica y rendimiento:

Si mechanical_efficiency_type es igual a 1 (se selecciona en parámetros del modelo), $R_{meca} = R_{meca\_nom}$

Si mechanical_efficiency_type es igual a 2, se calcula R_{meca} a partir de los coeficientes existentes en la pestaña parámetros. La expresión aparece en el text view del modelo.

$W_{meca} = R_{meca} * W_{comb}$

Dónde: R_{meca} es el rendimiento mecánico, R_{meca_nom} es el rendimiento mecánico nominal y W_{meca} es la potencia mecánica del motor.

Potencia eléctrica producida:

Si $W_{meca} > P_{nom} * 0.5$ entonces $W_{elec} = (W_{meca} + R_{elec}) * (0.0479 * Cosphi + 0.952)$, sino $W_{elec} = (W_{meca} + R_{elec\_red}) * (0.0479 * Cosphi + 0.952)$

Dónde: P_{nom} es la potencia nominal del motor, W_{elec} es la potencia eléctrica del motor, R_{elec} es el rendimiento eléctrico, R_{elec_red} es el rendimiento eléctrico a media carga y Cosphi es el cos (phi) de la red eléctrica

FUNCIONA: SÍ

Tabla 4.1: Parámetros modelo motor a gas de ThermoSysPro

Parámetros:

Nombre	Valor por defecto	Descripción
Mechanical_efficiency_type	1	1: fixed nominal efficiency - 2: Linear efficiency using Coef_Rm_a, Coef_Rm_b and Coef_Rm_c - 3: Beau de Rochas cycle efficiency
Rmeca_nom	0.40	Fixed nominal mechanical efficiency (active if mechanical_efficiency_type=1)
Coef_Rm_a	-5.727E-09	Coefficient a for the linear mechanical efficiency (active if mechanical_efficiency_type=2)

Coef_Rm_b	4.5267E-05	Coefficient b for the linear mechanical efficiency (active if mechanical_efficiency_type=2)
Coef_Rm_c	0.312412946	Coefficient c for the linear mechanical efficiency (active if mechanical_efficiency_type=2)
Relec	0.97	Engine electrical efficiency
Relec_red	0.967	Engein electrical efficiency at half load
Pnom	5e8	Engein nominal power
Cosphi	1	Cos(phi) of the eletrical grid
Xpth	0.03	Thermal loss fraction - cooling (0-1 sur Q.PCI)
Xref	0.2	Cooling power fraction (0-1 sur Q.PCI)
MMg	30	Gas average molar mass (g/mol)
DPe	0	Water pressure loss as percent of the pressure at the inlet
DPaf	0	Pressure difference between the air pressure at the inlet and the flue gases pressure at the outlet
RV	6	Engine volume ratio (> 1)
Kc	1.2	Compression polytropic coefficient
Kd	1.4	Expansion polytropic coefficient

4.2.2. Bomba de agua (circulación)

Path: Modelica.Fluid.Machines.ControlledPump

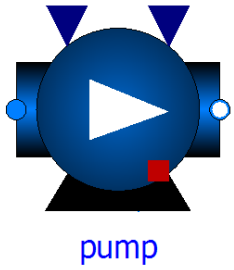


Figura 4.2: Icon View del modelo bomba de agua de MSL

Este modelo describe una bomba centrífuga (o un grupo de nParallel bombas) con velocidad fijada, ya sea asignando un valor fijo en la pestaña parameters o por una señal externa. El modelo se extiende de PartialPump. Si se selecciona N_in, se obtiene la velocidad de rotación de la señal externa conectada a ese conector. De lo contrario, la velocidad de rotación será constante y de valor N_const (que por supuesto puede ser diferente de N_nominal).

FUNCIONA: SÍ

Los espacios en blanco significan que no hay valores por defecto. Hay que introducirlos porque si no da error en la simulación.

Tabla 4.2: Parámetros modelo bomba de agua de MSL

Parámetros:

Nombre	Valor por defecto	Descripción
p_a_nominal		Nominal inlet pressure for predefined pump characteristics
p_b_nominal		Nominal outlet pressure, fixed if not control_m_flow and not use_p_set
m_flow_nominal		Nominal mass flow rate, fixed if control_m_flow and not use_m_flow_set
control_m_flow	true	= false to control outlet pressure port_b.p instead of m_flow

use_m_flow_set	false	= true to use input signal m_flow_set instead of m_flow_nominal
use_p_set	false	= true to use input signal p_set instead of p_b_nominal

Características:

Nombre	Valor por defecto	Descripción
nParallel	1	Number of pumps in parallel
N_nominal	1500	Nominal rotational speed for flow characteristic
rho_nominal	Medium.density_Ptx(Medium.p_default, Medium.T_default, Medium.X_default)	Nominal fluid density for characteristic
use_powerCharacteristic	false	Use powerCharacteristic (vs. efficiencyCharacteristic)

Path: ThermoSysPro.WaterSteam.Machines.StaticCentrifugalPump

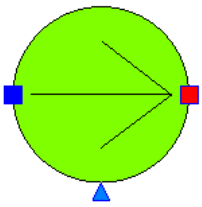


Figura 4.3: Icon View del modelo bomba de agua de ThermoSysPro

Este modelo representa una bomba centrífuga. Las expresiones que describen su comportamiento son las siguientes:

Potencia mecánica:

$W_m = Q * deltaH/rm$

Dónde: W_m es la potencia mecánica de la bomba, Q es el caudal másico, deltaH es la variación de entalpía específica entre la salida y la entrada y rm es el producto del rendimiento mecánico y eléctrico de la bomba.

Potencia hidráulica:

$W_h = Qv * deltaP/rh$

Dónde: W_h es la potencia hidráulica, Qv es el caudal volumétrico, deltaP es la variación de presión entre la salida y la entrada y rh es el rendimiento hidráulico.

Fijación de la velocidad de rotación o potencia mecánica:

Si fixed_rot_or_power es igual a 1 (se selecciona en parámetros del modelo), entonces V_r = rpm_or_mpower.signal

Si fixed_rot_or_power es igual a 2, entonces W_m = rpm_or_mpower.signal

Si rpm_or_mpower es 0 entonces si:

Si fixed_rot_or_power es igual a 1, entonces rpm_or_mpower.signal = VR_{ot}

Si fixed_rot_or_power es igual a 2, entonces rpm_or_mpower.signal = MP_{ower}

Siendo: rpm_or_mpower el conector de entrada para la señal de "control", V_r la velocidad de rotación, VR_{ot} la velocidad de rotación fijada, MP_{ower} la potencia mecánica fijada.

Velocidad de rotación reducida:

$R = V_r / VR_{otn}$

Donde VR_{otn} es la velocidad de rotación nominal.

FUNCIONA: SÍ

Tabla 4.3: Parámetros del modelo bomba de agua de ThermoSysPro

Parámetros:

Nombre	Valor por defecto	Descripción
VRot	1400	Fixed rotational speed (active if fixed_rot_or_power=1 and rpm_or_mpower connector not connected)

MPower	0.1e6	Fixed mechanical power (active if fixed_rot_or_power=2 and rpm_or_mpower connector not connected)
VRotn	1400	Nominal rotational speed
rm	0.85	Product of the pump mechanical and electrical efficiencies
fixed_rot_or_power	1	1: fixed rotational speed - 2: fixed mechanical power
adiabatic_compression	false	true: compression at constant enthalpy - false: compression with varying enthalpy
continuous_flow_reversal	false	true: continuous flow reversal - false: discontinuous flow reversal
fluid	1	1: water/steam - 2: C3H3F5
p_rho	0	If > 0, fixed fluid density
mode	0	IF97 region. 1:liquid - 2:steam - 4:saturation line - 0:automatic
a1	-88.67	x^2 coef. of the pump characteristics hn = f(vol_flow) (s2/m5)
a2	0	x coef. of the pump characteristics hn = f(vol_flow) (s/m2)
a3	43.15	Constant coef. of the pump characteristics hn = f(vol_flow) (m)
b1	-3.7751	x^2 coef. of the pump efficiency characteristics rh = f(vol_flow) (s2/m6)

b2	3.61	x coef. of the pump efficiency characteristics rh = f(vol_flow) (s/m3)
b3	-0.0075464	Constant coef. of the pump efficiency characteristics rh = f(vol_flow) (s.u.)

4.2.3. Intercambiador de calor

Path: Buildings.Fluid.HeatExchangers.ConstantEffectiveness

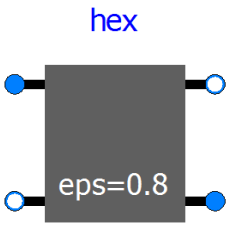


Figura 4.4: Icon View del modelo intercambiador de calor agua/agua de Buildings

Modelo para un intercambiador de calor con efectividad constante. Este modelo transfiere calor siguiendo la siguiente expresión:

$Q = Q_{max} * \varepsilon$,

Dónde ε es una constante de efectividad y Q_{max} el calor máximo que se puede transferir. Para un intercambiador de calor y humedad es aconsejable usar el modelo Buildings.Fluid.MassExchangers.ConstantEffectiveness en lugar de este modelo.

FUNCIONA: SÍ

Tabla 4.4: Parámetros del modelo intercambiador de calor agua/agua de Buildings

Parámetros:

Nombre	Valor por defecto	Descripción
Eps	0.8	Heat exchanger effectiveness

Condición nominal:

Nombre	Valor por defecto	Descripción
m1_flow_nominal		Nominal mass flow rate
m2_flow_nominal		Nominal mass flow rate
dp1_nominal		Pressure difference
dp2_nominal		Pressure difference

Path: ThermoSysPro.WaterSteam.HeatExchangers.StaticWaterWaterExchanger

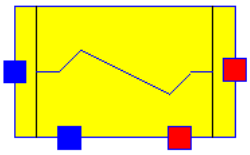


Figura 4.5: Icon View del modelo intercambiador de calor agua/agua de ThermoSysPro

Este modelo representa un intercambiador de calor agua/agua. Este modelo presenta muchas ecuaciones, es recomendable ver su código al completo en el text view. De todas formas, se pondrá algunas que permiten describir el funcionamiento térmico del intercambiador:

Intercambio de calor entre los fluidos (caliente y frío):

$K = h_c * h_f / (h_c + h_f + h_c * h_f * emetal/lambdam)$

$W = K * S * DT_m$

Si ($|Q_c|>1.e-3$) entonces $W = Q_c * proc.cp * (T_{ec} - T_{sc})$ sino $T_{ec} = T_{sc}$

Si ($|Q_f|>1.e-3$) entonces $W = Q_f * prof.cp * (T_{ec} - T_{sc})$ sino $T_{sf} = T_{ef}$

Dónde: K es el coeficiente global de transferencia de calor, h_c es el coeficiente de transferencia de calor del fluido caliente, h_f es el coeficiente de transferencia de calor del fluido frío, emetal es el espesor del metal, lambdam es la conductividad térmica del metal, W es la potencia intercambiada entre los dos conductos, S es la superficie de intercambio

de calor, DT_m es la diferencia media de temperatura (mirar en el código, en OpenModelica, como se calcula), Q_c es el caudal del fluido caliente, Q_f es el caudal del fluido frío, T_{ec} y T_{sc} son las temperaturas de entrada y salida del fluido caliente, T_{ef} y T_{sf} son las temperaturas de entrada y salida del fluido frío. Proc.cp y prof.cp se refieren al calor específico a presión constante. La forma de hallar este valor es a través de varias funciones. Aunque no se explicará aquí, en el apartado de adaptadores de este trabajo se explica estas funciones.

Superficie de intercambio (para un intercambiador de calor a placas):

S = (nbp - 2) * Sp

Dónde: nbp es el número de placas y Sp es la superficie de la placa.

FUNCIONA: SÍ

Tabla 4.5: Parámetros del modelo intercambiador de calor agua/agua de ThermoSysPro

Parámetros:

Nombre	Valor por defecto	Descripción
lambdam	15	Metal thermal conductivity
p_hc	6000	Heat transfer coefficient for the hot side if not computed by the correlations
p_hf	3000	Heat transfer coefficient for the cold side if not computed by the correlations
p_Kc	100	Pressure loss coefficient for the hot side if not computed by the correlations
p_Kf	100	Pressure loss coefficient for the cold side if not computed by the correlations
emetal	0.0006	Wall thickness
Sp	2	Plate area
nbp	499	Number of plates
c1	1.12647	Correction coefficient

p_rhoc	0	If > 0, fixed fluid density for the hot fluid
p_rhof	0	If > 0, fixed fluid density for the cold fluid
modec	0	IF97 region for the hot fluid. 1:liquid - 2:steam - 4:saturation line - 0:automatic
modef	0	IF97 region for the cold fluid. 1:liquid - 2:steam - 4:saturation line - 0:automatic
exchanger_type	1	Exchanger type - 1: countercurrent. 2: cocurrent
heat_exchange_correlation	1	Correlation for the computation of the heat exchange coefficient - 0: no correlation. 1: SRI correlations
pressure_loss_correlation	1	Correlation for the computation of the pressure loss coefficient - 0: no correlation. 1: SRI correlations

Path: ThermoSysPro.MultiFluids.HeatExchangers.StaticExchangerWaterSteam FlueGases

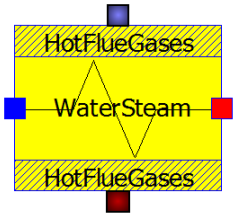


Figura 4.6: Icon View del modelo intercambiador de calor agua/gases de combustión de ThermoSysPro

Este modelo es imprescindible, ya que representa un intercambiador agua/vapor – gases de combustión. Este componente se conectará con el motogenerador para que los gases que salen del motor entren en este intercambiador y transfiera la energía térmica a un

circuito de agua, que representa el circuito de alta temperatura de la caldera de recuperación. Algunas ecuaciones que describen dicha transferencia son:

Potencia intercambiada entre los conductos (caliente y frío):

Si exchanger_type es igual a 1 (se selecciona en la etiqueta parameters) entonces

$W = noEvent (min(Q_e * Cpe, Q_f * Cpf)) * EffEch * (T_{ef} – T_{ee})$

$W = Q_f * (H_{ef} - H_{sf})$

$W = Q_e * (H_{se} – H_{ee})$

Si exchanger_type es igual a 2 entonces

$W = W0$

$W = Q_f * (H_{ef} - H_{sf})$

$W = Q_e * (H_{se} – H_{ee})$

Siendo: W la potencia intercambiada, Q_e caudal de agua, Q_f caudal de los gases de combustión, Cpe el calor específico del agua, Cpf calor específico de los gases de combustión, EffEch rendimiento intercambio térmico, T_{ef} temperatura de los gases de combustión en la entrada, T_{ee} temperatura del agua en la entrada, H_{ef} y H_{sf} entalpía específica de los gases de combustión a la entrada y a la salida, H_{ee} y H_{se} entalpía específica del agua a la entrada y a la salida y W0 potencia intercambiada fijada.

En las ecuaciones se observan dos operadores de la suite OpenModelica: noEvent() y min(). El primer operador tiene que ver con la activación de eventos de estado y tiempo, y el segundo operador elige entre el menor elemento de los dos.

FUNCIONA: SÍ

Tabla 4.6: Parámetros del modelo intercambiador de calor agua/gases de combustión de ThermoSysPro

Parámetros:

Nombre	Valor por defecto	Descripción
exchanger_type	1	Exchanger type - 1: Delta temperature is fixed - 2: delta power is fixed - 3: heat transfer is fixed
EffEch	0.9	Thermal exchange efficiency
W0	0	Power exchanged (active if exchanger_type=2)

K	100	Global heat transfer coefficient (active if exchanger_type=3)
S	10	Global heat exchange surface (active if exchanger_type=3)
Kdpf	10	Pressure loss coefficient on the flue gas side
Kdpe	10	Pressure loss coefficient on the water/steam side
exchanger_conf	1	Exchanger configuration - 1: counter-current. 2: co-current
mode	0	IF97 region of the water. 1:liquid - 2:steam - 4:saturation line - 0:automatic

4.2.4. Máquina de absorción

Path: ThermoSysPro.HeatNetworksCooling.AbsorptionRefrigeratorSystem

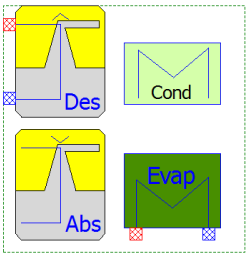


Figura 4.7: Icon View del modelo máquina de absorción de ThermoSysPro

Este modelo representa un sistema de refrigeración por absorción. Este modelo es altamente complejo, ya que está formado a su vez, por modelos presentes en la librería ThermoSysPro. El modelo consta de un absorbedor, de un condensador, de un evaporador, de un generador (desorber), de un intercambiador de calor para disoluciones, elementos de perdida de presión y algún que otro modelo más. Dada la extensión que supondría describirlo todo, solo se mostrará los parámetros que el usuario puede rellenar. Solo apuntar que la mayoría de parámetros hacen referencia a los rendimientos de los intercambiadores. Se puede imponer el caudal de la bomba de la solución.

FUNCIONA: NO

No consigo hacer funcionar este modelo, el error que se presenta en la simulación es el siguiente:

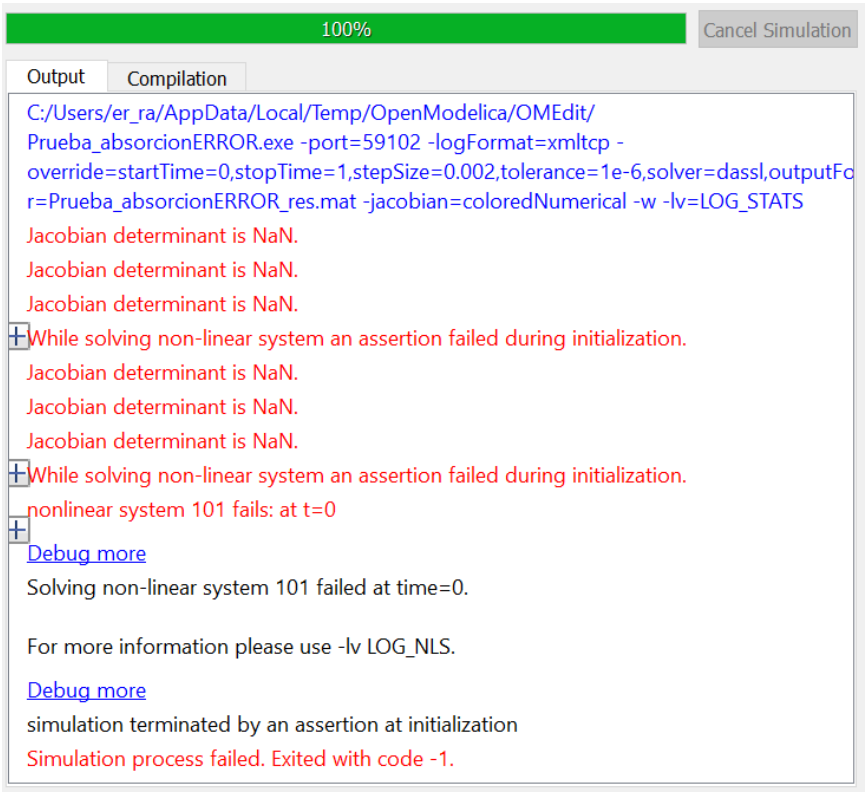


Figura 4.8: Error al simular el modelo máquina de absorción de ThermoSysPro

Tabla 4.7: Parámetros del modelo máquina de absorción de ThermoSysPro

Parámetros:

Nombre	Valor por defecto	Descripción
DesEff	0.362979	Desorber efficiency
Pth	0.33	Desorber thermal losses (0-1 %W)
ExchEff	0.99	Exchanger water LiBr efficiency
EvapEff	0.99	Evaporator efficiency
Qsol	8.856	Solution mass flow rate

Qnom	8.856	Pump solution nominal mass flow rate
DPnom	0.0338605	Pump solution nominal delta pressure

4.2.5. Torre de refrigeración

Path: Buildings.Fluid.HeatExchangers.CoolingTowers.FixedApproach

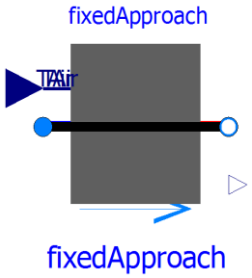


Figura 4.9: Icon View del modelo torre de refrigeración de Buildings

Este modelo describe una torre de refrigeración con temperatura aproximada constante para un estudio en régimen permanente o dinámico. A la temperatura aproximada se le considera la diferencia entre la temperatura de salida del agua, y la de entrada del aire. La temperatura del aire de entrada es añadida desde la señal TAir. Si está conectado a una temperatura seca, se modela una torre de refrigeración "seca". Si está conectado a una temperatura de bulbo húmedo, se modela una torre de refrigeración húmeda. Mediante la conexión de una señal que contiene o bien, una temperatura seca, o bien, una temperatura de bulbo húmedo, este modelo se puede utilizar para estimar la temperatura de retorno de agua de una torre de refrigeración. Para un modelo más detallado, se puede acudir al modelo YorkCalc.

FUNCIONA: SÍ

Tabla 4.8: Parámetros del modelo torre de refrigeración de Buildings

Parámetros:

Nombre	Valor por defecto	Descripción
TApp	2	Approach temperature difference

Condición nominal:

Nombre	Valor por defecto	Descripción
m_flow_nominal		Nominal mass flow rate
dp_nominal		Pressure difference

4.2.6. Bombas de calor (agua-agua y aire-agua)

Path: Buildings.Fluid.HeatPumps.Carnot_TCon

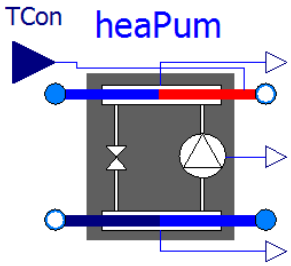


Figura 4.10: Icon View del modelo bomba de calor (calefacción) de Buildings

Este modelo representa una bomba de calor con salida de temperatura del condensador fijada y curva de comportamiento ajustada basada en el rendimiento de Carnot. El coeficiente de operación, COP, cambia con las temperaturas en la misma forma que cambia el rendimiento de Carnot. La entrada de control es la temperatura de consigna de la salida del condensador, que se alcanza exactamente en estado estacionario si la bomba de calor tiene la suficiente capacidad.

Este modelo permite especificar el rendimiento de Carnot $\eta Carnot_{,0}$ o el COP_0 en condiciones nominales, junto a la temperatura del evaporador $T_{eva,0}$ y la del condensador $T_{con,0}$, en cual caso el modelo calcula el rendimiento de Carnot como:

$$\eta_{Carnot,0} = COP_0 / (T_{con,0} / (T_{con,0}-T_{eva,0})).$$

El COP de la bomba de calor se calcula como:

$$COP = \eta_{Carnot,0}\ COP_{Carnot}\ \eta_{PL},$$

Dónde COP_{Carnot} es el rendimiento de Carnot y η_{PL} es un polinomio que expresa el ratio calentamiento- carga parcial y_{PL} que puede ser usado para tener en cuenta un cambio en el COP en condiciones de carga parcial. El polinomio tiene la forma:

$\eta_{PL} = a_1 + a_2\, y_{PL} + a_3\, y_{PL}^2 + \ldots$

Dónde los coeficientes a_i son declarados en el parámetro a. En la etiqueta Dynamics, el modelo puede ser parametrizado para calcularse una respuesta en estado estacionario o en estado transitorio. La respuesta transitoria del modelo se calcula utilizando una ecuación diferencial de primer orden para los volúmenes de fluido del evaporador y condensador. Las temperaturas de salida de la bomba de calor son iguales a la temperatura de estos volúmenes concentrados.

Cuando se use este componente, hay que asegurarse que el condensador tiene suficiente caudal. En función del caudal del evaporador, la diferencia de temperatura y el rendimiento, el modelo calcula el calor transferido en el evaporador. Si el caudal es demasiado pequeño, se pueden dar temperaturas de salida muy bajas, posiblemente por debajo de cero.

El calor del transferido en el condensador QCon_flow_nominal se utiliza para asignar el valor por defecto del caudal, que se utiliza para los cálculos de caídas de presión. También es usado para el rendimiento a carga parcial. Por lo tanto, hay que asegurarse que QCon_flow_nominal se establece en un valor razonable. La capacidad máxima de calentamiento se establece con QCon_flow_max, que por defecto es "infinito", valor = Modelica.Constants.inf.

Por defecto, el coeficiente de operación depende de la temperatura de entrada al evaporador y la de salida del condensador. Esto puede ser cambiado con los parámetros effInpEva y effInpCon.

FUNCIONA: NO

Este modelo presenta el siguiente error (de este error es consciente la comunidad que se encarga de desarrollar, en este caso, la librería Buildings):

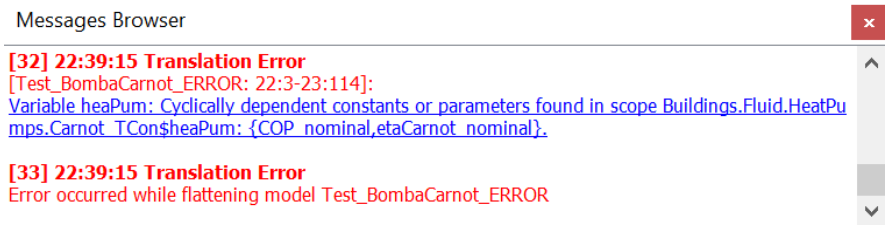


Figura 4.11: Error al simular el modelo bomba de calor (calefacción) de Buildings

Tabla 4.9: Parámetros del modelo bomba de calor (calefacción) de Buildings

Parámetros:

Nombre	Valor por defecto	Descripción

QCon_flow_max	Modelica.Constants.inf	Maximum heat flow rate for heating (positive)
effInpEva	Buildings.Fluid.Types.EfficiencyInput.port_b	Temperatures of evaporator fluid used to compute Carnot efficiency
effInpCon	Buildings.Fluid.Types.EfficiencyInput.port_b	Temperatures of condenser fluid used to compute Carnot efficiency
etaCarnot_nominal	COP_nominal/(TUse_nominal/(TCon_nominal – TEva_nominal))	Carnot effectiveness (=COP/COP_Carnot) used if use_eta_Carnot_nominal = true
COP_nominal	etaCarnot_nominal*(TUse_nominal/(TCon_nominal – TEva_nominal))	Coefficient of performance at TEva_nominal and TCon_nominal, used if use_eta_Carnot_nominal = false
TCon_nominal	30	Condenser temperature used to compute COP_nominal if use_eta_Carnot_nominal=false
TEva_nominal	5	Evaporator temperature used to compute COP_nominal if use_eta_Carnot_nominal=false

Condición nominal:

Nombre	Valor por defecto	Descripción
m1_flow_nominal	QCon_flow_nominal/cp1_default/dTCon_nominal	Nominal mass flow rate
m2_flow_nominal	QEva_flow_nominal/cp2_default/dTEva_nominal	Nominal mass flow rate
QCon_flow_nominal		Nominal heating flow rate
dTEva_nominal	-10	Temperature difference evaporator outlet-inlet

dTCon_nominal	10	Temperature difference condenser outlet-inlet
dp1_nominal		Pressure difference over condenser
dp2_nominal		Pressure difference over evaporator

Eficiencia:

Nombre	Valor por defecto	Descripción
use_eta_Carnot_nominal	QCon_flow_nominal/cp1_default/dTCon_nominal	Set to true to use Carnot effectiveness etaCarnot_nominal rather than COP_nominal
a	QEva_flow_nominal/cp2_default/dTEva_nominal	Coefficients for efficiency curve (need p(a=a, yPL=1)=1)

Path: Buildings.Fluid.Chillers.Carnot_TEva

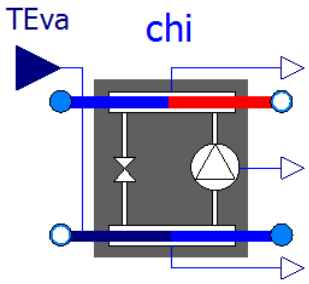


Figura 4.12: Icon View del modelo bomba de calor (refrigeración) de Buildings

Este modelo es exactamente igual que el modelo anterior pero está pensado para funcionar como refrigerador. Se fija la temperatura de salida del evaporador.

FUNCIONA: NO

Se presenta el mismo error de traducción que en el modelo anterior.

Parámetros:

No se pondrá las tablas con los parámetros de entrada ya que son exactamente iguales que en el modelo anterior. Solo cambia QCon_flow_max por QEva_flow_min y QCon_flow_nominal por QEva_flow_nominal.

Path: BuildSysPro.Systems.HVAC.Production.HeatPump.FixedSpeed.

HPHeatingAir2Water

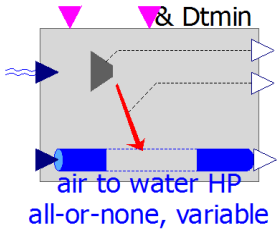


Figura 4.13: Icon View del modelo bomba de calor aire/agua de BuildSysPro

Este modelo representa una bomba de calor aire-agua con compresor a velocidad fija. Se trata de una bomba de calor aire/agua con funcionamiento on/off (control intermitente), solo para modo calentamiento. El modelo es minimalista en términos de parametrización. Está basado en una aproximación empírica para determinar la potencia en régimen permanente de acuerdo con las temperaturas de operación del interior y del exterior. El régimen transitorio es modelado usando un constante de tiempo para la potencia suministrada. La regulación de inicio está determinado por la entrada booleana "u": Verdadero, la bomba debe estar en funcionamiento, Falso, la bomba se detiene.

El control es todo-nada, por lo tanto las fases de comienzo y parada son dadas por el sistema de control. Para proteger la máquina, es común que el mínimo de encendido y los tiempos de apagado sean definidos (DtminOn y DtminOff). Estos parámetros intervienen en el control interno de la máquina.

Dos opciones de parámetros son posibles:

1. Indicación para la potencia nominal y COP
2. Indicación de tres puntos de operación, el modelo empírico requiere de tres puntos de operación para delimitar el rango de las temperaturas del modelo.

FUNCIONA: SÍ

Aunque este modelo funcione perfectamente, sus conectores se encuentran definidos con las interfaces propias de las señales de control (ej: RealImput). Esto dificulta muchísimo la comunicación con componentes de otras librerías. Debido a este problema (que quizá se pueda solucionar con la construcción de un adaptador), y que solo funciona

en modo calentamiento, no se ha considerado un modelo viable para las necesidades de la instalación que se quiere modelar.

Tabla 4.10: Parámetros del modelo bomba de calor aire/agua de BuildSysPro

Parámetros:

Nombre	Valor por defecto	Descripción
Choix	1	

Parametrización elección 1:

Nombre	Valor por defecto	Descripción
Qnom	6490	Nominal heating power in Enom temperature conditions
COPnom	4.3	Nominal coefficient of performance in Enom temperature conditions

Parametrización elección 2:

Nombre	Valor por defecto	Descripción
Enom	{2, 45, 1800, 4950}	Nominal manufacturer data: {Text(°C), ToutputHP(°C), Electric power consumed (W), Supplied heat (W)}
E1	{-15, 55, 1980, 2480}	Manufacturer data: {Text(°C), ToutputHP(°C), Electric power consumed (W), Supplied heat (W)}
E2	{20, 25, 1690, 7750}	Manufacturer data: {Text(°C), ToutputHP(°C), Electric power consumed (W), Supplied heat (W)}

Otros parámetros:

Nombre	Valor por defecto	Descripción
MegRat	0.1	Nominal water flow inside
QfanextRat	0	Outdoor fan power, if QfanextRat included in Qa then choose 0
Cdegi	0.9	Degradation coefficient due to icing: 10% of the defrosting time below 2°C
TauOn	120	Switch-on time constant [GAR 2002]
alpha	0.01	Percentage of standby power (Eco Design Draft report of Task 4: 1, 2 or 3% according to Henderson2000 work)
dtminOn	360	Minimum operating time
dtminOff	360	Minimum stop time before restarting

4.2.7. Disipador

Path: Buildings.Fluid.HeatExchangers.HeaterCooler_T

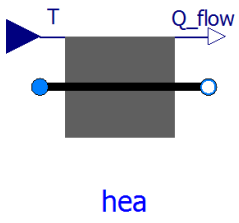


Figura 4.14: Icon View del modelo calentador-refrigerador ideal de Buildings

Modelo de un calentador o refrigerador ideal en el que podemos fijar la temperatura de salida. Nos servirá como disipador.

Este modelo fuerza la temperatura de salida en port_b, para que sea igual a la temperatura de entrada Tset., sujeto a los límites opcionales en la capacidad de calefacción o refrigeración Q_flow_max y Q_flow_min.

Para capacidad de calentamiento o enfriamiento ilimitada, se establece Q_flow_maxHeat = Modelica.Constant.inf y Q_flow_maxCool = -Modelica.Constant.inf.

La señal de salida Q_flow es el calor aportado (para calentamiento) o retirado (para refrigeración) del fluido si el sentido del flujo es del port_a al port_b. Si el flujo es inverso, entonces Q_flow = 0. La temperatura de salida en el port_a no sería afectada por el modelo.

Si el parámetro energyDynamics no es igual a Modelica.Fluid.Types.Dynamics.SteadyState, el modelo responde dinámicamente usando una ecuación diferencial de primer orden. La constante de tiempo en el componente es igual al parámetro tau. Esta constante de tiempo es ajustada a través del caudal usando la ecuación:

$$\tau_{eff} = \tau \ |\dot{m}| / \dot{m}_{nom}$$

Dónde τ_{eff} es la constante de tiempo efectiva dada para el caudal $\dot{m}$ y τ es la constante de tiempo para el caudal nominal $\dot{m}_{nom}$. Este tipo de dinámica es igual a la dinámica que con un volumen de control tendría.

Opcionalmente, este modelo puede tener una resistencia de flujo. Si no se solicita una resistencia al flujo, se establece dp_nominal = 0.

Para un modelo que usa una señal de control u, comprendida entre 0 y 1, multiplicada por la potencia nominal de calentamiento o refrigeración para obtener el calor cedido o retirado al fluido, usar el modelo Buildings.Fluid.HeatExchangers.HeaterCooler_u.

FUNCIONA: SÍ

Tabla 4.11: Parámetros del modelo calentador-refrigerador ideal de Buildings

Parámetros:

Nombre	Valor por defecto	Descripción
Q_flow_maxHeat	Modelica.Constants.inf	Maximum heat flow rate for heating (positive)
Q_flow_maxCool	-Modelica.Constants.inf	Maximum heat flow rate for cooling (negative)

Condición nominal:

Nombre	Valor por defecto	Descripción
m_flow_nominal		Nominal mass flow rate

dp_nominal		Pressure difference
m_flow_nominal		Nominal mass flow rate, used for regularization near zero flow

4.2.8. Válvula de tres vías

Path: Buildings.Fluid.Actuadors.Valves.ThreeWayLinear

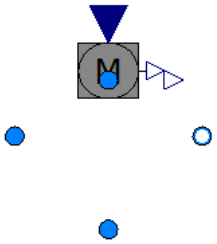


Figura 4.15: Icon View del modelo válvula de tres vías de Buildings

Este modelo representa una válvula de tres vías con características lineales de apertura. Este modelo está basado en los modelos parciales de válvulas:

Buildings.Fluid.Actuators.BaseClasses.PartialThreeWayValve
Buildings.Fluid.Actuators.BaseClasses.PartialTwoWayValve

Aunque este modelo presenta características de apertura lineales, el modelo base está pensado para válvulas con diferentes características de apertura (lineales, de igual porcentaje o apertura rápida). El modelo consiste en un mezclador donde las válvulas se colocan en dos tramos de flujo. El tercer tramo no presenta fricción. El coeficiente de flujo Kv para el flujo de port_1 -> port_2 es un parámetro. El coeficiente de flujo del camino secundario, desde el port_3 -> port_2 se calcula como:

$$fraK = \frac{Kv\ (port\_3 \rightarrow port\_2)}{Kv\ (port\_1 \rightarrow port\_2)}$$

Dónde 0 < fraK < 1 es un parámetro con valor por defecto de 0,7.

Ya que los coeficientes de flujo es un parámetro que el modelador ha de tener en cuenta, es importante saber que es un coeficiente de flujo en válvulas de control. El coeficiente de flujo es un factor diseñado que relaciona la caída de presión con el caudal. Cada válvula tiene su propio coeficiente de flujo. Esto depende de cómo la válvula fue diseñada para dejar pasar el flujo a través de ella. Por lo tanto, las principales diferencias

entre los diferentes coeficientes de flujo vienen dados por el tipo de válvula y la posición de apertura de ésta. Si la válvula va a trabajar gran parte de su tiempo abierta, probablemente se debería seleccionar una válvula con baja perdida de presión para ahorrar energía. O si la válvula se necesita como control, los rangos de los coeficientes para las diferentes posiciones de apertura de la válvula fijarían los requerimientos de la aplicación. El coeficiente de flujo puede ser expresado de muchas formas. Puede ser dimensional o con unidades si los parámetros, tal como el diámetro o densidad, son considerados dentro del coeficiente. El modelo permite elegir que coeficiente de flujo utilizar. En Buildings.Fluid.Actuators.BaseClasses.ValveParameters se especifican los diferentes coeficientes de flujo en condiciones totalmente abiertas. CvData puede fijarse a los coeficientes Av, Kv, Cv, OpPoint. Cada uno dispone de diferentes unidades, y tiene en consideración diferentes variables y condiciones de operación.

FUNCIONA: SÍ

Tabla 4.12: Parámetros del modelo válvula de tres vías de Buildings

Parámetros:

Nombre	Valor por defecto	Descripción
X_start	Medium.X_default	Start value of mass fractions m_i/m
fraK	0.7	Fraction Kv(port_3→port_2)/Kv(port_1→port_2)
l	{0.0001, 0.0001}	Valve leakage, l=Kv(y=0)/Kv(y=1)

Coeficiente de flujo:

Nombre	Valor por defecto	Descripción
CvData	Buildings.Fluid.Types.CvTypes. OpPoint	Selection of flow coefficient
Kv		Kv (metric) flow coefficient [m3/h/(bar)^(1/2)]
Cv		Cv (US) flow coefficient [USG/min/(psi)^(1/2)]

Av		Av (metric) flow coefficient

Linealización Presión-flujo:

Nombre	Valor por defecto	Descripción
deltaM	0.02	Fraction of nominal flow rate where linearization starts, if y=1
deltaM	0.02	Fraction of nominal flow rate where linearization starts, if y=1

Condición nominal:

Nombre	Valor por defecto	Descripción
m_flow_nominal		Nominal mass flow rate
dpValve_nominal		Nominal pressure drop of fully open valve, used if CvData=Buildings.Fluid.Types.CvTypes.OpPoint
dpFixed_nominal	{0, 0}	Nominal pressure drop of pipes and other equipment in flow legs at port_1 and port_3

Path: ThermoSysPro.WaterSteam.PressureLosses.ThreeWayValve

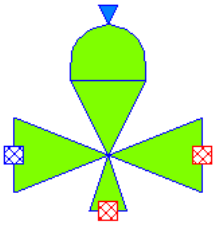


Figura 4.16: Icon View del modelo válvula de tres vías de ThermoSysPro

Este modelo representa una válvula de tres vías en el que las características de apertura pueden ser dadas por el usuario. Igualmente, consiste en un mezclador donde las válvulas (una en cada tramo) se colocan en dos tramos de flujo. El modelo de las válvulas es: ThermoSysPro.WaterSteam.PressureLosses.ControlValve. Las ecuaciones que describen su comportamiento son principalmente las siguientes:

Cv como función de la posición de la válvula:

Si mode_caract es igual a 0 (se selecciona en la etiqueta parameters) entonces

Cv = Ouv.signal * Cvmax

Si mode_caract es igual a 1, entonces

Si option_interpolation es igual a 1,

Cv = ThermoSysPro.Pro.Functions.LinearInterpolation (caract[: , 1], caract[: , 2], Ouv.signal)

Si option_interpolation es igual a 2,

Cv = ThermoSysPro.Pro.FunctionsSplineInterpolation (caract[: , 1], caract[: , 2], Ouv.signal)

Siendo: Cvmax el máximo Cv, modo_caract parámetro para darle a la válvula características lineales o dadas por caract[], Ouv.signal señal de entrada de control de las válvulas. Las funciones de ThermosysPro cuenta con diversas funciones para interpolar, en este caso los dos modos de interpolación son lineal o mediante splines (uso de polinomios de bajo grado que evitan las oscilaciones de los polinomios de alto grado en la interpolación polinomial).

Ambas válvulas están controladas por un sistema de control basado en el bloque: ThermoSysPro.InstrumentationAndControl.Blocks.Math.Add, donde la señal de salida es:

y.signal = k1 * u1.signal +k2 * u2.signal (siendo el valor de las constantes k1 = +1, k2 = -1, y la señal u1 = 1).

FUNCIONA: SÍ

Tabla 4.13: Parámetros del modelo válvula de tres vías de ThermoSysPro

Parámetros:

Nombre	Valor por defecto	Descripción
Cvmax1	8005.42	Valve 1 max CV
Cvmax2	8005.42	Valve 2 max CV

caract1	[0, 0; 1, Cvmax1]	Valve 1 - Position vs. Cv characteristics (active if mode_caract1=true)
caract2	[0, 0; 1, Cvmax2]	Valve 2 - Position vs. Cv characteristics (active if mode_caract2=true)
mode_caract1	0	Valve 1 - 0:linear characteristics - 1:characteristics is given by caract1[]
mode_caract2	0	Valve 2 - 0:linear characteristics - 1:characteristics is given by caract2[]
V	1	Three way valve volume
continuous_flow_reversal	false	true: continuous flow reversal - false: discontinuous flow reversal
fluid	1	1: water/steam - 2: C3H3F5
p_rho	0	If > 0, fixed fluid density
mode	0	IF97 region. 1:liquid - 2:steam - 4:saturation line - 0:automatic

4.2.9. Colectores/ Uniones/ Separaciones

Existen muchos modelos en las distintas librerías de uniones y separaciones de flujo. Algunos de éstos son:

Path: ThermoSysPro.WaterSteam.Junctions.Mixer2(3)

Path: ThermoSysPro.WaterSteam.Junctions.Splitter2(3)

Path: Buildings.Fluid.FixedResistances.SplitterFixedResistanceDpM

Path: ThermoSysPro.WaterSteam.Junctions.Mixer8

Vamos a seleccionar uno de cada librería:

Path: Buildings.Fluid.FixedResistances.SplitterFixedResistanceDpM

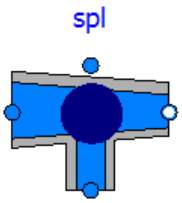


Figura 4.17: Icon View del modelo separador-mezclador de flujos de Buildings

Modelo de un divisor o mezclador de flujo con una resistencia fija en cada puerto. En cada separación de flujo, se puede modelar una caída de presión, y donde se unen los fluidos, se puede modelar un volumen de mezcla. La caída de presión se implementa utilizando el modelo Buildings.Fluid.FixedResistances.FixedResistanceDpM. Si se implementa una caída de presión nominal igual a cero, el modelo de caída de presión se eliminará. Por ejemplo, la declaración siguiente modelaría un mezclador que tiene los caudales nominales y caídas de presión que se muestran en la ilustración de abajo:

```
m_flow_nominal={ 0.1, 0.1,  -0.2},
dp_nominal =   {500,    0, -6000}
```

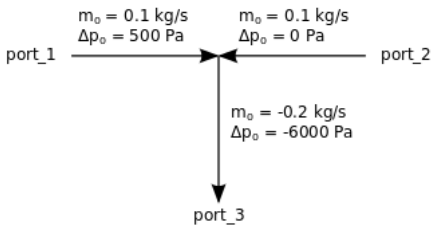


Figura 4.18: Representación de caudales y caídas de presión de los flujos

Hay que tener en cuenta que el port_3 se establece en valores negativos. Los valores negativos indican que en las condiciones nominales, el fluido está saliendo del componente.

Opcionalmente la unión de los fluidos se puede modelar con un volumen de control. Se implementa usando el modelo Buildings.Fluid.Delay.DelayFirstOrder. El volumen es modelado si energyDynamics es distinto de Modelica.Fluid.Types.Dynamics.SteadyState. El volumen de control tiene el tamaño:

V = sum(abs(m_flow_nominal[:])/3)*tau/rho_nominal,

Dónde tau es un parámetro y rho_nominal es la densidad del medio en el volumen en condiciones nominales.

Si configuramos energyDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial, puede ayudar a reducir el tamaño del sistema de ecuaciones no lineales.

FUNCIONA: SÍ

Tabla 4.14: Parámetros del modelo mezclador-separador de flujos de Buildings

Parámetros:

Nombre	Valor por defecto	Descripción
X_start	Medium.X_default	Start value of mass fractions m_i/m

Condición nominal:

Nombre	Valor por defecto	Descripción
m_flow_nominal		Mass flow rate. Set negative at outflowing ports
dp_nominal		Pressure drop at nominal mass flow rate, set to zero or negative number at outflowing ports

Transición a laminar

Nombre	Valor por defecto	Descripción
use_dh	false	= true, use dh and ReC, otherwise use deltaM
deltaM	0.3	Fraction of nominal mass flow rate where transition to turbulent occurs
dh	{1, 1, 1}	Hydraulic diameter
ReC	{4000, 4000, 4000}	Reynolds number where transition to turbulent starts

Path: ThermoSysPro.WaterSteam.Junctions.Mixer8

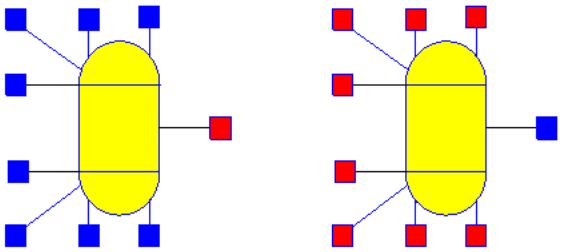


Figura 4.19: Icon View del modelo mezclador de flujos de ThermoSysPro y separador de flujos (modificado del mezclador)

El componente simplemente esta modelado con la implementación de las ecuaciones de conservación de masa y energía. Presenta la peculiaridad que en los puntos de conexión que no tienen conectado otro modelo, el caudal en ese conector es cero.

Este modelo se ha modificado para que también funcione como splitter. Su código se encuentra en el Anexo B: Text View Colectores modificados.

En este Anexo también aparece el código del modelo anterior, Buildings.Fluid.FixedResistances.SplitterFixedResistanceDpM, modificado, para poder admitir más entradas/salidas.

FUNCIONA: SÍ

Tabla 4.15: Parámetros del modelo mezclador-separador de flujos de ThermoSysPro

Parámetros:

Nombre	Valor por defecto	Descripción
fluid	1	1: water/steam - 2: C3H3F5
mode	0	IF97 region. 1:liquid - 2:steam - 4:saturation line - 0:automatic

4.2.10. Líneas de Tuberías

Al igual que pasaba con el apartado anterior, existe multitud de modelos en las distintas librerías. Para no extender demasiado el apartado solo nombraré algunos modelos, y describiré con más detalle los dos modelos que más he utilizado.

Path: Modelica.Fluid.Pipes.StaticPipe

Path: Modelica.Fluid.Pipes.DynamicPipe

Path: Buildings.Fluid.FixedResistances.Pipe

Path: ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe

Path: ThermoSysPro.WaterSteam.HeatExchangers.DynamicOnePhaseFlowPipe

Casi todos los modelos son muy parecidos. La diferencia significativa es que algunas tuberías son con almacenamiento de masa y energía (dynamic) y las otras no. Las que se utilizarán para la planta son sin almacenamiento de masa y energía. Los modelos son los siguientes:

Path: Modelica.Fluid.Pipes.StaticPipe

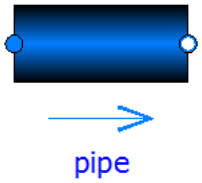


Figura 4.20: Icon View del modelo tubería de MSL

Modelo de una tubería recta con sección transversal constante y balances de masa, momentum y energía en régimen permanente. Existen dos estados termodinámicos, uno en cada puerto de fluido (fluid port). El balance de momentum es definido por dos estados, teniendo en cuenta el flujo de momentum, la fricción y la gravedad. El mismo resultado se puede obtener utilizando el modelo DynamicPipe poniendo steady-state (régimen permanente) en las opciones dynamic. El uso pretendido es proporcionar conexiones simples de recipientes u otros dispositivos con almacenamiento, como se hacen en los siguientes ejemplos que aparecen en OpenModelica:

- Examples.Tanks.EmptyTanks
- Examples.InverseParameterization

FUNCIONA: SÍ

Tabla 4.16: Parámetros del modelo tubería de MSL

Parámetros:

Nombre	Valor por defecto	Descripción
isCircular	true	= true if cross sectional area is circular

Geometría:

Nombre	Valor por defecto	Descripción
nParallel	1	Number of identical parallel pipes
length		Length
diameter		Diameter of circular pipe
crossArea	Modelica.Constants.pi*diameter*diameter/4	Inner cross section area
perimeter	Modelica.Constants.pi*diameter	Inner perimeter
roughness	2.5e-5	Average height of surface asperities (default: smooth steel pipe)

Presión estática:

Nombre	Valor por defecto	Descripción
height_ab	0	Height(port_b) - Height(port_a)

Path: ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe

Figura 4.21: Icon View del modelo tubería de ThermoSysPro

Este modelo representa líneas de tuberías rectas (conducto circular). El modelo está basado en la ecuación de Darcy-Weisback. Las expresiones fundamentales que explican las opciones de configuración principales son:

khi = lam * L/D

Si lambda_fixed es verdadero entonces lam = lambda, sino, se calcula lam en función de una seria de variables, como la rugosidad de la tubería, el número de Reynolds, etc.

Khi es el coeficiente de perdida de presión hidráulica, lambda el coeficiente de fricción dado por parameters, L longitud de la tubería y D, su diámetro.

FUNCIONA: SÍ

Tabla 4.17: Parámetros del modelo tubería de ThermoSysPro

Parámetros:

Nombre	Valor por defecto	Descripción
L	10	Pipe length
D	0.2	Pipe internal hydraulic diameter
lambda	0.03	Friction pressure loss coefficient (active if lambda_fixed=true)
rugosrel	0.0001	Pipe roughness (active if lambda_fixed=false)
z1	0	Inlet altitude
z2	0	Outlet altitude
Lambda_fixed	true	true: lambda given by parameter - false: lambde computed using Idel'Cik correlation
Inertia	false	true: momentum balance equation with inertia - false: without inertia
continuous_flow_reversal	false	true: continuous flow reversal - false: discontinuous flow reversal
fluid	1	1: water/steam - 2: C3H3F5
p_rho	0	If > 0, fixed fluid density

mode	0	IF97 region. 1:liquid - 2:steam - 4:saturation line - 0:automatic

4.3. Modelos Alternativos

Debido a los problemas que hay con los modelos de OpenModelica de los componentes: Maquina de absorción y Bomba de calor, se ha decidido modelar estos equipos en base a que tienen curvas que recogen el comportamiento de alguna propiedad determinada cuando se modifican las condiciones bajo las cuales el valor de esa propiedad fue establecida. De esta forma se puede introducir, por ejemplo, como varía la capacidad total de refrigeración de un equipo autónomo cuando se modifican las temperaturas a las cuales se encuentran sometidos el evaporador y el condensador.

Para la realización de esta tarea se ha acudido al Manual de Curvas CALENER-GT Grandes edificios terciarios.

Nuestros dos equipos corresponden a Plantas enfriadoras en el documento anterior. En particular se ha tomada el tipo de equipo Bomba de calor 2T condensada (CD) por agua y Absorción simple etapa condensada (CD) por agua. Con respecto a las bombas de calor, el Manual da la posibilidad de elegir entre Bombas de calor 2T o 4T:

- ✓ Bombas de calor 2T: estos equipos hacen inversión de ciclo para calentar o enfriar el líquido de trabajo que llega al equipo por una sola tubería de impulsión y otra de retorno.
- ✓ Bombas de calor 4T: estos equipos no necesitan hacer inversión de ciclo para calentar o enfriar el líquido de trabajo que llega al equipo por dos tuberías de impulsión y dos de retorno, una para calor y otra para frío.

Las variables de este tipo de equipo que son afectadas por curva de comportamiento son:

POT *POT = POTNOM . POT(T)*

CoolEIR *CoolEIR = CoolEIRNOM . CoolEIR(T).CoolEIR(PLR)*

HeatEIR *HeatEIR = HeatEIRNOM . HeatEIR(T).HeatEIR(PLR)*

HIR *HIR = HIRNOM . HIR (T) . HIR(PLR)*

POTCAL *POTCAL = POTCALNOM . POTCAL (T)*

El significado de cada una de estas variables no se expondrá aquí, ya que aparece de forma muy clara y explícita en el Manual de Curvas. Solo resaltar que las variables que aparecen posteriormente en el Manual, son para calcular el EER y COP correspondiente, y no los valores de CoolEIR Y HeatEIR. Por lo tanto la expresión sería la siguiente (ejemplo de refrigeración, EER).

EER = EERnom * EER_ELEC_T * EER_ELEC_FCP

Siendo: EER = 1/CoolEIR EERnom = 1/CoolEIRNOM

El código completo de ambos equipos (Máquina de absorción y Bomba de calor) aparece en el Anexo C: Text View Bombas de Calor y Máquina de absorción. En la siguiente tabla se muestra las variables calculadas por curva de comportamiento, se muestra el tipo de curva que es y la ecuación para calcular la curva. La curva presenta una serie de coeficientes. El valor de estos coeficientes se encuentra igualmente en el Manual de Curvas CALENER-GT.

Tabla 4.18: Descripción del equipo Absorción simple etapa de Manual de Curvas CALENER-GT

Refrigeración	**Absorción simple etapa (Subtipo CD agua)**
Variable	POT-NOM_T (POT(T))
Tipo de curva	Bi-Cuadrática-T (BQUT)
Ecuación	$f(T\_sal, T\_ent) = a + b * T\_sal + c * (T\_sal)^2 + d * T\_ent + e * (T\_ent)^2 + f * T\_sal * T\_ent$
Variable	EER-TERM_T (HIR(T))
Tipo de curva	Bi-Cuadrática-T (BQUT)
Ecuación	$f(T\_sal, T\_ent) = a + b * T\_sal + c * (T\_sal)^2 + d * T\_ent + e * (T\_ent)^2 + f * T\_sal * T\_ent$
Variable	EER-TERM_FCP (HIR(PLR))
Tipo de curva	Cuadrática (QU)
Ecuación	$f(PLR) = a + b * PLR + c * PLR^2$

T_sal = temperatura del agua a la salida del evaporador.

T_ent = temperatura del agua a la entrada del condensador.

PLR = factor de carga parcial, definido como cociente entre la carga del sistema y la capacidad sensible del mismo.

Tabla 4.19: Descripción del equipo Bomba de calor 2T (refrigeración) de Manual de Curvas CALENER-GT

Refrigeración	Bomba de calor 2T (Subtipo CD agua)
Variable	POT-NOM_T (POT(T))
Tipo de curva	Bi-Cuadrática-T (BQUT)
Ecuación	f (T_sal, T_ent) = a + b * T_sal + c * (T_sal)^2 + d * T_ent + e * (T_ent)^2 + f * T_sal * T_ent
Variable	EER-ELEC_T (CoolEIR(T))
Tipo de curva	Bi-Cuadrática-T (BQUT)
Ecuación	f (T_sal, T_ent) = a + b * T_sal + c * (T_sal)^2 + d * T_ent + e * (T_ent)^2 + f * T_sal * T_ent
Variable	EER-ELEC_FCP (CoolEIR(PLR))
Tipo de curva	Cúbica (CUB)
Ecuación	f (PLR) = a + b * PLR + c * PLR^2 + d * PLR^3

Tabla 4.20: Descripción del equipo Bomba de calor 2T (calefacción) de Manual de Curvas CALENER-GT

Calefacción	**Bomba de calor 2T (Subtipo CD agua)**
Variable	POT-NOM_CAL_T (POTCAL(T))
Tipo de curva	Bi-Cuadrática-T (BQUT)
Ecuación	f (T_sal, T_ent) = a + b * T_sal + c * (T_sal)^2 + d * T_ent + e * (T_ent)^2 + f * T_sal * T_ent

Variable	COP-ELEC_T (HeatEIR(T))
Tipo de curva	Bi-Cuadrática-T (BQUT)
Ecuación	f (T_sal, T_ent) = a + b * T_sal + c * (T_sal)^2 + d * T_ent + e * (T_ent)^2 + f * T_sal * T_ent
Variable	COP-ELEC_FCP (HeatEIR(PLR))
Tipo de curva	Cúbica (CUB)
Ecuación	f (PLR) = a + b * PLR + c * PLR^2 + d * PLR^3

Una puntualización final es que las temperaturas tienen que estar en ºF. Por lo tanto, para la construcción de los modelos se ha hecho necesario utilizar una función de conversión presente en OpenModelica. Para ilustrar como mayor facilidad el proceso, se presenta el siguiente código (text view de ejemplo):

```
model Test_Conversion
  Modelica.SIunits.Temperature Te = 293.15;
  Modelica.SIunits.Conversions.NonSIunits.Temperature_degF Ts;
equation
  Ts = Modelica.SIunits.Conversions.to_degF(Te);
end Test_Conversion;
```

En primer lugar se hace necesario declarar las variables (con sus respectivas unidades). En el caso que nos ocupa la clase especializada type es:

`Modelica.SIunits.Conversions.NonSIunits.Temperature_degF`

Posteriormente en "equation" se realiza la conversión. Siendo Ts, Te pasado a ºF. La función utilizada es: `Modelica.SIunits.Conversions.to_degF`.

Todo el proceso aparece en el anexo mencionado con anterioridad, donde está el Text View de los modelos.

4.4. Problemas y consideraciones en OpenModelica

Durante el trascurso de este TFG se ha trasteado mucho con la suite OpenModelica. Aunque se esperaba que la mayoría de los modelos funcionasen sin problemas, y que se pudiesen conectar unos con otros (incluso de otras librerías) sin el más mínimo percance, la realidad es que durante el proyecto se han producido muchas incidencias. En esta sección

se describirá algunos de los problemas que se han dado durante el trabajo y se propondrá algunas soluciones que pueden servir de ayuda. También se pasará por encima de algunas consideraciones que ayudarán al entendimiento de la suite y del código, para que los modelos funcionen y no den problemas al simularlos.

4.4.1. Replaceable, Redeclare

Tal como están construidas las librerías MSL y Buildings, cuando se utilizan instancias de algún modelo, y le damos al botón de simulación, ésta no puede empezar por el siguiente error de traducción:

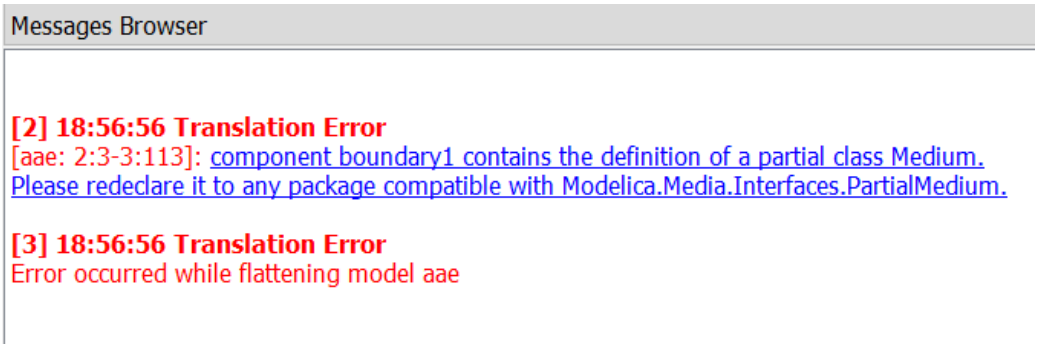


Figura 4.22: Error de traducción por la definición de la clase parcial Medium en componente boundary1

El componente boundary1 corresponde a una fuente (condición de contorno), aunque esto no es relevante. El error de traducción es debido a que el componente boundary1 contiene la definición de una clase parcial Medium. Y te pide redeclarar algún paquete compatible con Modelica.Media.Interfaces.PartialMedium. Esto pasa con la mayoría de modelos de las librerías antes referidas.

En primer lugar expliquemos de donde viene esto de redeclarar paquetes o clases.

En OpenModelica existen 2 casos de parametrización de clases genéricas. Los parámetros de clase pueden ser:

- ✓ Parámetros de instancias (cuyos valores son instancias)
- ✓ Parámetros de tipo (cuyos valores son tipos (Real, Integer, Boolean, String))

En este contexto parámetro de clase no hace referencia a los parámetros de los modelos que se declaran con la palabra clave PARAMETER.

Este nuevo térmico designa a los parámetros formales de clase. Tales parámetros formales llevan la palabra clave **replaceable.**

- o *Parámetros de clase que son instancias:*

Veamos este ejemplo:

```
class C
  replaceable ClaseVerde pobj1(p1=5);
  replaceable ClaseAmarilla pobj2;
  replaceable ClaseVerde pobj3;
  ClaseRojo obj4;
equation
  ...
end C;
```

La clase C tiene tres parámetros de clase señalados por la palabra clave replaceable. El obj4 no es declarado como reemplazable, por lo tanto no es parámetro de clase.

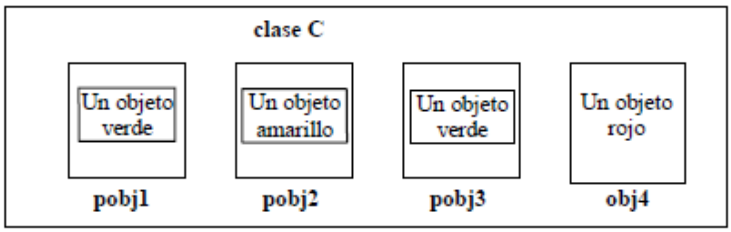


Figura 4.23: Esquema de la clase C

Ahora declaramos otra clase:

```
class C2 = C(redeclare ClaseRoja pobj1, redeclare ClaseVerde pobj2);
```

Lo que se ha conseguido utilizando la palabra redeclare es lo siguiente:

```
class C2
  ClaseRoja pobj1(p1=5);
  ClaseVerde pobj2;
  ClaseVerde pobj3;
  ClaseRojo obj4;
equation
...
end C2;
```

- *Parámetros de clase que son tipos:*

Un parámetro de clase puede ser un tipo. Esto es útil para cambiar la clase de varios objetos simultáneamente.

Veamos un ejemplo:

```
class C
  replaceable class ClaseColoreada = ClaseVerde;
  ClaseColoreada obj1(p1=5);
  replaceable ClaseAmarilla obj2;
  ClaseColoreada obj3;
  ClaseRoja obj4;
equation
  ...
end C;
```

Al definir el parámetro de tipo ClaseColoreada en la clase C, es fácil cambiar el color de todos los objetos del tipo ClaseColoreada:

```
class C2 = C(redeclare class ClaseColoreada = ClaseAzul);
```

Esto equivale a la siguiente definición de C2:

```
class C2
  ClaseAzul obj1(p1=5);
  ClaseAmarilla obj2;
  ClaseAzul obj3;
  ClaseRoja obj4;
equation
  ...
end C2;
```

Una vez dicho esto, los problemas de traducción que se dan en los componentes de las librerías anteriores son debido a que existen clases con parametrización de clases (tipo). Tales parámetros llevan la palabra clave replaceable.

Suelen ser packages que contienen clases sobre las propiedades del medio (fluido de trabajo, Medium). Por lo tanto se hace necesario redeclarar. Por ejemplo:

```
Modelica.Fluid.Sources.MassFlowSource_T   boundary(redeclare
package Medium = Buildings.Media.Water)
```

Dentro de la clase `Modelica.Fluid.Sources.MassFlowSource_T` o otra clase de la cual se ha extendido ésta (ver palabra clave extends) aparecerá un parámetro formal de clase con la palabra clave replaceable.

También podemos hacerlo así:

```
replaceable package Medium = Buildings.Media.Water;

  Modelica.Fluid.Sources.MassFlowSource_T boundary(redeclare
package Medium = Medium)
```

No es necesario poner la palabra replaceable en el segundo caso.

Algunos modelos contienen declarados dos Medium (Medium1, Medium2), como es el caso de los intercambiadores. En este caso la solución podría ser esta:

```
package Medium1 = Buildings.Media.Water;

package Medium2 = Buildings.Media.Water;

  Buildings.Fluid.HeatExchangers.ConstantEffectiveness
hex(redeclare package Medium1 = Medium1, Medium2 = Medium2)
```

4.4.2. Loopbreakers

Existe un problema especial con respecto a los sistemas de ecuaciones resultantes de modelos con estructura en bucle. Si una variable se extiende desde un componente en ambas direcciones (su entrada y su salida), habrá ubicaciones en el sistema con una conexión sobre-constreñida (la misma variable viene de ambos lados). Para superar esto, se necesita hacer uso de loopbreakers (interruptores de bucle). Hay clases pequeñas que tienen que colocarse en el centro de la conexión donde aparece el problema. Se especifica que variables no deben transmitirse a través de la conexión en ese punto entre las variables de los conectores. Esto eliminará las ecuaciones redundantes en la conexión, haciendo que el modelo de circuito cerrado funcione.

En la librería ThermoSysPro existen diferentes loopbreakers para distintas variables. Los modelos para sistemas cuyo medio sea agua o vapor (WaterSteam) son los siguientes:

Path: ThermoSysPro.WaterSteam.LoopBreakers.LoopBreakerH

Path: ThermoSysPro.WaterSteam.LoopBreakers.LoopBreakerQ

Path: ThermoSysPro.WaterSteam.LoopBreakers.LoopBreakerP

Path: ThermoSysPro.WaterSteam.LoopBreakers.LoopingPQ

En el siguiente ejemplo se verá el error que se produce cuando no se introducen loopbreakers.

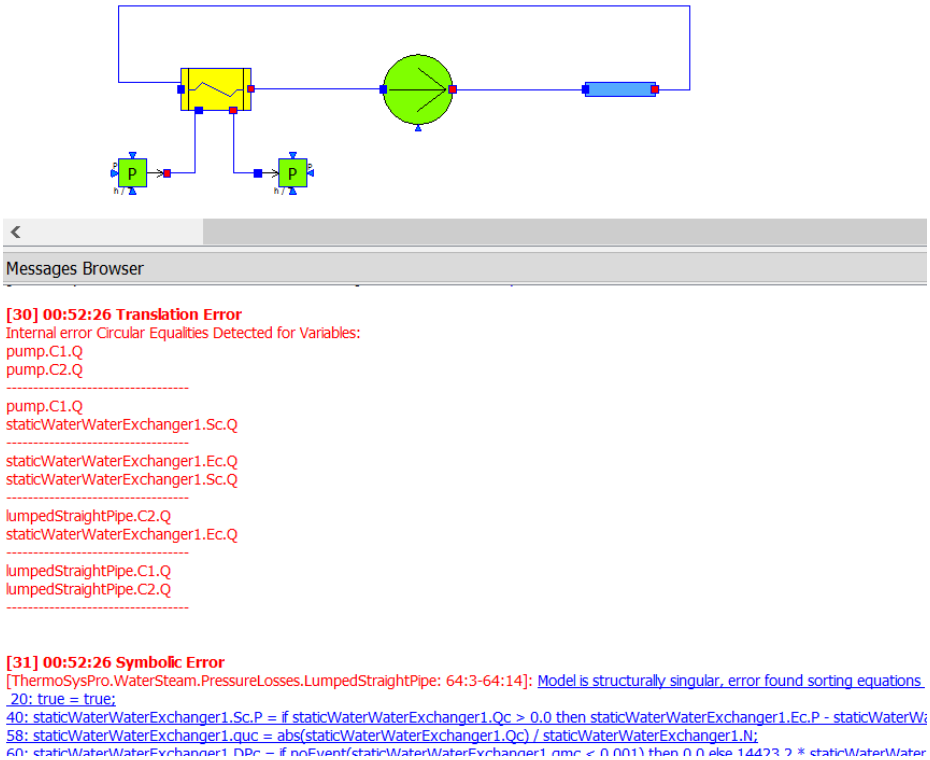


Figura 4.24: Error de traducción por detectar variables con igualdades circulares

Este es un modelo muy simple, compuesto por una bomba de circulación, una tubería y un intercambiador de calor. Para el otro circuito del intercambiador de calor se han utilizado BoundaryConditions (condiciones de contorno), una fuente (source) y un sumidero (sink). Más allá de la simplicidad del circuito, esto ocurre con circuitos más complejos. Una solución poco ortodoxa es la utilización de tanques de agua. Aunque la utilización de tanques modifica el comportamiento dinámico del modelo, y por lo tanto, hay que contar con ello.

Ahora se muestra un ejemplo acompañado de loopbreakers y de otros modelos que fijan el valor de alguna variable potencial o de flujo. Estos modelos también están dentro del paquete BoundaryConditions de la librería ThermoSysPro. En este caso se fija la presión (1 bar) y la temperatura (50ºC):

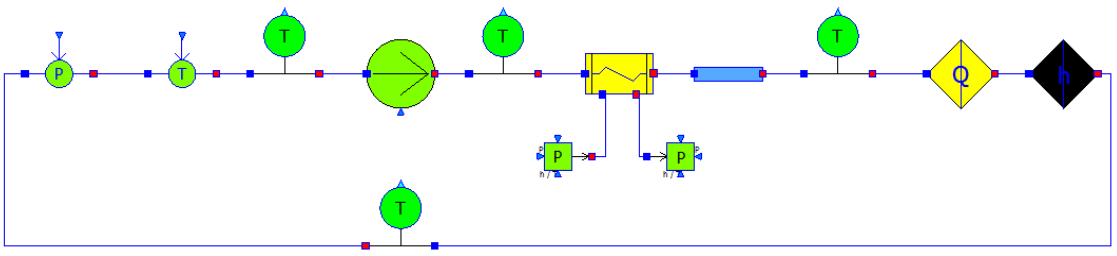
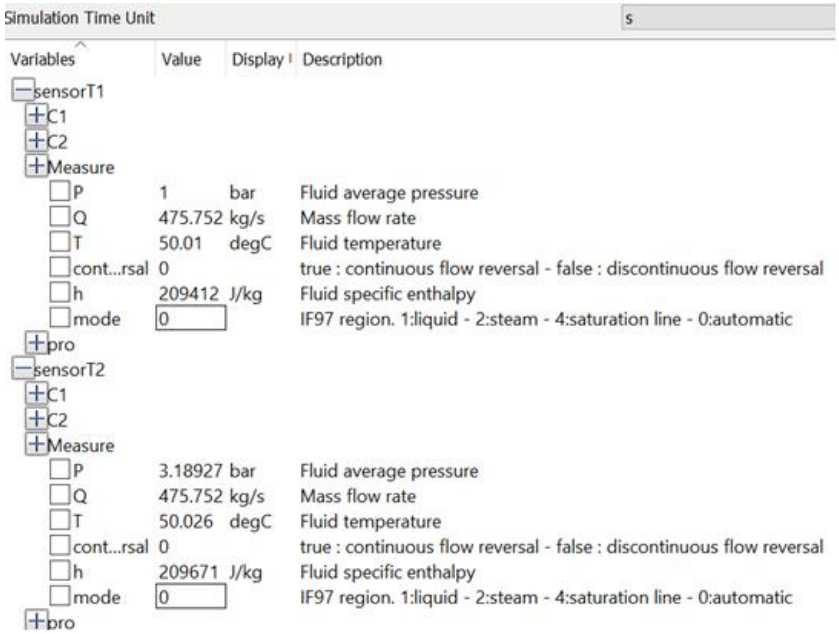


Figura 4.25: Modelo de ejemplo con valores de referencia y loopbreakers

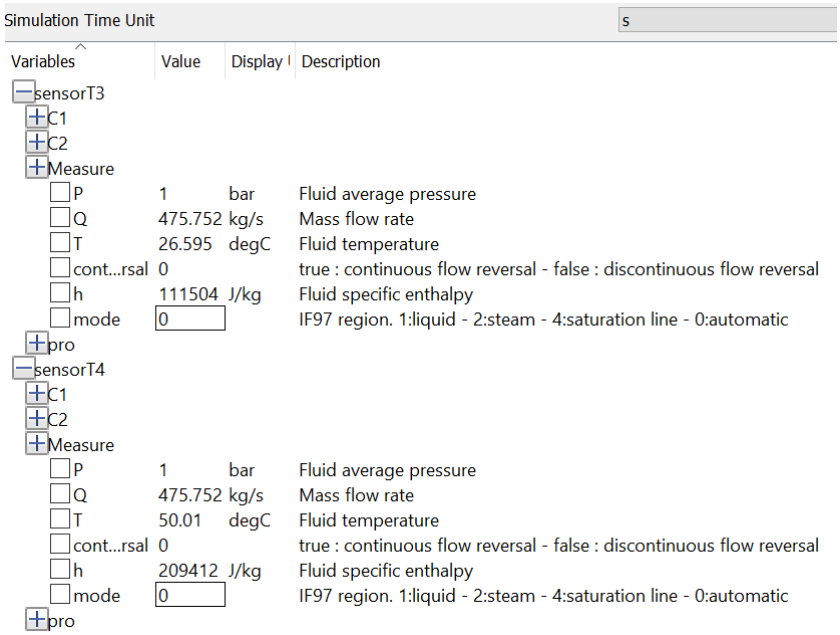
Para observar lo que pasa en el bucle, se han colocado sensores. Los sensores, empezando por el que se encuentra a la izquierda de la bomba y en sentido de las agujas del reloj, se le han nombrado como SensorT1, SensorT2, SensorT3, SensorT4.



Simulation Time Unit: s

Variables	Value	Display	Description
sensorT1			
C1			
C2			
Measure			
P	1	bar	Fluid average pressure
Q	475.752	kg/s	Mass flow rate
T	50.01	degC	Fluid temperature
cont...rsal	0		true : continuous flow reversal - false : discontinuous flow reversal
h	209412	J/kg	Fluid specific enthalpy
mode	0		IF97 region. 1:liquid - 2:steam - 4:saturation line - 0:automatic
pro			
sensorT2			
C1			
C2			
Measure			
P	3.18927	bar	Fluid average pressure
Q	475.752	kg/s	Mass flow rate
T	50.026	degC	Fluid temperature
cont...rsal	0		true : continuous flow reversal - false : discontinuous flow reversal
h	209671	J/kg	Fluid specific enthalpy
mode	0		IF97 region. 1:liquid - 2:steam - 4:saturation line - 0:automatic
pro			

Figura 4.26: Valores de las variables de los sensores T1 y T2



Simulation Time Unit: s

Variables	Value	Display	Description
sensorT3			
C1			
C2			
Measure			
P	1	bar	Fluid average pressure
Q	475.752	kg/s	Mass flow rate
T	26.595	degC	Fluid temperature
cont...rsal	0		true : continuous flow reversal - false : discontinuous flow reversal
h	111504	J/kg	Fluid specific enthalpy
mode	0		IF97 region. 1:liquid - 2:steam - 4:saturation line - 0:automatic
pro			
sensorT4			
C1			
C2			
Measure			
P	1	bar	Fluid average pressure
Q	475.752	kg/s	Mass flow rate
T	50.01	degC	Fluid temperature
cont...rsal	0		true : continuous flow reversal - false : discontinuous flow reversal
h	209412	J/kg	Fluid specific enthalpy
mode	0		IF97 region. 1:liquid - 2:steam - 4:saturation line - 0:automatic
pro			

Figura 4.27: Valores de las variables de los sensores T3 y T4

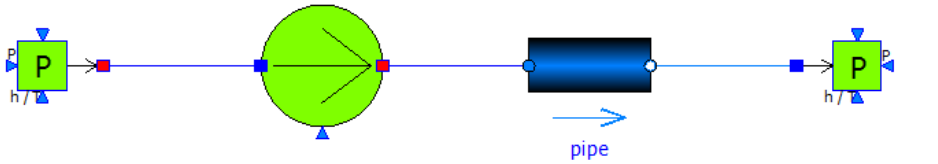
Observando solo la temperatura, se observa como empezando con una temperatura de referencia de 50ºC, tras pasar por la bomba, el sensorT2 sigue marcando

aproximadamente 50ºC (apenas tiene influencia en la temperatura, como es de esperar). Al pasar por el intercambiador de calor, el sensorT3 marca 26.595ºC. Posteriormente al colocar el loopbreaker de entalpía, se "corta" el valor de la variable, volviendo a marcar el sensorT4, los 50ºC de referencia.

Para lo confección de los ejemplos de la sección 4.5 y de la planta total se utilizarán algunos de estos loopbreakers y condiciones de contorno de referencia.

4.4.3. Adaptadores

Cuando se intenta conectar modelos de diferentes librerías, como en el ejemplo de abajo, se produce el siguiente error de traducción:



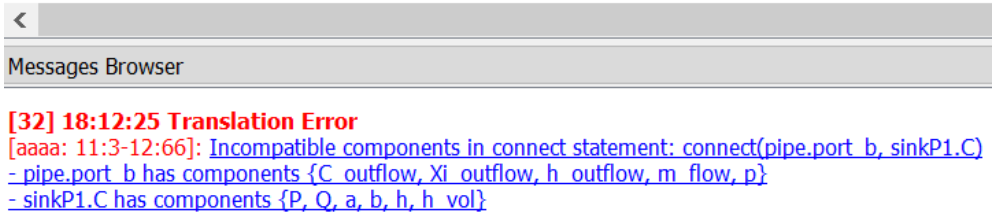


Figura 4.28: Error de traducción al conectar componentes de las librerías ThermoSysPro y MSL

En este caso, las condiciones de contorno y la bomba de circulación proceden de la librería ThermoSysPro. Y la tubería pertenece a otra librería, en este caso a la MSL. El error se produce debido a que existe una incompatibilidad entre las conexiones de los componentes.

El error tiene que ver con las variables que se declaran en los conectores de los modelos. Como se explicó en el apartado 2: Herramienta software Modelica, existe una clase especial utilizada para los conectores de los modelos (connector). Esta clase sirve para almacenar la información que se comparte entre dos componentes que se conectan. Dadas las características de la planta que se pretende construir, los conectores suelen ser "puertos de fluido", ya que nuestra instalación es un sistema termohidráulico.

Las variables que se declaran en dichos conectores para las dos librerías anteriores (y que son los conectores de los modelos que se han utilizado para este trabajo) son las siguientes:

- ✓ Conector ThermoSysPro (en este caso, conector de entrada)

```
connector FluidInlet
  Modelica.SIunits.AbsolutePressure P(start=1.e5);
  Modelica.SIunits.SpecificEnthalpy h_vol(start=1.e5);
  Modelica.SIunits.MassFlowRate Q(start=500);
  Modelica.SIunits.SpecificEnthalpy h(start=1.e5);
  input Boolean a=true;
  output Boolean b;
  annotation (…);
end FluidInlet;
```

- ✓ Conector MSL/Buildings (en este caso, conector de entrada)

```
connector FluidPort
  replaceable package Medium =
Modelica.Media.Interfaces.PartialMedium
    annotation (choicesAllMatching=true);
  flow Medium.MassFlowRate m_flow;
  Medium.AbsolutePressure p;
  stream Medium.SpecificEnthalpy h_outflow;
  stream Medium.MassFraction Xi_outflow[Medium.nXi];
  stream Medium.ExtraProperty C_outflow[Medium.nC];
end FluidPort;
```

Existe algunas diferencias en las que no nos vamos a detener demasiado, como es el caso de como en las librerías MSL/Buildings, se utilizan la palabra clave flow/stream para declarar algunas variables. Esto se explica en el manual de OpenModelica accesible gratuitamente desde su web oficial. Se utiliza flow para declarar variables que son de flujo, como es el caso del caudal, o corriente eléctrica, por ejemplo. Cuando no se pone nada delante de la variable, significa que dicha variable no es de flujo (potencial), como puede ser la temperatura o la presión. Para variables algo más especiales, como es el caso de la entalpía, se creó la declaración stream que describe flujo bidireccional de materia.

Otra diferencia, es la declaración del package Medium en el conector de MSL/Buildings.

Con respecto al problema principal, se observa que las variables que son declaradas en cada conector son diferentes (por lo menos a nivel de escritura de código):

ThermoSysPro: P, h_vol, Q, h, a, b

MSL/Buildings: m_flow, p, h_outflow, Xi_outflow, C_outflow

P y p es la presión, Q y m_flow es caudal, h_vol es la entalpía en el volumen de control, h y h_outflow es la entalpía en la frontera del volumen de control, a y b son pseudo-variables (booleanas) para verificar la orientación de la conexión, y Xi_outflow, C_outflow hacen referencia a determinadas propiedades como mezcla de fracciones de masas independientes y otras propiedades.

Como se puede deducir, dichas variables que son las que le sirven a los modelos para comunicar su estado a otros modelos, y transmitir su información, semánticamente son

muy semejantes, pero el software detecta incompatibilidad por estar escritas con diferentes signos.

Una solución puede ser modificar el código de los modelos. Esto es bastante problemático, ya que la forma de creación de modelos (herencia y jerarquía de clases) en las librerías MSL y Buildings hace la tarea imposible, debido a las continuas referencias que se hacen entre los modelos.

Mencionar que se ha intentado hacer funcionar los modelos cambiando solamente los componentes de las librerías ThermoSysPro, que presentan un acceso al código mucho más fácil y entendible. Pero no se ha conseguido hacer funcionar los modelos.

Por lo tanto, la única solución que ha resultado ser la acertada, es la creación de adaptadores, que permiten que los modelos se puedan comunicar correctamente colocándolos entre los componentes de distintas librerías.

Dadas las diferentes necesidades, se han creado tipos diferentes de adaptadores. Unos son puertos para fluidos, otros, para señales de control fundamentalmente. Los adaptadores son los siguientes:

- AdapWaterMT

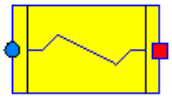


Figura 4.29: Icon View del adaptador AdapWaterMT

Permite conectar un modelo de MSL/Buildings a uno de ThermoSysPro.

Tiene como conector de entrada la clase *connector:*

`Modelica.Fluid.Interfaces.FluidPort_a.`

Como conector de salida tiene declarada la clase *connector:*

`ThermoSysPro.WaterSteam.Connectors.FluidOutlet`

- AdapWaterTM

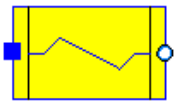


Figura 4.30: Icon View del adaptador AdapWaterTM

Permite conectar un modelo de ThermoSysPro a uno de MSL/Buildings.

Tiene como conector de entrada la clase *connector:*

`ThermoSysPro.WaterSteam.Connectors.FluidInlet`

Como conector de salida tiene declarada la clase *connector:*

`Modelica.Fluid.Interfaces.FluidPort_b.`

- AdapRealMT

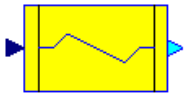


Figura 4.31: Icon View del adaptador AdapRealMT

Permite conectar un modelo de MSL/Buildings a uno de ThermoSysPro.

Tiene como conector de entrada la clase *connector:*

`Modelica.Blocks.Interfaces.RealInput`

Como conector de salida tiene declarada la clase *connector:*

`ThermoSysPro.InstrumentationAndControl.Connectors.OutputReal`

- AdapRealTM

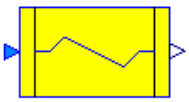


Figura 4.32: Icon View del adaptador AdapRealTM

Permite conectar un modelo de ThermoSysPro a uno de MSL/Buildings.

Tiene como conector de entrada la clase *connector:*

`ThermoSysPro.InstrumentationAndControl.Connectors.InputReal`

Como conector de salida tiene declarada la clase *connector:*

`Modelica.Blocks.Interfaces.RealOutput`

- AdapBooleanMT

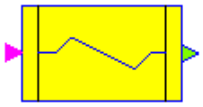


Figura 4.33: Icon View del adaptador AdapBooleanMT

Permite conectar un modelo de MSL/Buildings a uno de ThermoSysPro.

Tiene como conector de entrada la clase *connector:*

`Modelica.Blocks.Interfaces.BooleanInput`

Como conector de salida tiene declarada la clase *connector:*

`ThermoSysPro.InstrumentationAndControl.Connectors.OutputLogical`

Los códigos de cada uno de los adaptadores se encuentran en el Anexo D: Text View Adaptadores.

Un problema derivado de las variables que se definen en los conectores de los modelos de las librerías presentes en OpenModelica, es, el de si se quiere trabajar con variables distintas a las que aparecen en los conectores vistos. Este problema se dio en la confección de los modelos alternativos presentados en la sección 4.3, en el que se creó modelos para una máquina de absorción y bombas de calor. Estos modelos trabajan con las temperaturas de entrada y salida del modelo. Para estos casos, se hace necesario buscar una función que determine las temperaturas en los conectores en función de otras variables de estado definidas en los puertos.

Las dos funciones utilizadas en los modelos son:

`ThermoSysPro.Properties.WaterSteam.IF97.Water_Ph`

`ThermoSysPro.Properties.WaterSteam.IF97.Water_PT`

En los modelos, también se hace uso de las clases especiales record (recordar que se utiliza para encapsular propiedades):

`ThermoSysPro.Properties.WaterSteam.Common.ThermoProperties_ph`

`ThermoSysPro.Properties.WaterSteam.Common.ThermoProperties_pT`

En los siguientes trozos de código de las dos funciones se observa su funcionamiento:

```
function Water_Ph
  input Modelica.SIunits.AbsolutePressure p "Pressure";
  input Modelica.SIunits.SpecificEnthalpy h "Specific enthalpy";
  input Integer mode = 0 "IF97 region. 0:automatic";

  output
ThermoSysPro.Properties.WaterSteam.Common.ThermoProperties_ph
pro;
```

```
function Water_PT
  input Modelica.SIunits.AbsolutePressure p "Pressure";
  input Modelica.SIunits.Temperature T "Temperature";
  input Integer mode=0 "IF97 region. 0:automatic";

  output
ThermoSysPro.Properties.WaterSteam.Common.ThermoProperties_pT
pro;
```

Por lo tanto para Water_Ph, introduciendo la presión, la temperatura, y la región del medio IF97 (sabiendo que con dos variables de estado termodinámicas, se nos permite caracterizar el estado de un determinado medio), de salida, se rellena todas las variables que aparecen en las clases record antes mencionadas.

En el caso de Water_PT, el proceso es el mismo, pero se introduce la presión, la temperatura y la región, y de salida, se completa la clase record correspondiente.

En dichas clases se declaran las variables densidad, energía interna específica, temperatura, entropía específica, etc.

Ahora solo falta obtener la variable que se busca, en este caso la temperatura. Si, por ejemplo, a la función: `ThermoSysPro.Properties.WaterSteam.IF97.Water_PT`, se le denomina proce, el código quedaría de la siguiente forma:

```
proce = ThermoSysPro.Properties.WaterSteam.IF97.Water_Ph(P, h, 0);

T = proce.T;
```

Todo este proceso, se puede observar con mayor claridad en el Anexo C: Text View Bombas de calor y Maquina de absorción, donde están los códigos de los modelos.

4.4.4. Ecuaciones de balance

Muchos de los modelos, sobre todo los pertenecientes a las librerías MSL y Buildings, permite definir la formulación de las ecuaciones de balance (seleccionable a través del menú de opciones: Parameters -> Dynamics -> Equations -> ej: energyDynamics)

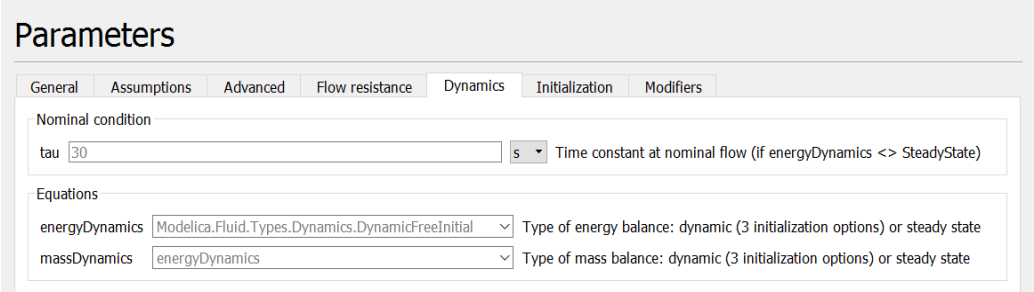


Figura 4.34: Pestaña Dynamics donde se permite configurar ecuaciones de balance

Todas las elecciones posibles se muestran en la siguiente tabla:

Modelica.Fluid.Types.Dynamics.DynamicFreeInitial

Modelica.Fluid.Types.Dynamics.FixedInitial

Modelica.Fluid.Types.Dynamics.SteadyStateInitial

Modelica.Fluid.Types.Dynamics.SteadyState

Tabla 4.21: Diferentes configuraciones de dinámica en Modelica

Dinámica	Significado
DynamicFreeInitial	Balance Dinámico, valor inicial supuesto
FixedInitial	Balance Dinámico, valor inicial fijado
SteadyStateInitial	Balance Dinámico, Régimen permanente con valor inicial supuesto
SteadyState	Balance Régimen estacionario, valor inicial supuesto

La "enumeración Dinámica" se utiliza para las ecuaciones de masa, energía y equilibrio de momentum. El significado exacto de las tres ecuaciones de equilibrio se indica en las siguientes tablas:

Tabla 4.22: Descripción de cada configuración para Balance de masa

Balance de masa		
Dinámica	**Ecuación de Balance**	**Condición Inicial**

DynamicFreeInitial	No restricciones	No condiciones iniciales
FixedInitial	No restricciones	Si Medium.singleState entonces no condiciones iniciales sino p = p_start
SteadyStateInitial	No restricciones	Si Medium.singleState entonces no condiciones iniciales sino der(p) = 0
SteadyState	der(m) = 0	No condiciones iniciales

Tabla 4.23: Descripción de cada configuración para Balance de energía

Balance de energía		
Dinámica	**Ecuación de Balance**	**Condición Inicial**
DynamicFreeInitial	No restricciones	No condiciones iniciales
FixedInitial	No restricciones	T = T_start ó h = h_start
SteadyStateInitial	No restricciones	der(T) = 0 ó der(h) = 0
SteadyState	der(U) = 0	No condiciones iniciales

Tabla 4.24: Descripción de cada configuración para Balance de momentum

Balance de momentum		
Dinámica	**Ecuación de Balance**	**Condición Inicial**
DynamicFreeInitial	No restricciones	No condiciones iniciales
FixedInitial	No restricciones	m_flow = m_flow_start
SteadyStateInitial	No restricciones	der(m_flow) = 0
SteadyState	der(m_flow) = 0	No condiciones iniciales

En las tablas anteriores, las ecuaciones se dan para fluidos de una sustancia. Para fluidos de sustancias múltiples y/o trazas, se mantienen ecuaciones equivalentes.

Medium.singleState es una propiedad del Medio que define si el medio solo es descrito por un estado (mas las fracciones de masa en el caso de un fluido multisustancia). En tal caso, debe proporcionarse una condición inicial menos. Por ejemplo, los medios incompresibles tienen Medium.singleState = true.

Para ilustrar como cambia la respuesta de determinadas variables en función del tipo de ecuaciones de balance, utilizaremos el ejemplo de un calentador que está conectado a una fuente (BoundaryConditions) a 35ºC. La salida de temperatura del calentador (Tset) es impuesta por nosotros. En este caso variará como un pulso, que tiene 25ºC de amplitud, un periodo de 200 s y un ancho de pulso de 100 s.

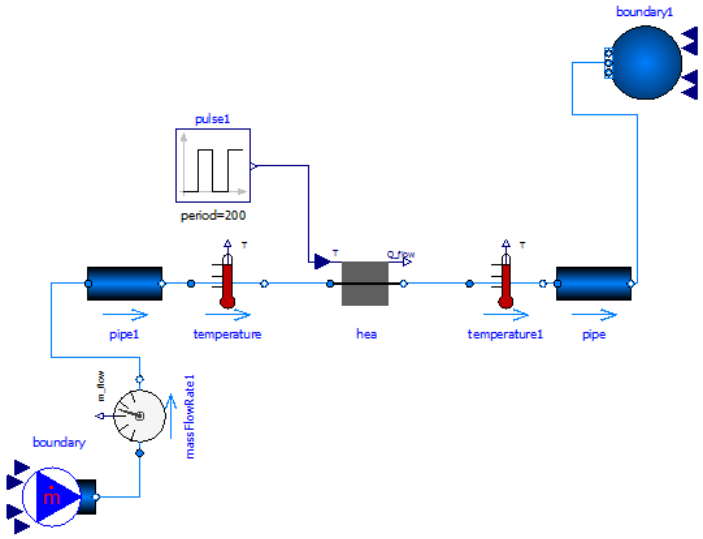


Figura 4.35: Ejemplo de un calentador ideal para experimentar con configuraciones dinámicas diferentes

Se permite variar el tipo de balance energético, con 3 opciones de inicialización dinámicas (antes mencionadas) y régimen estacionario. En inicialización, el valor inicial o supuesto será T_start = Medium.T_default, que corresponde con 20ºC.

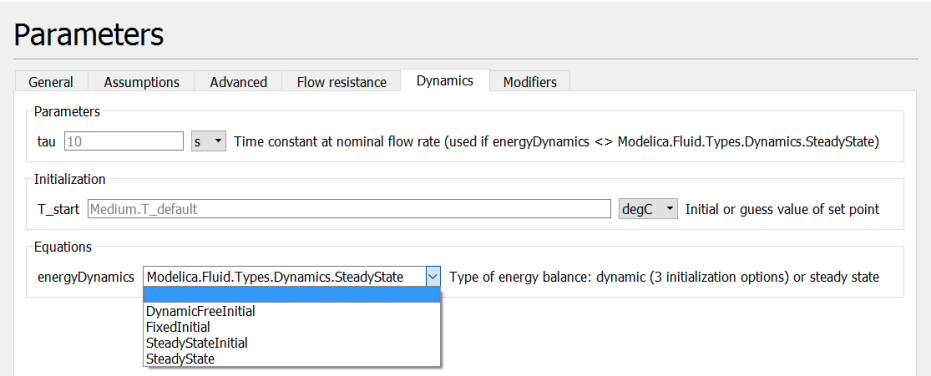


Figura 4.36: Diferentes tipos de personalización en la pestaña dynamics para el modelo calentador

Las respuestas de la temperatura a la salida del calentador son las siguientes (el tiempo de simulación es de 500 s):

- DynamicFreeInitial:

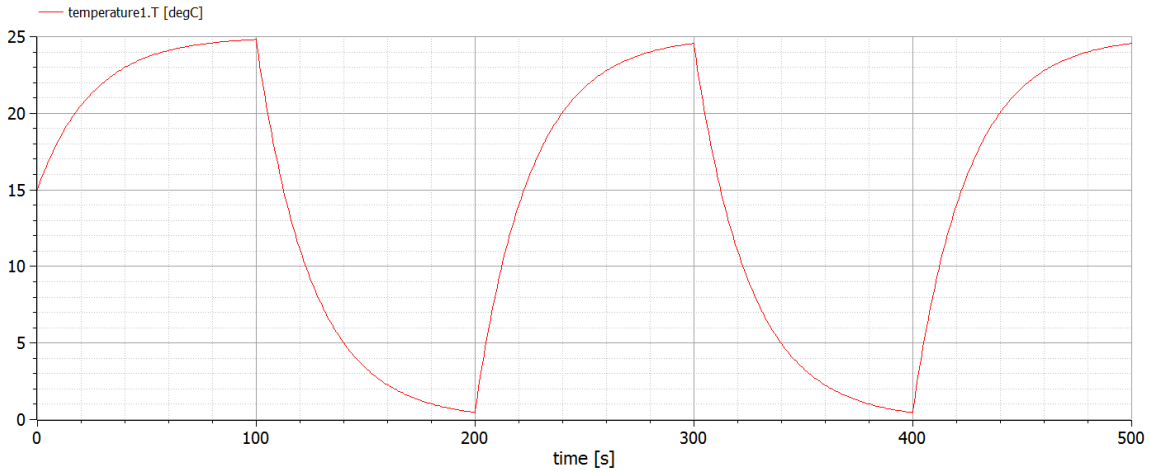


Figura 4.37: Respuesta de la variable temperatura del sensor temperature1 para DynamicFreeInitial

- FixedInitial

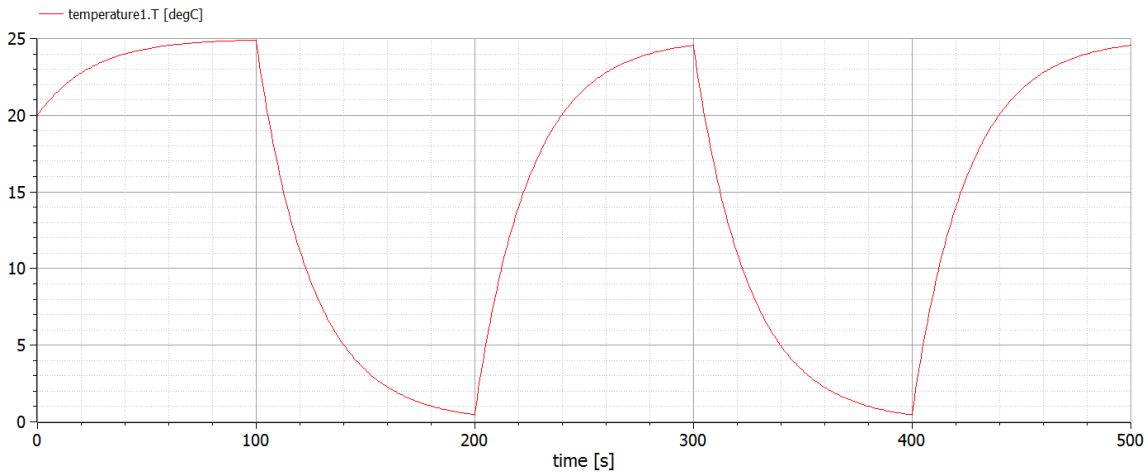


Figura 4.38: Respuesta de la variable temperatura del sensor temperature1 para FixedInitial

- SteadyStateInitial

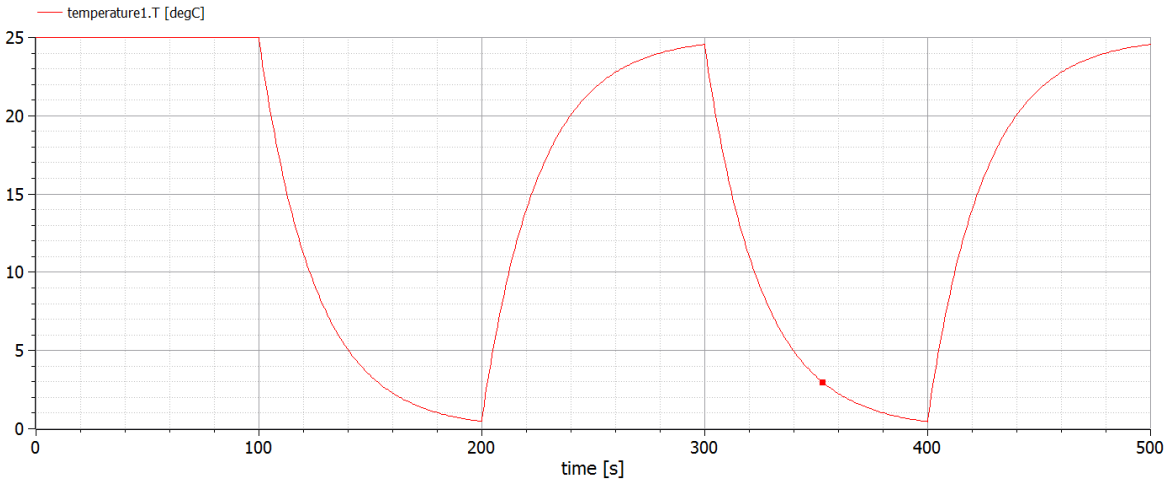


Figura 4.39: Respuesta de la variable temperatura del sensor temperature1 para SteadyStateInitial

- SteadyState

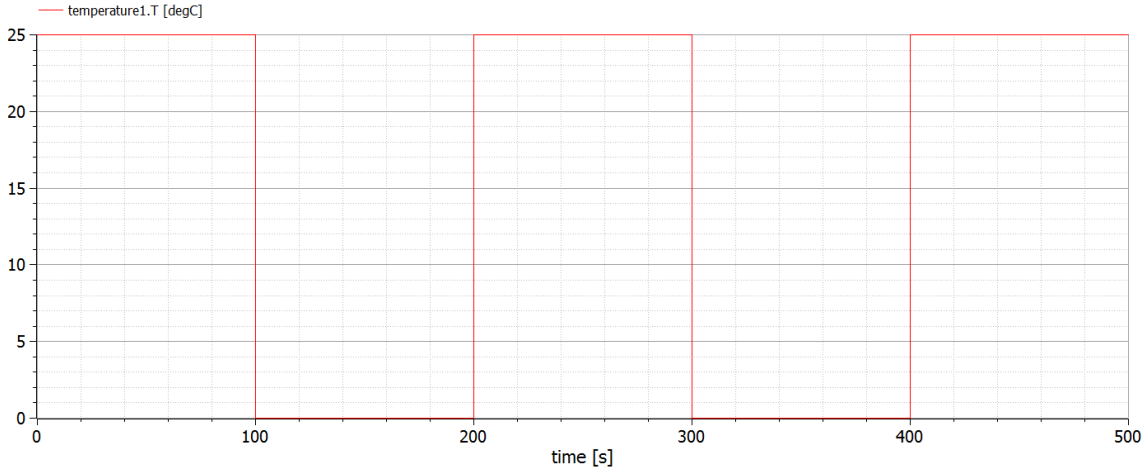


Figura 4.40: Respuesta de la variable temperatura del sensor temperature1 para SteadyState

Como se observa en las curvas las condiciones iniciales (los valores de inicialización) son diferentes. En DynamicFreeInitial el valor de la temperatura comienza en 15ºC. Es un valor estimado que el software da como valor de inicialización. En FixedInitial el valor inicial es fijado por nosotros. En este caso corresponde a 20ºC (T_start = Medium.T_default) pero podemos poner el que queramos. Tanto en SteadyStateInitial como SteadyState, se comienza inmediatamente en el valor de salida esperado, siguiendo (Tset (amplitud) = 25).

La decisión de que ecuaciones elegir depende de las características de nuestro estudio. Si el cálculo se quiere hacer en régimen permanente (no existen transitorios debidos a periodos de arranque, parada o cambio de régimen) o transitorio.

Como también se ilustra en las imágenes, la forma de la curva es diferente. Cuando el balance energético es dinámico se tiene en cuenta el parámetro ***tau.*** Ésta constante de tiempo, que aparece en una ilustración anterior, está por defecto en 10 s, y modifica la forma de la curva y como sigue a la señal pulso (Tset). Con un tau menor, más se parecerá la curva a la señal pulso.

4.4.5. Otros problemas de simulación

En un principio, la mayoría de las simulaciones con el fin de probar los modelos, resultaron fallidas. Algunos de estos errores eran por causas anteriormente dichas. Sin embargo, subrayaré algunas incidencias o puntos a tener en cuenta que difícilmente se han podido encontrar solución a todas.

En primer lugar uno de los errores de traducción más comunes que aparecen en el navegador de mensajes (Messages Browser) es:

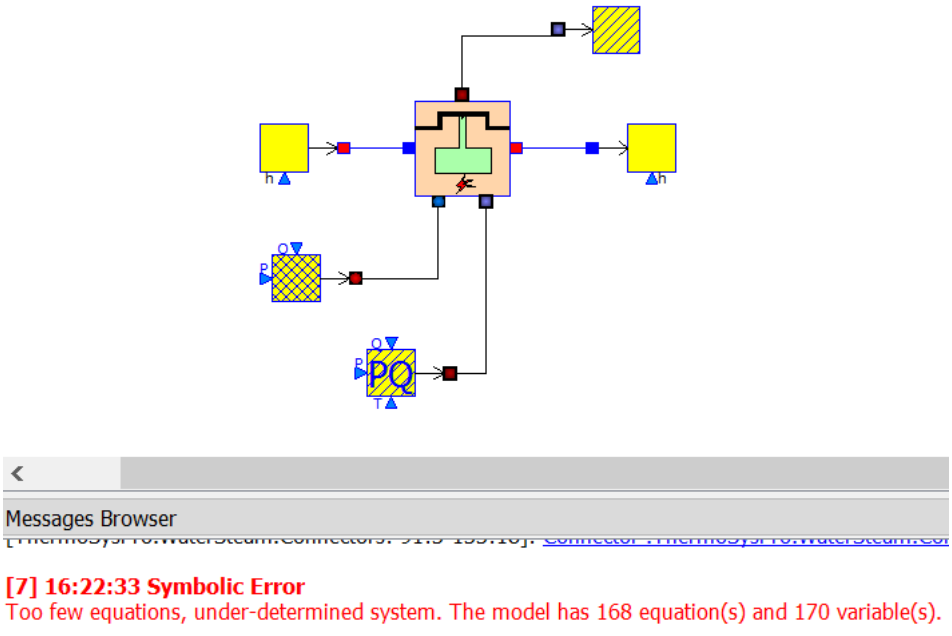


Figura 4.41: Error al simular por tener un sistema con menos ecuaciones que variables

Para empezar una simulación es imprescindible que el modelo tenga el mismo número de ecuaciones que de variables. Si no se produce el error simbólico aparecido en la imagen y el sistema no es determinado. Estos errores se pueden solventar eligiendo las condiciones de contorno apropiadas para cada modelo, o utilizando loopbreakers. Claro

está, suponiendo que el modelo está bien construido y no falta ninguna variable en el que hace falta asignarle algún valor.

En el ejemplo anterior, que es un motor alternativo con salida eléctrica, se soluciona el problema utilizando las condiciones de contorno apropiadas, para que no existan ecuaciones redundantes en las ecuaciones.

Otro error que se puede dar en el proceso de simulación es el siguiente (el modelo corresponde a un desorber de la librería ThermoSysPro):

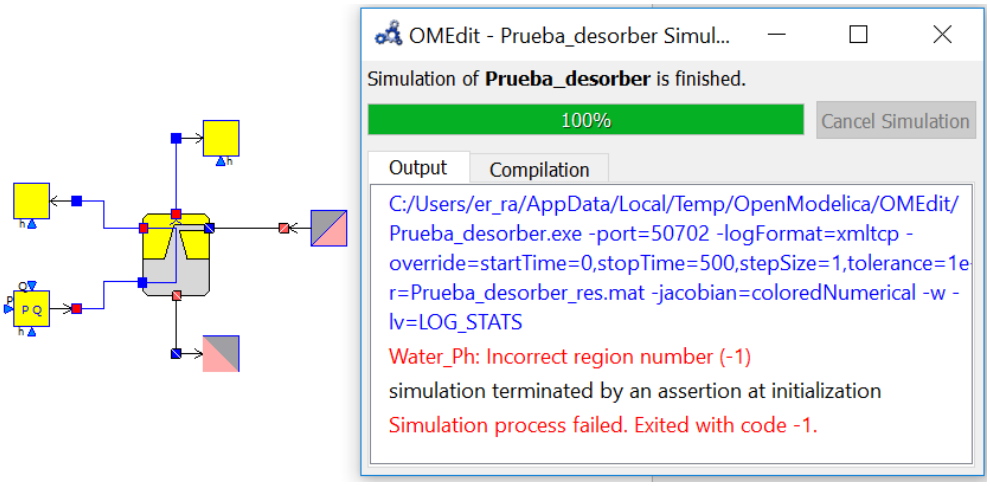


Figura 4.42: Error al simular por incorrecto número de región (Water_Ph)

En este caso se indica que el error de simulación es debido a:

Water_Ph : Incorrect region number (-1)

Estos tipos de errores son muy comunes en la suite OpenModelica. Se debe a que el modelo está trabajando en unas condiciones de operación que invalida el funcionamiento de determinadas funciones, ya que muchas de éstas incorporan valores límites para determinadas variables.

En este caso la función se encuentra en este paquete:

- *ThermoSysPro.Properties.WaterSteam.IF97_packages.IF97_vAJ.Water_Ph*

Para entender porque la región es incorrecta hay que entender lo escrito en el código específico de esta función.

El modelo desorber anterior se soluciona, por ejemplo, poniendo una entalpia de entrada (en la fuente de agua) de 250000 en vez de 100000 (como está por defecto).

Para cerrar este apartado, que a lo largo de la utilización del entorno de modelado se ha encontrado dificultades para simular grandes transitorios, principalmente debido al tamaño del modelo:

- Pobre depuración
- Simulación lenta
- Gran número de valores que el usuario debe proporcionar manualmente para las variables de iteración
- No eficiente manejo de estos valores

En particular, se observa que a veces no se puede calcular las variables de estado incluso cuando todas las variables de iteración se establecen muy cerca de sus valores de solución:

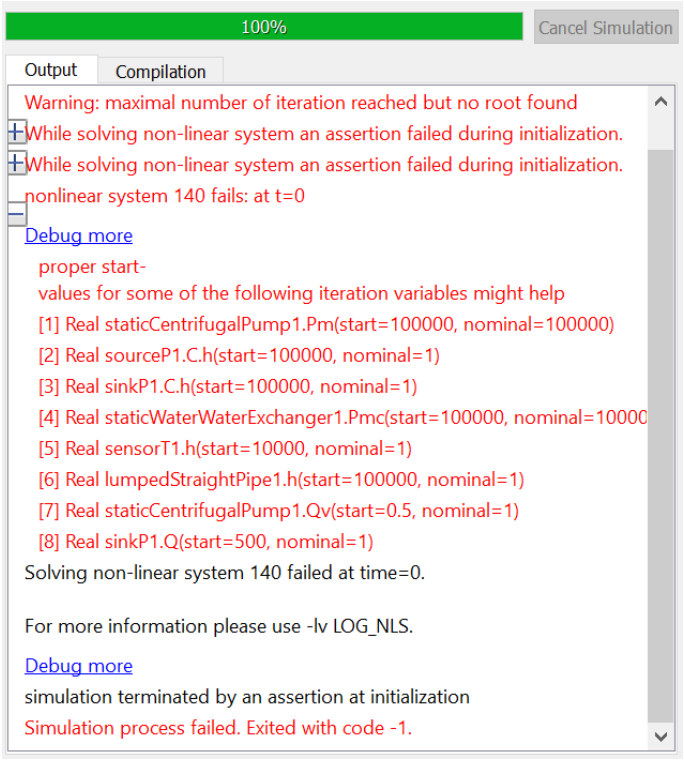


Figura 4.43: Error al simular por problemas de inicialización de variables

Esta fue una de las principales dificultades que se encontró al cerrar los bucles de muchos modelos. Cuando la simulación se detiene antes de finalizar, no se entrega un mensaje lo suficientemente claro para analizar las causas del fallo. Aunque se pongan los valores que se aconsejan, el error suele producirse de nuevo.

4.5. Validación y ejemplos

En este apartado se conectarán entre sí algunos de los modelos mostrados en la sección 4.2. En los ejemplos se expondrán los componentes que finalmente se utilizarán para la construcción de la instalación en OpenModelica. Aunque dadas las grandes posibilidades de testeo, se simulará los modelos bajo unas determinadas condiciones de operación y se mostrará la respuesta de algunas variables de importancia que puedan esclarecer o ratificar el funcionamiento de estos modelos presentados con anterioridad en este TFG. No se explicará la dinámica de cada modelo, el fin no es otro que el de comprobar que las respuestas son reproducibles y el ejemplo funciona. Otro fin de estos ejemplos es contemplar, además, la utilización de varios set points y sistemas de control que pueden ser de ayuda para una posterior optimización de la instalación. El código (text view) de los ejemplos se presentarán en el Anexo E: Text View Ejemplos. El tiempo de simulación de todos los ejemplos es de 500 segundos en todos los ejemplos.

4.5.1. Ejemplo 1

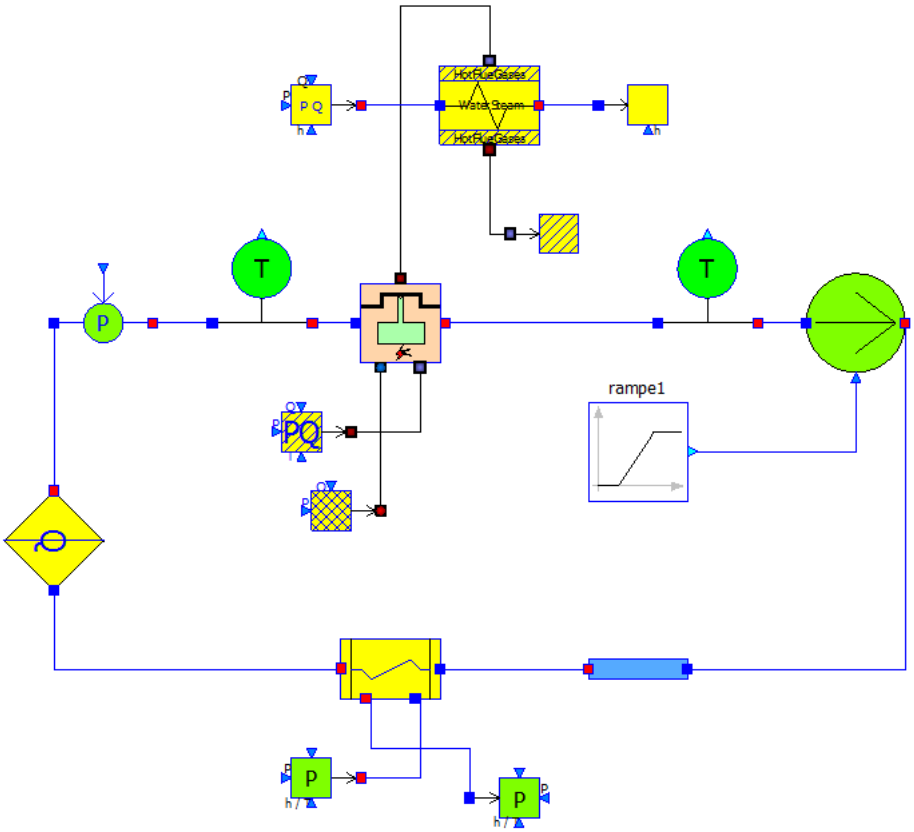


Figura 4.44: Diagram View del ejemplo 1

Como elementos auxiliares a los modelos reconocibles, se encuentran las fuentes y sumideros, tanto de agua como de combustible, gases y aire, que hacen de condiciones de contorno. Esto permite poder probar trozos de cualquier instalación sin tener que montarla

entera. Se hace uso de un loopbreaker, de caudal, y de un valor de presión de referencia en la entrada del motor a gas. Se utiliza una señal de control (rampa) para comunicarle a la bomba el caudal que se impondrá en el circuito. También se hace uso de dos sensores de temperatura. Los valores de todos los parámetros de cada uno de los componentes se pueden ver en el anexo anteriormente mencionado, ya que aparecen en el código de una forma muy clara.

El comportamiento del "sistema" ejemplo es el siguiente: El motor presenta una fuente de aire y otra de combustible. El proceso de combustión del motor hará que se libere energía térmica que será aprovechada por el circuito de refrigeración de agua. Por lo tanto el agua a la salida del motor, presentará mayor temperatura que a la entrada. Los gases del proceso de combustión serán recogidos por un intercambiador, que servirá para calentar otro circuito de agua. En este caso, el circuito no se ha dimensionado. Para ello, se ha utilizado dos condiciones de contorno. Igualmente, el agua intercambiará energía térmica con los gases de combustión, y su temperatura a la salida será mayor.

El caudal del circuito será impuesto por una bomba de circulación. Para ello, se ha utilizado una señal en rampa, que tiene un valor final de 1300. El agua procedente del motor se hace pasar por un intercambiador de agua/agua. La temperatura de entrada del agua fría es de 20 grados. Al pasar el agua del motor, transferirá calor a este circuito de agua fría (se ha utilizado condiciones de contorno), calentándola. El agua, menos caliente, procedente del motor, volverá a entrar en éste. La respuesta de algunas variables bajo estas condiciones de operación son las siguientes:

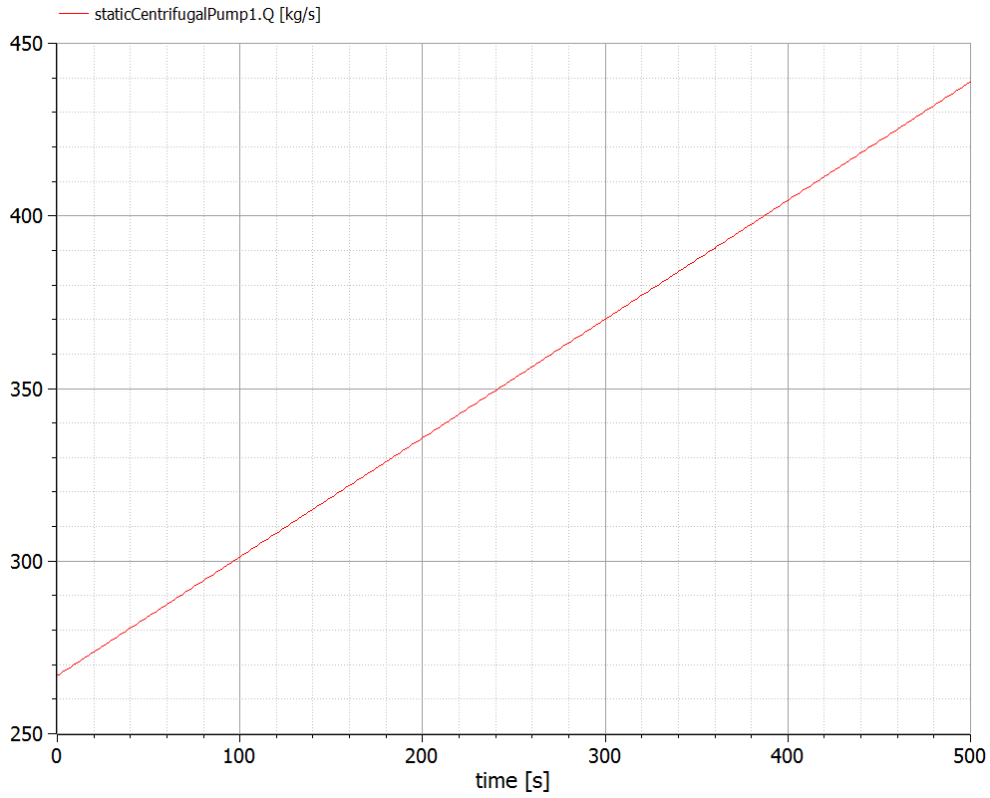


Figura 4.45: Respuesta del caudal de la bomba de circulación

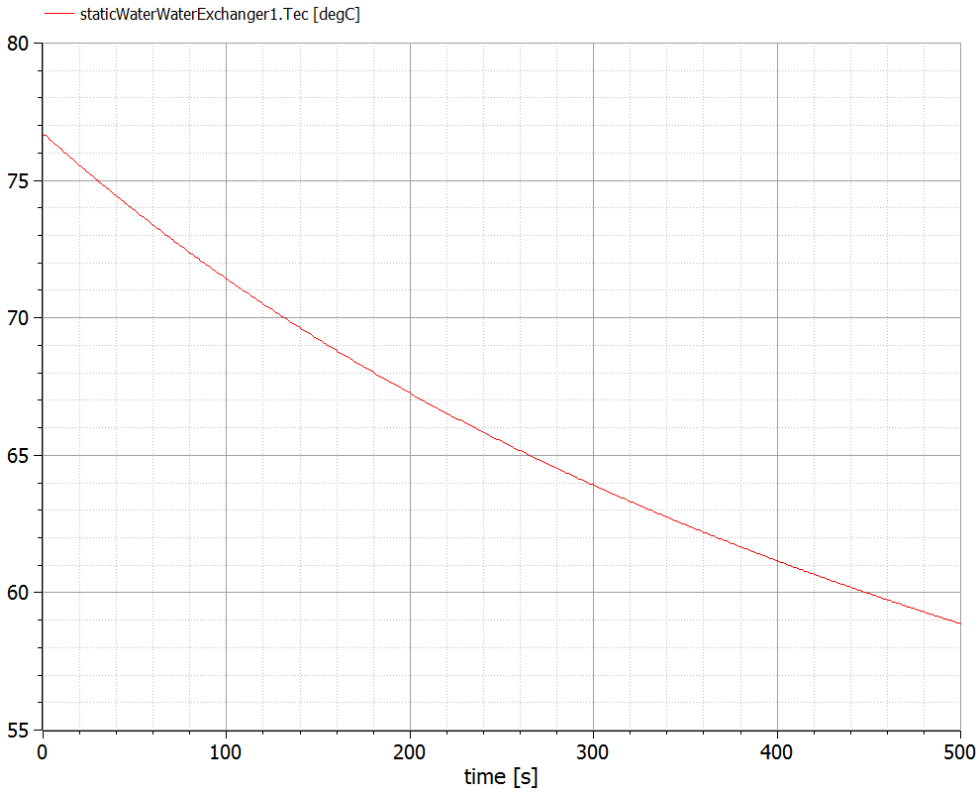


Figura 4.46: Respuesta de la temperatura de entrada del intercambiador agua/agua del fluido caliente

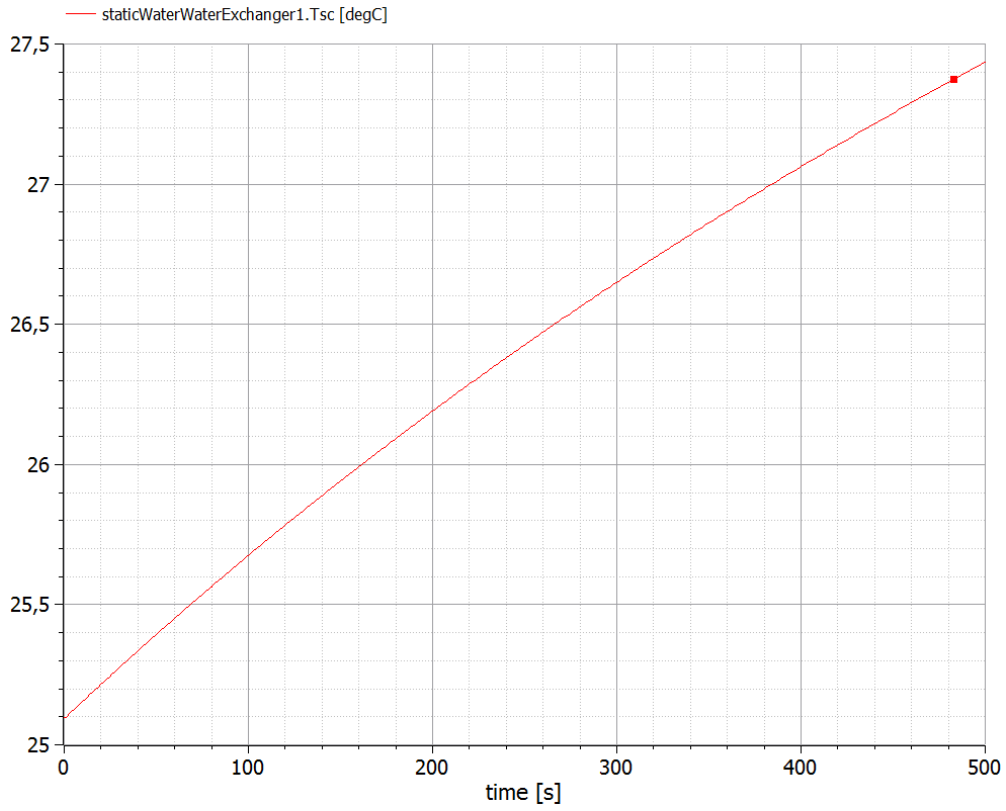


Figura 4.47: Respuesta de la temperatura de salida del intercambiador agua/agua del fluido caliente

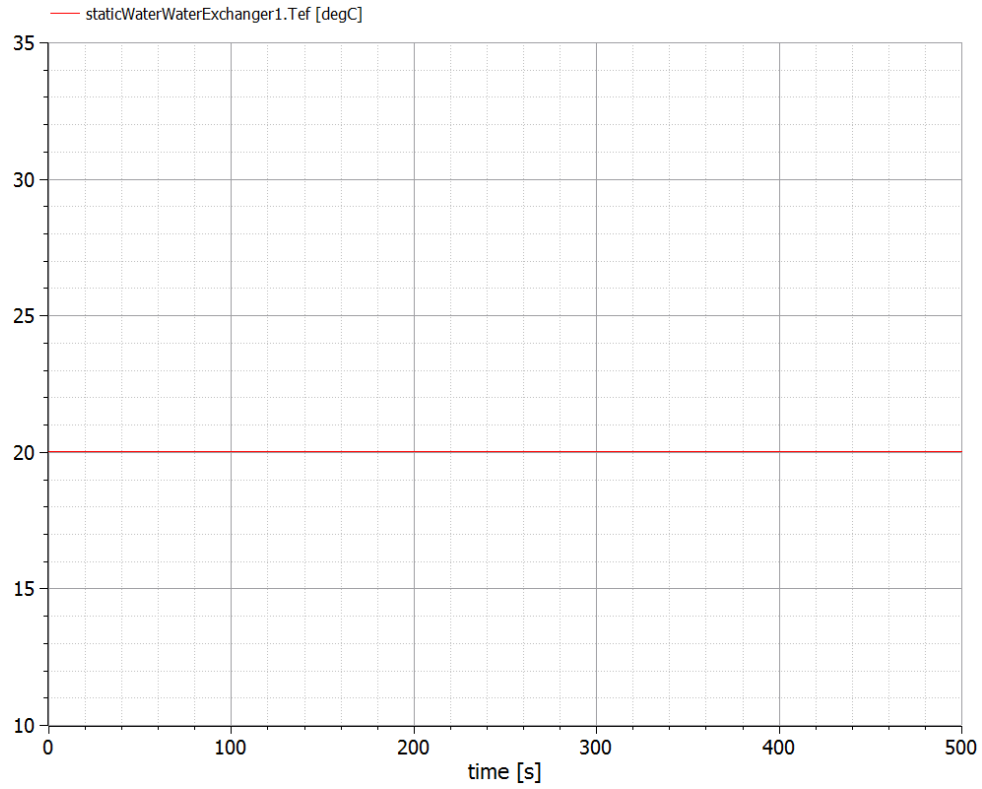


Figura 4.48: Respuesta de la temperatura de entrada del intercambiador agua/agua del fluido frío

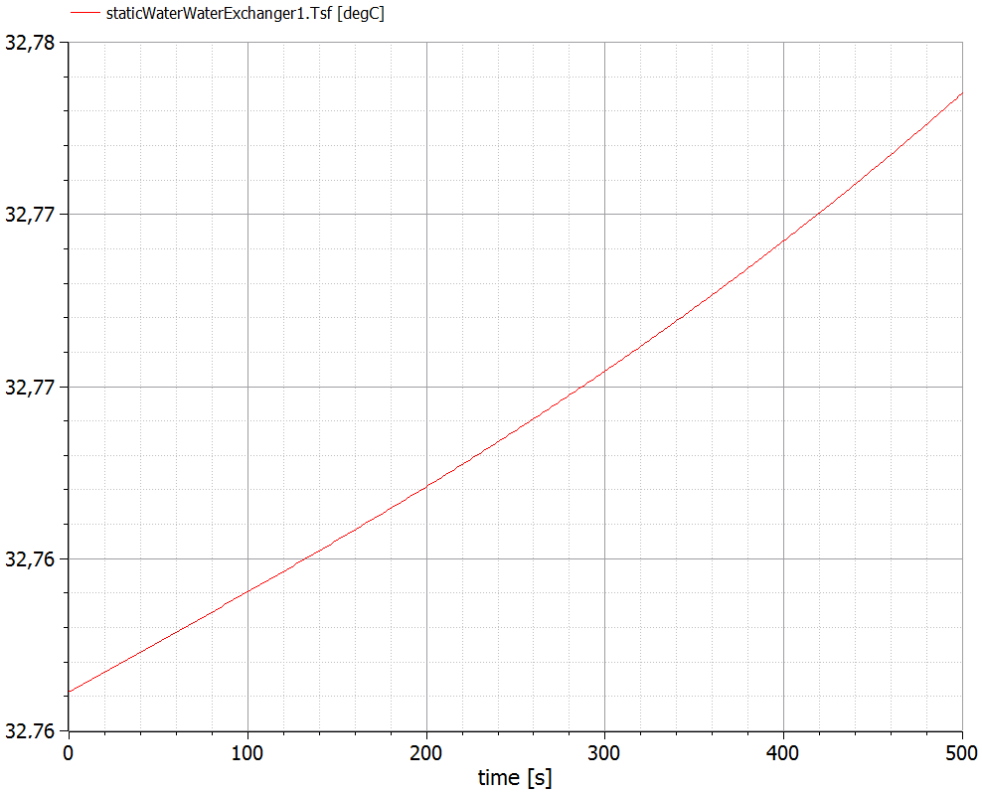


Figura 4.49: Respuesta de la temperatura de salida del intercambiador agua/agua del fluido frío

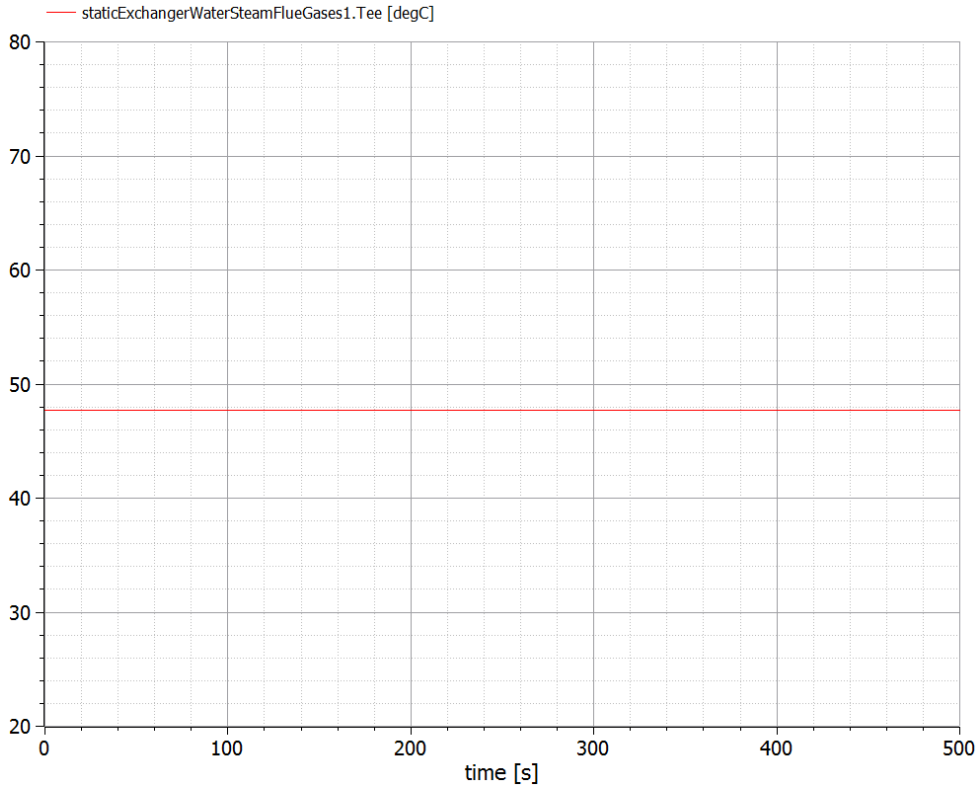


Figura 4.50: Respuesta de la temperatura de entrada del intercambiador agua/gases del agua

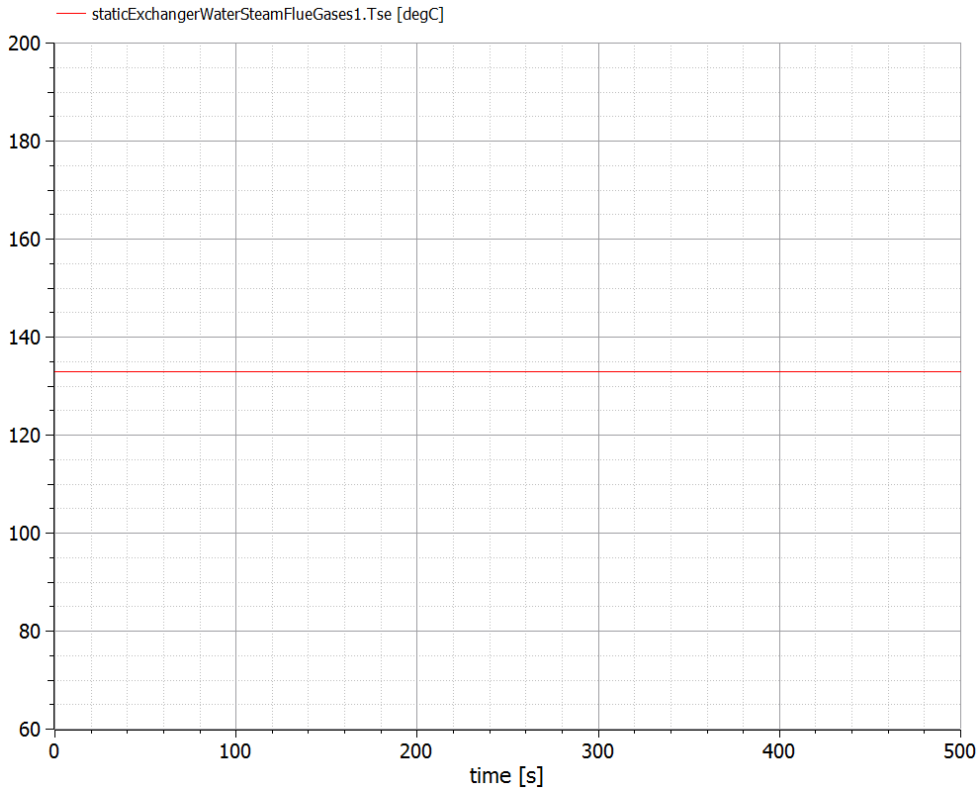


Figura 4.51: Respuesta de la temperatura de salida del intercambiador agua/gases del agua

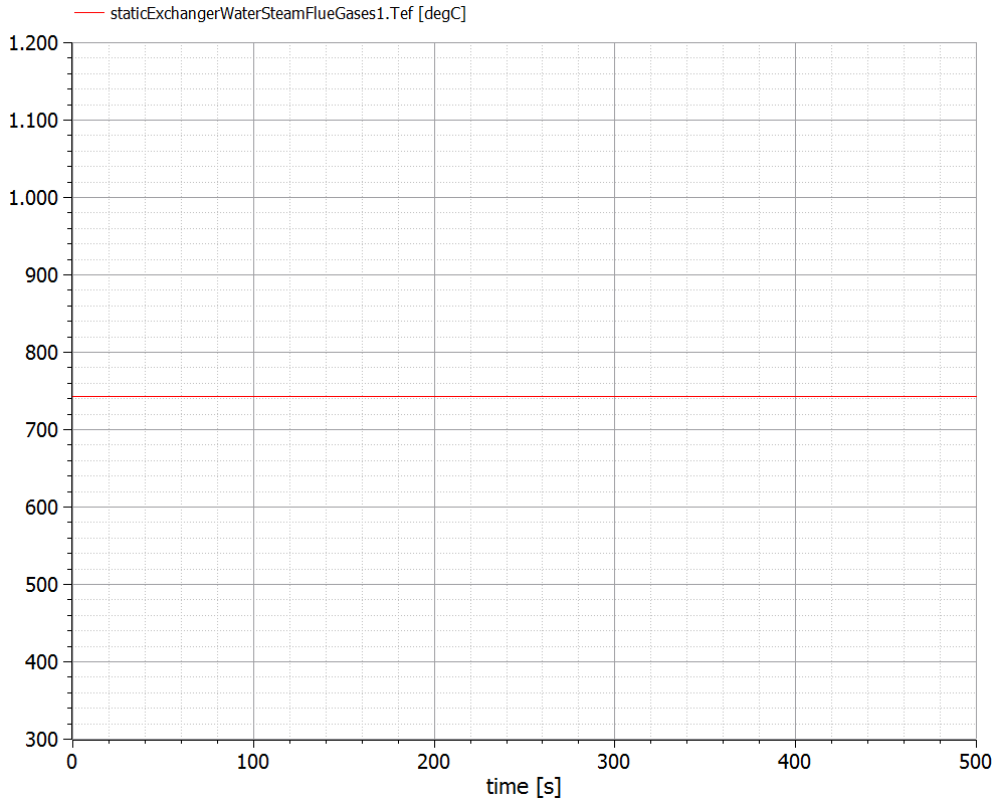


Figura 4.52: Respuesta de la temperatura de entrada del intercambiador agua/gases de los gases

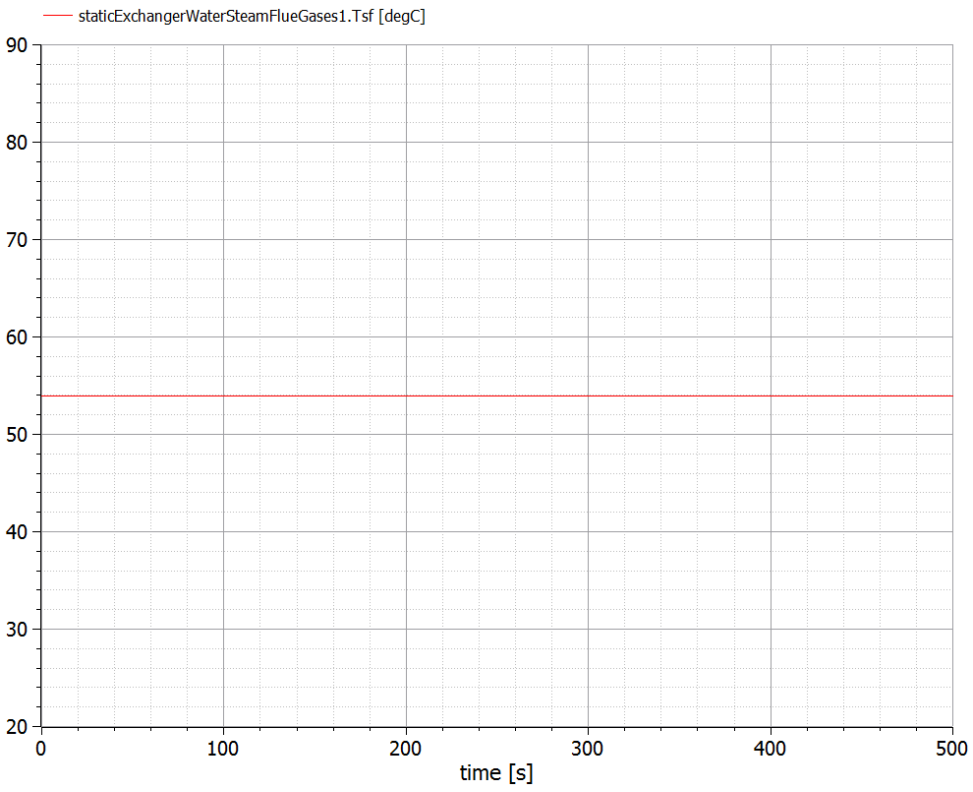


Figura 4.53: Respuesta de la temperatura de salida del intercambiador agua/gases de los gases

4.5.2. Ejemplo 2

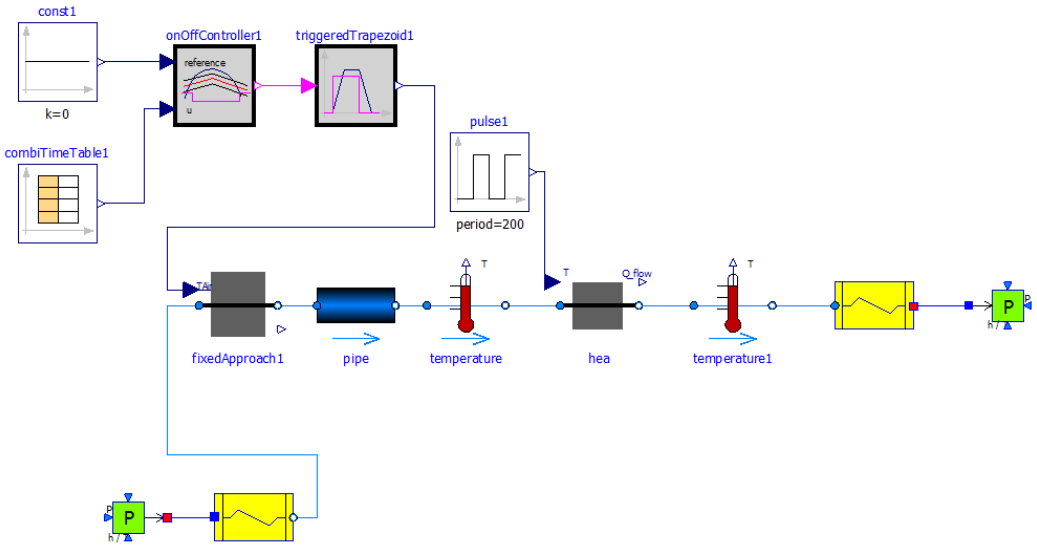


Figura 4.54: Diagram View del ejemplo 2

Este modelo consta de una torre de refrigeración y un calentador-refrigerador ideal que será utilizado como disipador en la instalación. La fuente y sumidero pertenecen a la librería ThermoSysPro. El resto de modelos corresponden a las librerías MSL y Buildings. Por lo tanto se toman dos adaptadores para que las variables sean compatibles y puedan transmitir información entre ellos correctamente. La temperatura de consigna de salida del disipador es un pulso. El pulso consta de un periodo de 200 segundos y un ancho de pulso de 100 s. Presenta un offset de 15ºC y una amplitud de 5ºC (sobre ese offset).

La torre de refrigeración permite configurar la temperatura del aire de la torre, esta intercambiará energía con el circuito de agua. La fuente de agua es de unos 35ºC.

Para trastear con las señales de control, se han implementado varios bloques para obtener la señal de esta temperatura. Los bloques son una señal constante de valor cero, una tabla que permite introducir valores de tiempo (primera columna) y valores reales en el resto de columnas, un controlador todo-nada y un triggered trapezoidal. El bloque y la señal de referencia servirán como entradas al controlador. Ambas señales se mostrarán en imágenes posteriores. El controlador entregará una respuesta booleana.

Si los valores de la tabla (se ha creado una curva con un suavizado: Modelica.Blocks.Types.Smoothness.MonotoneContinuousDerivative1) superan el valor de referencia, la respuesta del controlador es 0. Cuando la curva es menor al valor de referencia, la respuesta es 1.

Esta señal entrará en el triggered para generar una señal Real. Presenta un offset de 25ºC, con un valor de amplitud de 5ºC. El tiempo de subida y bajada es de 0 segundos.

Finalmente esta señal será la consigna de entrada en la torre.

Siguiendo al fluido de trabajo (agua), éste es enfriado en dos ocasiones, primeramente en la torre de refrigeración y posteriormente en el disipador. Las curvas son las siguientes:

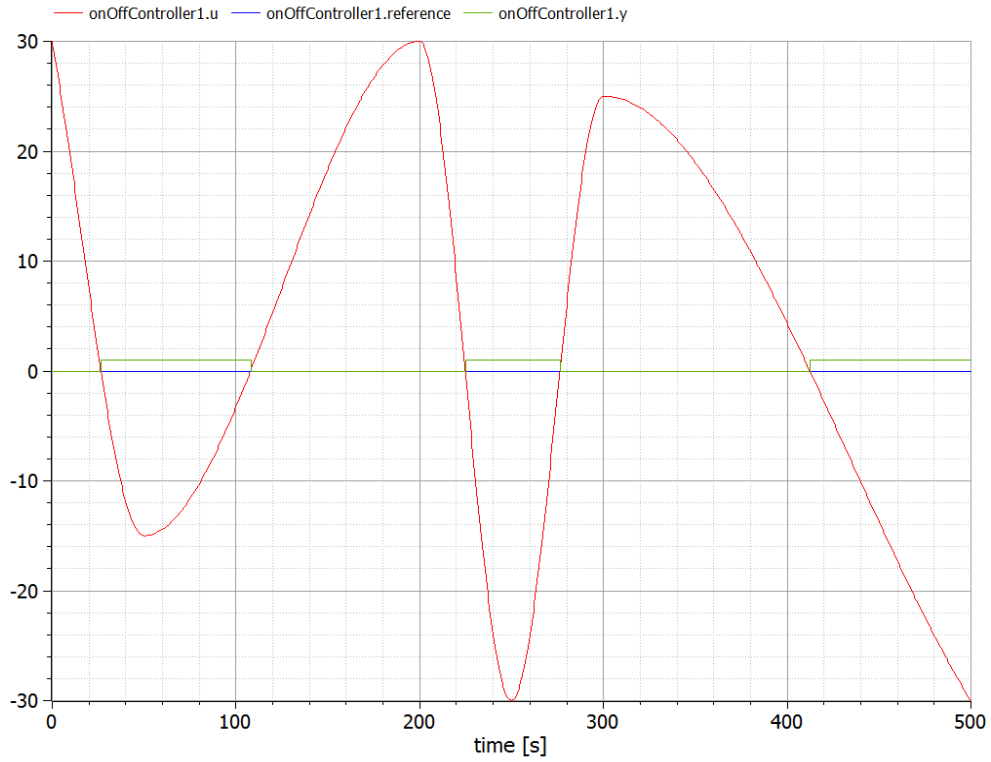


Figura 4.55: Respuesta de la entrada u, entrada reference y salida y del controlador on-off

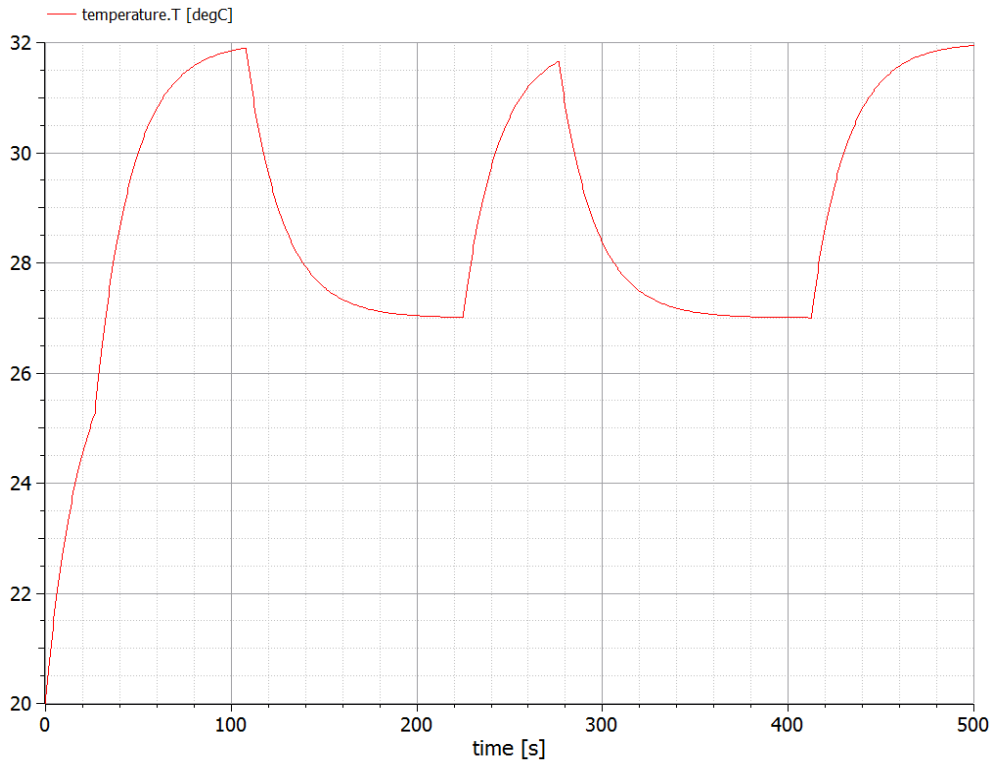


Figura 4.56: Respuesta de la temperatura del sensor temperature (salida de la torre de refrigeración)

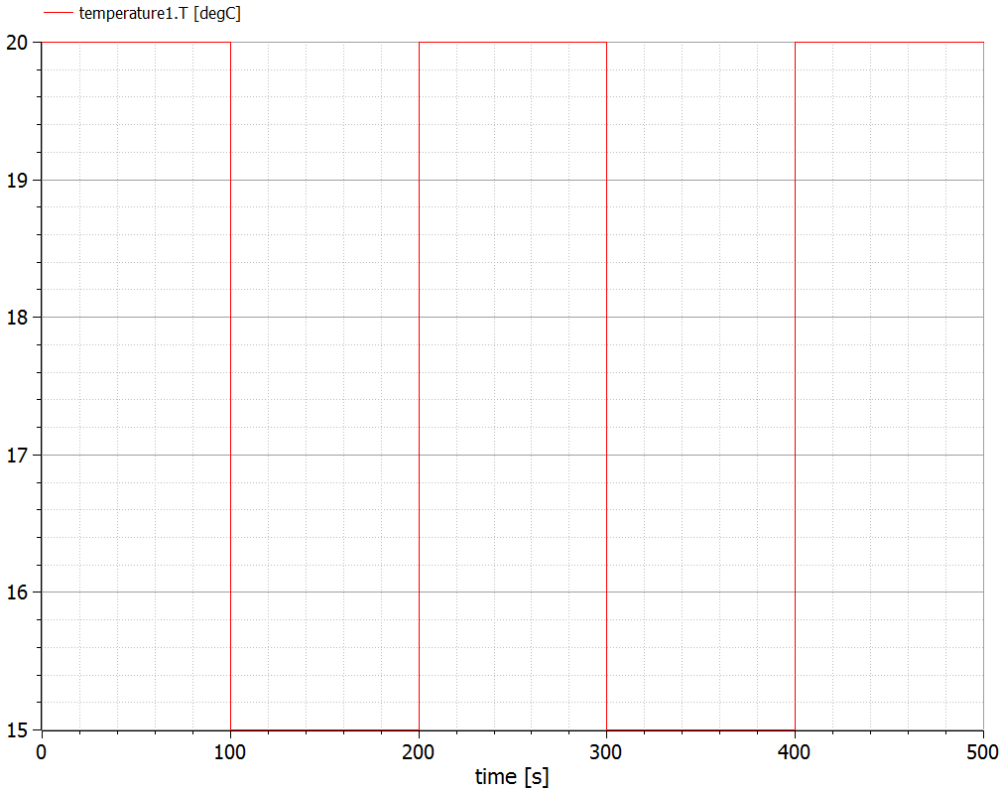


Figura 4.57: Respuesta de la temperatura del sensor temperature1 (salida impuesta en el calentador-refrigerador)

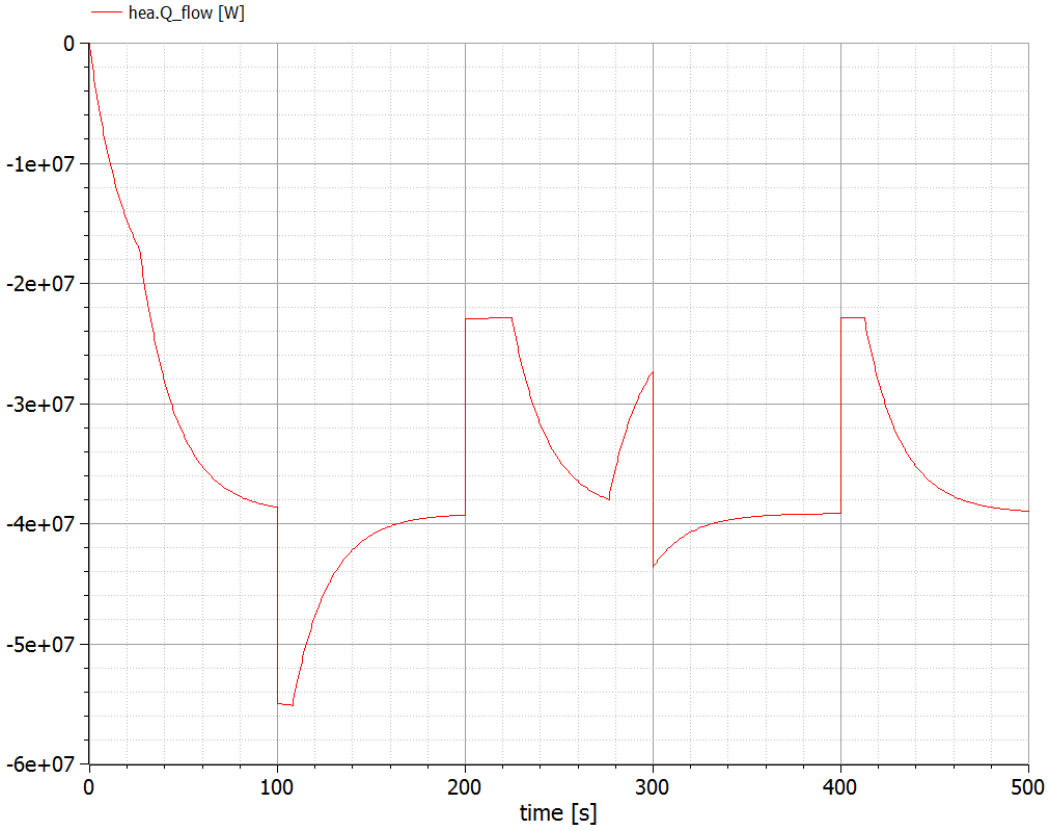


Figura 4.58: Respuesta de la potencia térmica del calentador-refrigerador (en este caso es negativo, se está refrigerando el fluido)

4.5.3. Ejemplo 3

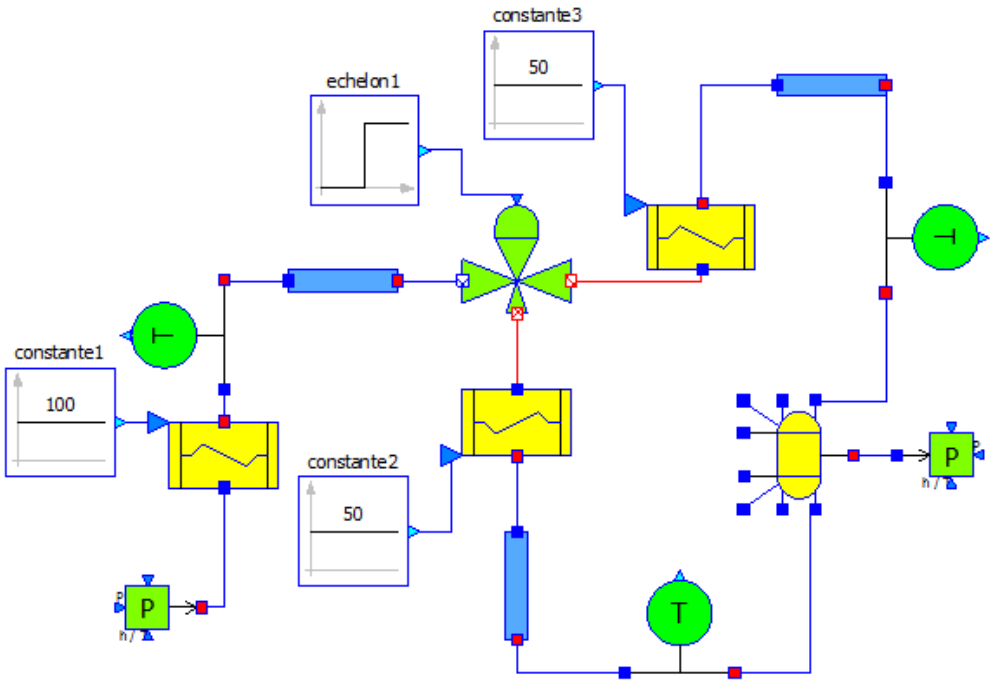


Figura 4.59: Diagram View del ejemplo 3

En este ejemplo aparecen los tres modelos alternativos que se han creado utilizando el Manual de Curvas de CALENER-GT.

Además, contiene una válvula de tres vías y un colector. La válvula está controlada por una señal en escalón de valor 1. El escalón comienza en el tiempo 250 s de la simulación. El funcionamiento de la válvula es el siguiente: Cuando la señal de control está en 1, se permite el flujo por el conector C1->C2. El conector C2 sería en este caso el contiguo al flujo de entrada. Cuando la señal de control está en cero, es decir, los 250 primeros segundos (ya que el escalón comienza en el 250), el flujo de agua pasa por el otro tramo de la válvula (C1->C3).

De izquierda a derecha, los equipos correspondientes son: bomba de calor (modo calor), máquina de absorción y bomba de calor (modo frío).

La temperatura de entrada al circuito es de unos 15ºC. La bomba de calor calentará el agua a unos 38ºC aproximadamente. Posteriormente, tras pasar, por una tubería, se adentrará en la válvula. Según las características de apertura de la válvula, ya descritas, el flujo pasará los 250 primeros segundos por el equipo máquina de absorción, y los otros restantes, por la bomba de calor (modo frío).

Como se observa en la imagen, las temperaturas de salida de los equipos es un valor de consigna que el modelador implementa. En este caso se ha utilizado dos bloques que proporcionan un valor constante.

Una consideración especialmente importante, como ya se dijo en la sección de los modelos alternativos, es que las temperaturas de consigna vienen expresadas en ºF, ya que las funciones que sirven para calcular las variables que caracterizan el comportamiento de los equipos utilizan las temperaturas de operación expresadas en ºF.

Tras pasar por los dos últimos equipos, ambos se encargan de enfriar el agua a 10ºC, los dos flujos se dirigen a un colector que hace de mezclador.

En las gráficas se observará que las temperaturas de salida de los dos equipos enfriadores describen una curva en forma de escalón, siguiendo la dinámica del caudal respectivo. Estos valores son realmente constantes aunque el software las presenta con una ligera variación en el orden de centésimas.

La respuesta de algunas variables bajo estas condiciones de operación son las siguientes:

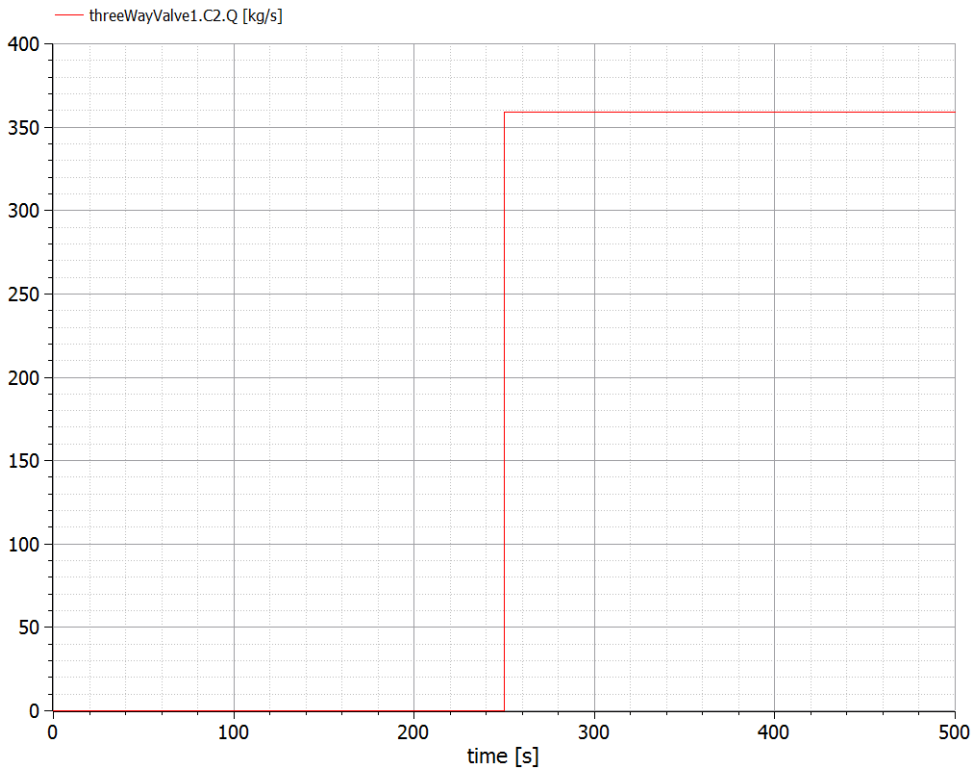


Figura 4.60: Caudal en la válvula de tres vías en el tramo C1->C2

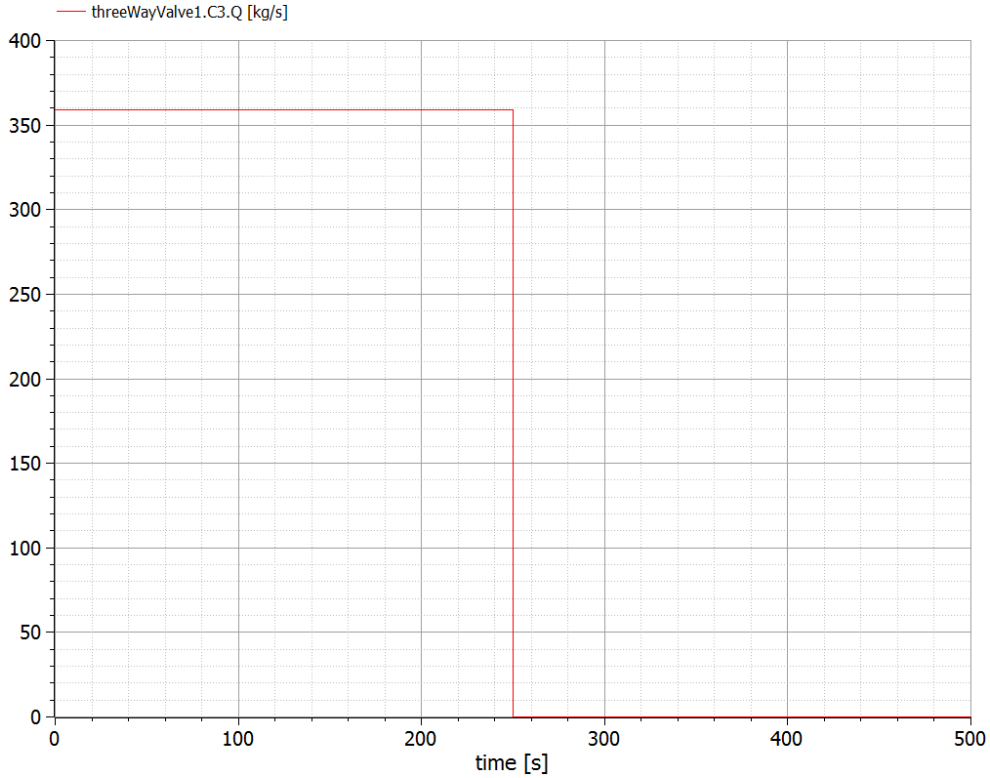


Figura 4.61: Caudal en la válvula de tres vías en el tramo C2->C3

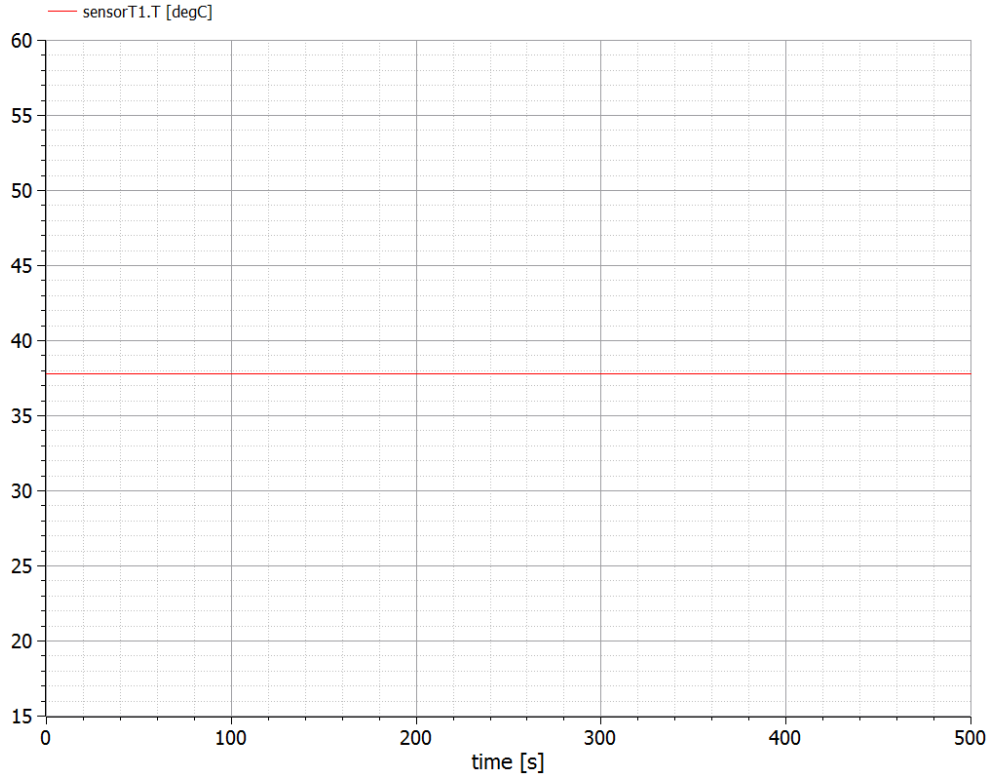


Figura 4.62: Temperatura de salida de la bomba de calor (calefacción)

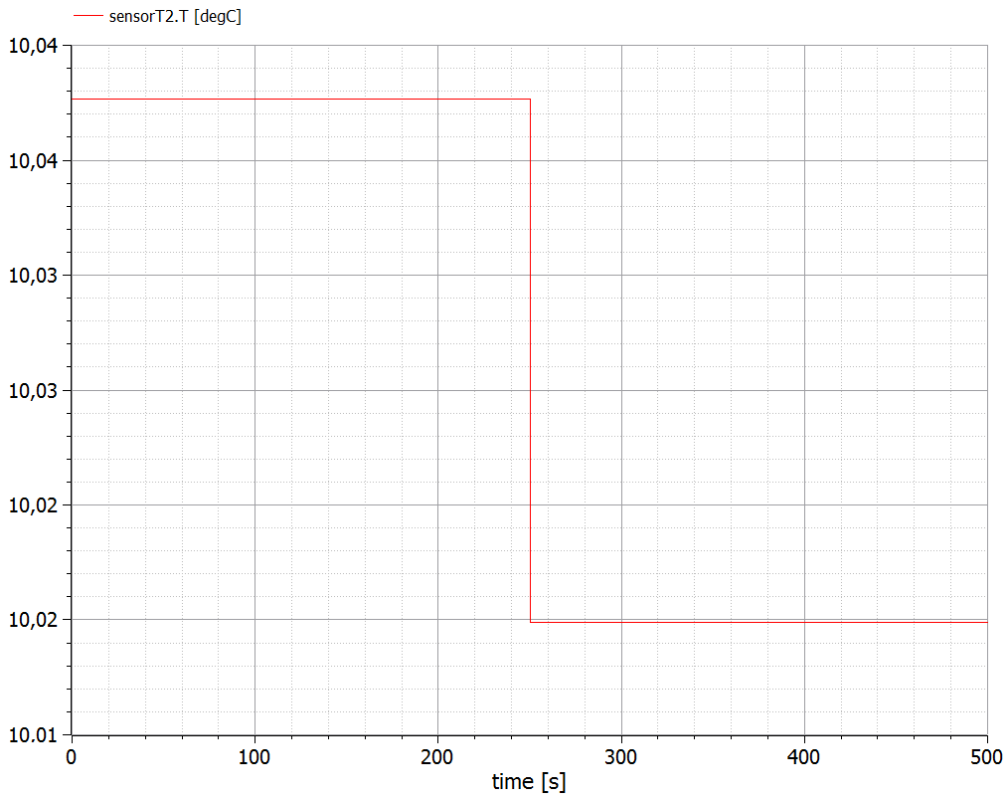


Figura 4.63: Temperatura de salida de la máquina de absorción

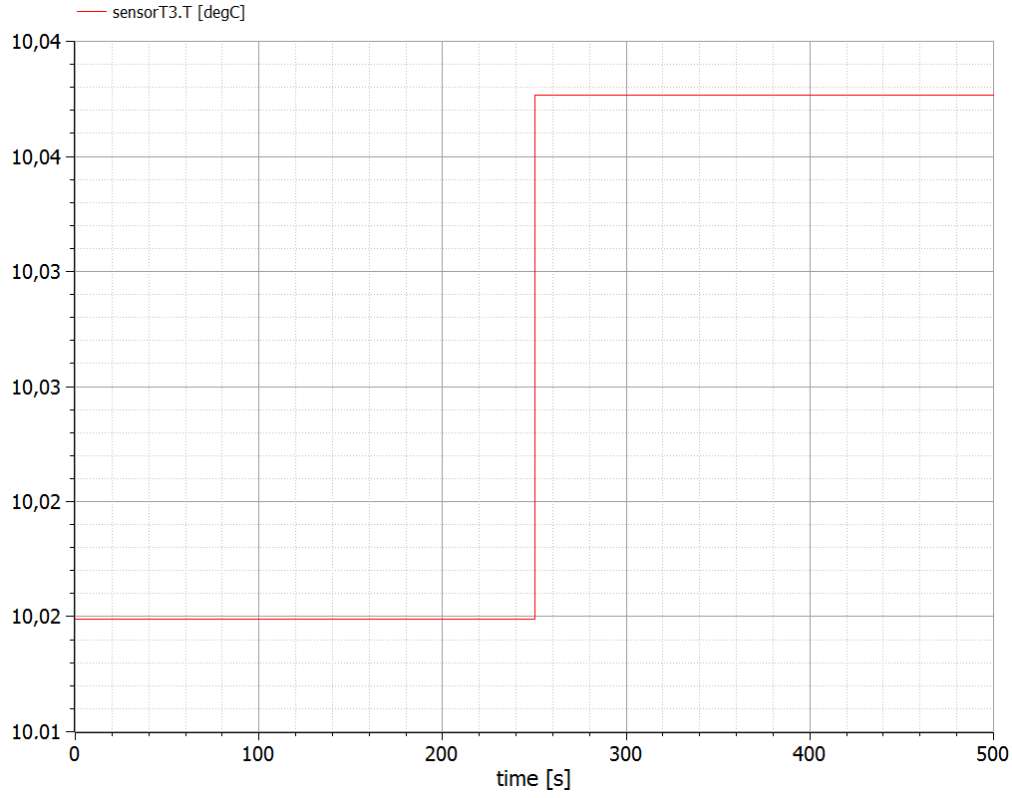


Figura 4.64: Temperatura d salida de la bomba de calor (refrigeración)

4.6. Desarrollo de la instalación

El modelo de la instalación completa de cogeneración en OpenModelica se encuentra en el Anexo F: Instalación de cogeneración en OpenModelica. Además se incluye el Text View de la planta. Algunas consideraciones finales:

Debido a la peculiaridad del modelo del motor a gas, y para simplificar un poco la instalación, el circuito de generación de alta temperatura solo está formado por un solo circuito. Consistiría en el circuito de agua que aprovecha la energía térmica del intercambiador de gases/agua. Por lo tanto, simularía el circuito de la caldera de recuperación. El circuito principal (el que sale del motor) correspondería al circuito de baja temperatura (cárter).

Otra simplificación no buscada, es debido al error del modelo de máquina de absorción de la librería ThermoSysPro. Este modelo estaba preparado para incorporar una torre de refrigeración. Como propuesta alternativa se ha creado un modelo utilizando el Manual de Curvas de CALENER-GT. Este modelo calculaba sus variables a través de funciones donde sus variables independientes eran las temperaturas de operación de salida del evaporador y entrada al condensador. Este modelo, en un principio, no está pensado para la incorporación de una torre de refrigeración, al solo tener el modelo una entrada y una salida.

La instalación montada al completo no se ha podido simular, ya que para el funcionamiento entero en la suite, se necesita de sistemas de control y de unas condiciones de operación determinadas para que los modelos no den errores.

5. Conclusiones y trabajos futuros

El objetivo de este trabajo era el de analizar una instalación de cogeneración en un software de modelado y simulación. El lenguaje Modelica es un lenguaje orientado a objetos propicio para esta tarea. Dentro de la gran variedad de entornos de modelado existentes en el mercado, se ha apostado por OpenModelica, una suite libre y de uso gratuito. Adentrarse en la suite puede ser difícil sino se está familiarizado con este tipo de lenguaje. Aunque no estaba previsto acceder al código de la suite (text view), las necesidades del trabajo han conducido a un estudio completo tanto del lenguaje Modelica como del entorno de modelado OpenModelica.

OpenModelica contiene gran variedad de librerías, permiten el modelado de gran variedad de sistemas (multidominio), ya sean electrónicos, mecánicos, termodinámicos, eléctricos, hidráulicos, etc. Para la construcción de nuestra instalación, se ha hecho necesario indagar por todas estas librerías en busca de componentes propicios para la instalación. Las librerías que han resultado ser útiles son principalmente MSL, Buildings y ThermoSysPro.

Antes de proceder a la construcción del modelo completo, es necesario conocer que componentes conforman una instalación de cogeneración, como se conectan y como es su funcionamiento. Una comprensión completa de cada modelo se hace necesaria, estudiando cuales son las ecuaciones que caracterizan su comportamiento, que variables definen el estado del componente y cuáles son las variables de flujo que permite comunicarse con otros componentes.

Como apoyo, para construir la instalación, se ha estudiado una planta de cogeneración real, situada en Montilla (Córdoba) de la empresa CIAT. CIAT ha proporcionado la información necesaria para conocer cuáles son los elementos que constituyen la planta. Es posible inferir el funcionamiento del modelo en régimen permanente a través del valor de algunas variables proporcionadas (en un único instante, o en varios, a través de tablas paramétricas).

Además, para una mejor visualización, se ha separado la instalación en partes. Esto permite un análisis apropiado y manejable. Se ha hecho uso del software TRNSYS, solo para presentar en el documento la separación de la planta y facilitar su descripción.

Una vez estudiada la instalación, se procede a su montaje en OpenModelica. Ha sido particularmente difícil encontrar y hacer funcionar todos los modelos requeridos. Este ha sido un punto de estancamiento claro en el desarrollo del TFG. Los errores de simulación permanentes, ha obligado a adentrarse en el código de cada modelo para conocer profundamente como se han construido éstos. Se han probado y estudiado muchos modelos. En la finalización de cada explicación se han evaluado para dictaminar si funcionan o no.

Toda la información obtenida con el continuo estudio de la suite, ha invitado a extender o modificar ligeramente los objetivos del TFG. Para ello se ha creado una sección donde se plasman algunas incidencias y se proponen algunas soluciones. Entre éstas, está la de construir modelos alternativos, precisamente para las bombas de calor y la máquina de absorción. Para ello, se ha hecho uso del Manual de Curvas de CALENER-GT. Estos modelos no cumplen del todo con las necesidades de la planta, teniendo que quitar en el montaje final, las torres de refrigeración que acompañaban a las máquinas de absorción. Vistos los problemas para hacer simular gran parte de modelos, se ha creado una sección en la que se presentan varios ejemplos de funcionamiento. En estos ejemplos se hacen uso de algunos bloques y señales de control que pueden ser de utilidad para un posterior control y optimización de la planta.

Finalmente se ha concluido con el montaje de la instalación. El modelo no se ha podido simular. Como propuesta inicial, estaba la de controlar la planta y experimentar con ella, haciendo simulaciones en régimen transitorio y permanente, con fines de optimizar el modelo para poder ser aplicado a la planta real.

Debido a la gran cantidad de incidencias imprevistas, se ha visto necesario limitar el alcance. Como líneas de trabajo futuro, está la de construir modelos alternativos más satisfactorios para sustituir a los modelos defectuosos presentes en las librerías. Ver cómo se comportan cuando se conectan entre ellos y si entregan una respuesta que permita validarlos y utilizarnos con fines de experimentación y optimización.

La principal propuesta, es la de continuar con esta instalación de cogeneración, y hacer conseguir que funcione correctamente al someterlo a simulaciones. Por último, está el propósito de controlar la instalación bajo una curva de potencia demandada, y conseguir que la instalación sea lo más eficiente posible, controlando los modos de funcionamiento, cuando se incorporan sistemas de apoyo (bombas) y cuando no, implementación de datos meteorológicos, etc. Todo ello con objetivo de adecuarse a condiciones de operación lo más realistas posibles y conseguir estudiar al modelo en régimen transitorio en un periodo de cálculo de un año.

Lista de referencias y bibliografía

Jose Mª Montserrat Muskiz 15/07/2011. Otros procesos sostenibles de generación de energía: Plantas de Cogeneración.

Durango, 17-19 Enero de 2006. Planes de mantenimiento de instalaciones. Giroa.

O.A. Jaramilla. Noviembre 20, 2007. Intercambiadores de calor. Centro de Investigación de Energía. Universidad Nacional Autónoma de México.

Ministerio de Industría, Turismo y Comercio. Calener-GT. Grandes edificios terciarios. Manual de Curvas. Calificación de Eficiencia Energética de Edificios.

CIATESA (27/11/2012). Datos Planta Cogeneración.

Grupo de Investigación Ingeniería Térmica-Universidad de Cádiz. PROYECTO CALORFRÍO.

Grupo de Investigación Ingeniería Térmica-Universidad de Cádiz. ESQUEMA DE PRINCIPIO DE LA PLANTA DE COGENERACIÓN.

Oscar G. Duarte V. Octubre 6 de 2014. UNVirtualLab. Un laboratorio Virtual basado en OpenModelica.

Baligh El Hefni, Daniel Bouskela, Gregory Lebreton. Dynamic modeling of a combined cycle power plant with ThermoSysPro. EDF R&D.

Rémi Allet. Development of models for designing industrial energy technologies related to cold production and store. Master's Thesis within the Sustainable Energy Systems programe. EDF R&D.

Peter Fritzson (2015). Introducción al Modelado y Simulación de Sistemas Técnicos y Físicos con Modelica. Derechos reservados de la edición en inglés: Introduction to Modeling and Simulation of Technical and Physical Systems with Modelica.

Modelica. Mayo 9, 2012. A Unified Object-Oriented language for systems Modeling. Language Specification. Version 3.3.

Berkeley Lab. Modelica Buildings Library.
http://simulationresearch.lbl.gov/modelica/releases/latest/help/Buildings.html#Buildings

EDF 2002-2014. ThermoSysPro.
https://build.openmodelica.org/Documentation/ThermoSysPro.html

Modelica. Documentation. https://build.openmodelica.org/Documentation/

Modelica. Modelica Libraries. https://www.modelica.org/libraries

RENOVETEC. Plantas de cogeneración. http://www.plantasdecogeneracion.com/index.php/las-plantas-de-cogeneracion

Mili-Viviani. Agosto 7, 2015. Coeficiente de flujo de válvulas de control. http://documents.tips/documents/coeficiente-de-flujo-de-valvulas-de-control.html

Dimplex. Detalles técnicos: Bomba de calor. http://www.dimplex.de/es/profesional/detalles-tecnicos/bombas-de-calor/el-funcionamiento-de-una-bomba-de-calor.html

Moris Arroes. Instalaciones y proyectos. Aerotermia. Aerotermo. http://www.morisarroes.es/aerotermia/aerotermo/

Idoia Arnabat CALORYFRIO. Miercoles, 26 Agosto 2015. Funcionamiento de la bomba de calor aire agua. Calor y frio.com. https://www.caloryfrio.com/calefaccion/bomba-de-calor/bombas-de-calor-reversibles-aire-agua-sistemas-integrales.html

Dosinda González-Mendizabal. Sartenejas, marzo de 2002. Guía de Intercambiadores de calor: Tipos Generales y Aplicaciones. http://www.academia.edu/7767151/GUIA_DE_INTERCAMBIADORES_DE_CALOR_TIPOS_GENERALES_Y_APLICACIONES

Anna Bonsfills Pedrós. Toni Dorado Castaño. Conxita Lao Luque. Xavier Gamisans Noguera. Montserrat Solé Sardans. Intercambiadores de calor. Departamento de Ingeniería Minera y Recursos Naturales. Universidad Politécnica de Cataluña. http://www.epsem.upc.edu/intercanviadorsdecalor/castella/intercanviadors_calor.html

Icogen. La refrigeración en ciclo de absorción. http://icogen-sa.com/refrigeraci%C3%B3n-t%C3%A9rmica-separador/la-refrigeraci%C3%B3n-en-ciclo-de-absorci%C3%B3n.html

Absorsistem. Funcionamiento del ciclo de absorción de simple efecto con bromuro de litio y agua, alimentado por agua caliente. http://www.absorsistem.com/tecnologia/absorcion/funcionamiento-del-ciclo-de-absorcion-de-simple-efecto-con-bromuro-de-litio-y-agua

Idoia Arrabat CALORYFRÍO. Jueves, 27 Agosto 2015. Bomba de calor agua agua. https://www.caloryfrio.com/calefaccion/bomba-de-calor/bomba-de-calor-agua-agua.html

Plan eficiencia energética. Motores Alternativos de Combustión Interna (MACI) de gas. Coselleria de Comerç, Indústria i Energia. Govern de les Illes Balears. http://www.caib.es/conselleries/industria/dgener/user/portalenergia/pla_eficiencia_energetica/produccioenergia_2.es.html

Listado de siglas, abreviaturas y acrónimos

TRNSYS: Transient Systems Simulation Program

TFG: Tranajo Fin de Grado

MSL: Modelica Standard Library. Librería estándar de Modelica

OM: OpenModelica

PCI: Poder calorífico inferior

PCS: Poder calorífico superior

GEI: Gas de efecto invernadero

ACS: Agua caliente sanitaria

OOP: Object-oriented programming. Programación orientada a objetos

OOM: Object-oriented modeling. Modelado orientado a objetos

HTML: HyperText Markup language. Lenguaje de marcas de hipertexto

OSMC: Open Source Modelica Consortium

MDT: The Modelica Developmetn Tooling. Herramienta de desarrollo Modelica

ISO: International Organization for Standardization. Organización Internacional de Normalización

I/O: Imput/Output. Entrada/Salida (E/S)

EDF: Électricité de France

SI: Sistema Internacional de unidades

I+D: Investigación y Desarrollo

MACI: Motores alternativos de combustión interna

PHE: Plate Heat Exchanger. Intercambiador de calor a placas

SHE: Spiral Heat Exchanger. Intercambiador de calor en espiral

COP: Coefficient of performance. Coeficiente de operación o rendimiento (calefacción)

EER: Energy Efficiency Ratio. Eficiencia energética (refrigeración)

SP: Solution Pump. Bomba de solución

MA: Máquina de absorción

TR: Torre de refrigeración

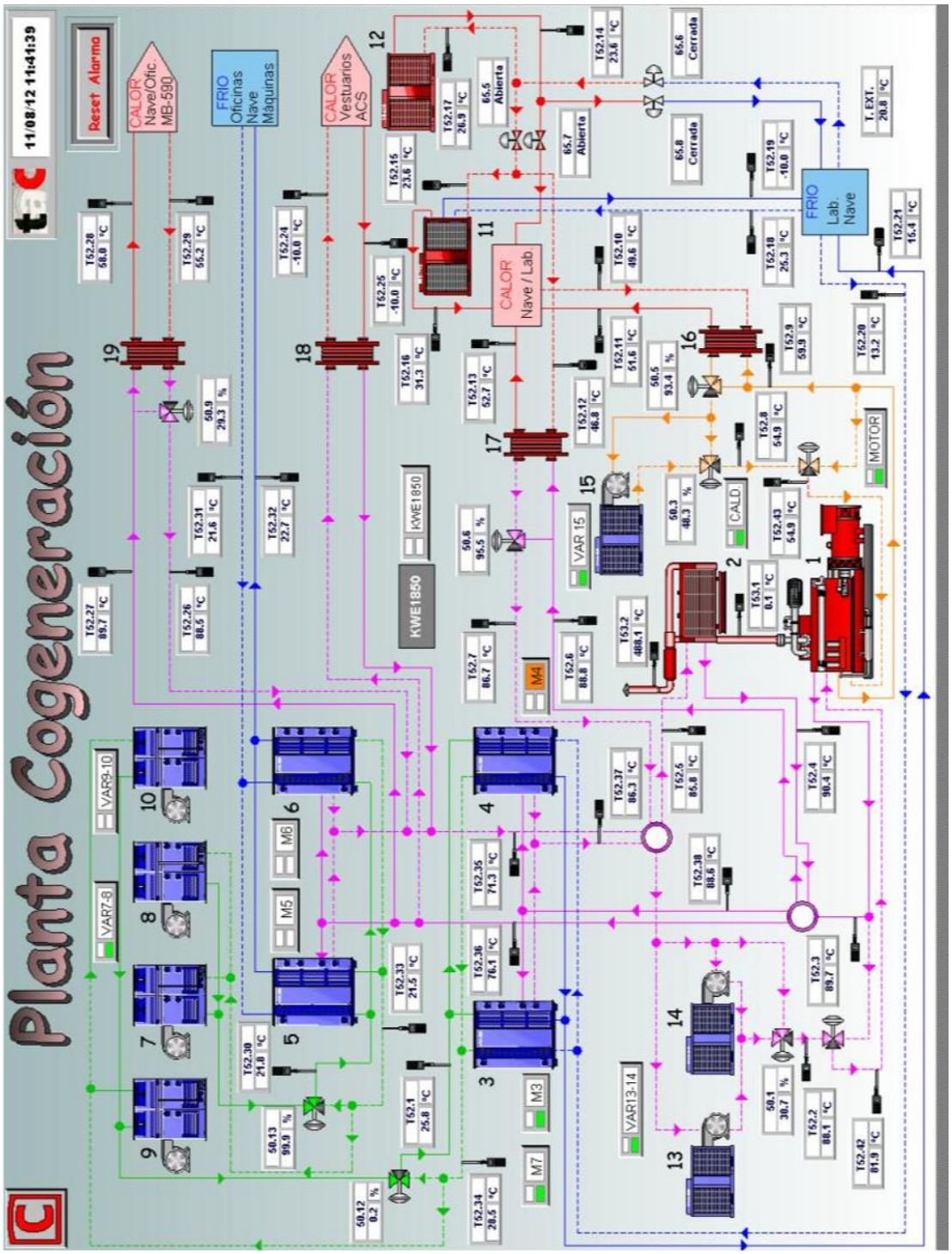
IC: Intercambiador de calor

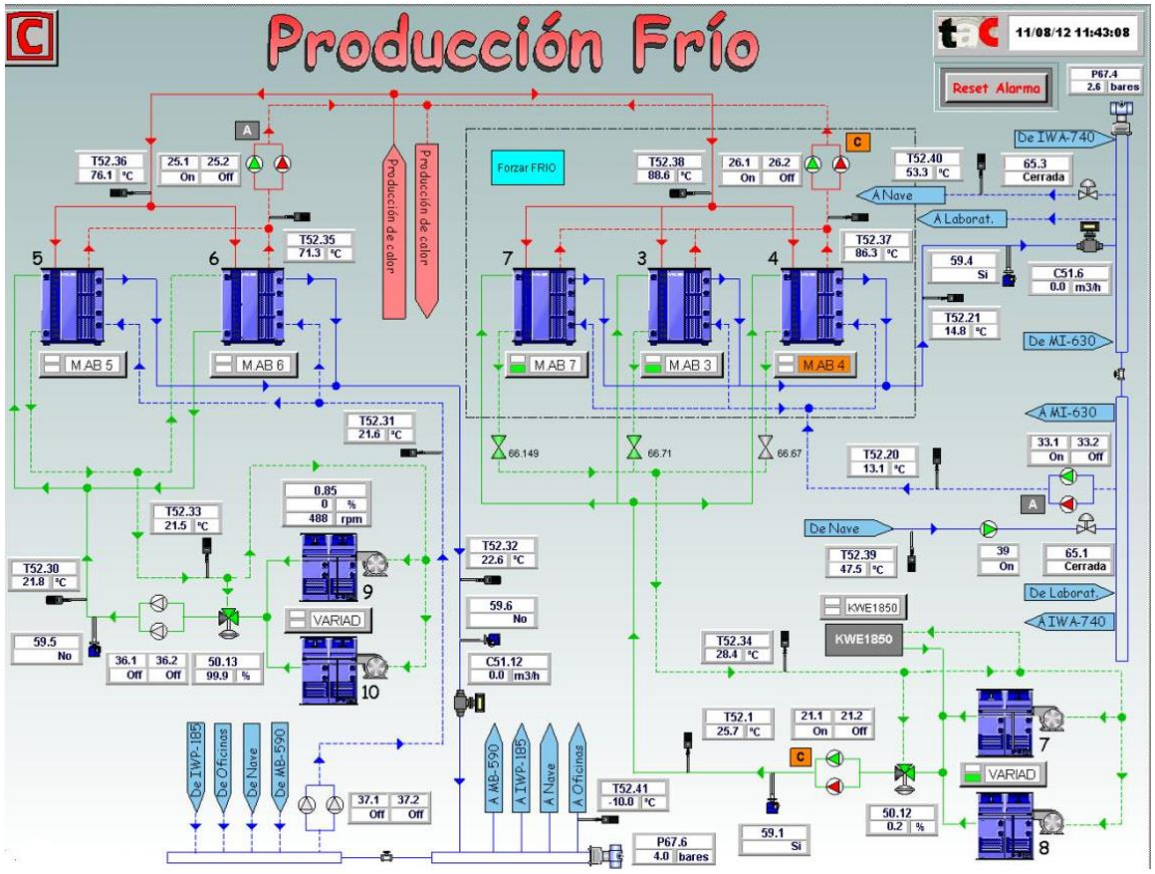
Dis: Disipador

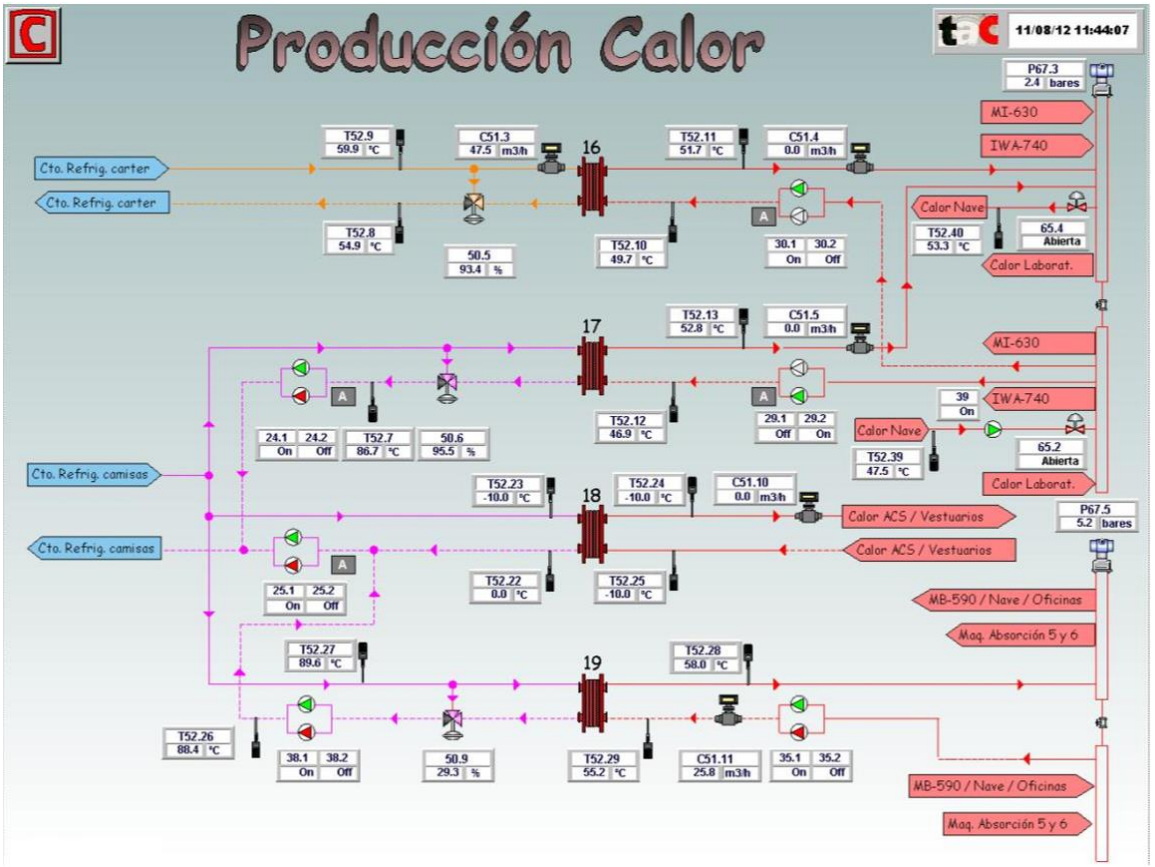
Cto: Circuito

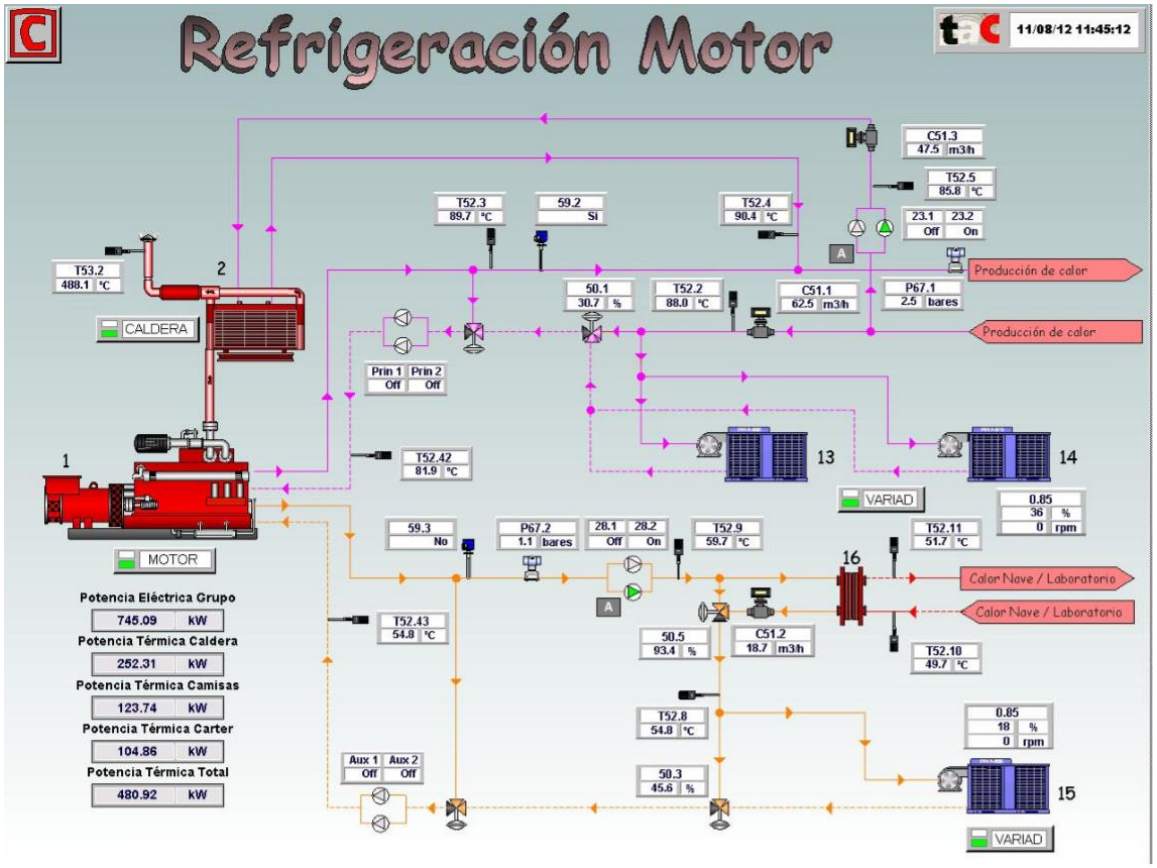
Anexo A: Datos Planta Cogeneración

Aquí se presentan la documentación gráfica aportada por CIAT de la instalación de cogeneración receptora. Los datos relevantes aparecen explícitamente en las imágenes.




Producción Frío
11/08/12 11:43:08
Reset Alarma
P67.4
2.6 bares
Forzar FRIO
Producción de calor
Producción de calor
T52.36
76.1 °C
25.1 25.2
On Off
T52.35
71.3 °C
T52.38
88.6 °C
26.1 26.2
On Off
T52.40
53.3 °C
65.3
Cerrada
A Nave
A Laborat.
T52.37
86.3 °C
59.4
Si
C51.6
0.0 m3/h
T52.21
14.8 °C
De IWA-740
De MI-630
A MI-630
5
6
7
3
4
MAB 5
MAB 6
MAB 7
MAB 3
MAB 4
T52.31
21.6 °C
66.149
66.71
66.67
T52.20
13.1 °C
33.1 33.2
On Off
0.85
0 %
488 rpm
T52.33
21.5 °C
T52.32
22.6 °C
De Nave
T52.39
47.5 °C
39
On
65.1
Cerrada
De Laborat.
A IWA-740
T52.30
21.8 °C
59.6
No
KWE1850
KWE1850
59.5
No
36.1 36.2
Off Off
50.13
99.9 %
C51.12
0.0 m3/h
T52.34
28.4 °C
9
VARIAD
10
T52.1
25.7 °C
21.1 21.2
On Off
7
VARIAD
8
De IWP-185
De Oficinas
De Nave
De MB-590
37.1 37.2
Off Off
A MB-590
A IWP-185
A Nave
A Oficinas
T52.41
-10.0 °C
50.12
0.2 %
P67.6
4.0 bares
59.1
Si


Producción Calor
11/08/12 11:44:07
P67.3
2.4 bares
MI-630
IWA-740
Cto. Refrig. carter
Cto. Refrig. carter
T52.9
59.9 °C
C51.3
47.5 m3/h
16
T52.11
51.7 °C
C51.4
0.0 m3/h
Calor Nave
T52.40
53.3 °C
65.4
Abierta
Calor Laborat.
T52.8
54.9 °C
50.5
93.4 %
T52.10
49.7 °C
30.1 30.2
On Off
T52.13
52.8 °C
C51.5
0.0 m3/h
17
MI-630
39
On
IWA-740
T52.12
46.9 °C
29.1 29.2
Off On
Calor Nave
T52.39
47.5 °C
65.2
Abierta
Calor Laborat.
24.1 24.2
On Off
T52.7
86.7 °C
50.6
95.5 %
Cto. Refrig. camisas
T52.23
-10.0 °C
18
T52.24
-10.0 °C
C51.10
0.0 m3/h
Calor ACS / Vestuarios
P67.5
5.2 bares
Cto. Refrig. camisas
Calor ACS / Vestuarios
25.1 25.2
On Off
T52.22
0.0 °C
T52.25
-10.0 °C
MB-590 / Nave / Oficinas
Maq. Absorción 5 y 6
T52.27
89.6 °C
19
T52.28
58.0 °C
T52.26
88.4 °C
38.1 38.2
On Off
50.9
29.3 %
T52.29
55.2 °C
C51.11
25.8 m3/h
35.1 35.2
On Off
MB-590 / Nave / Oficinas
Maq. Absorción 5 y 6



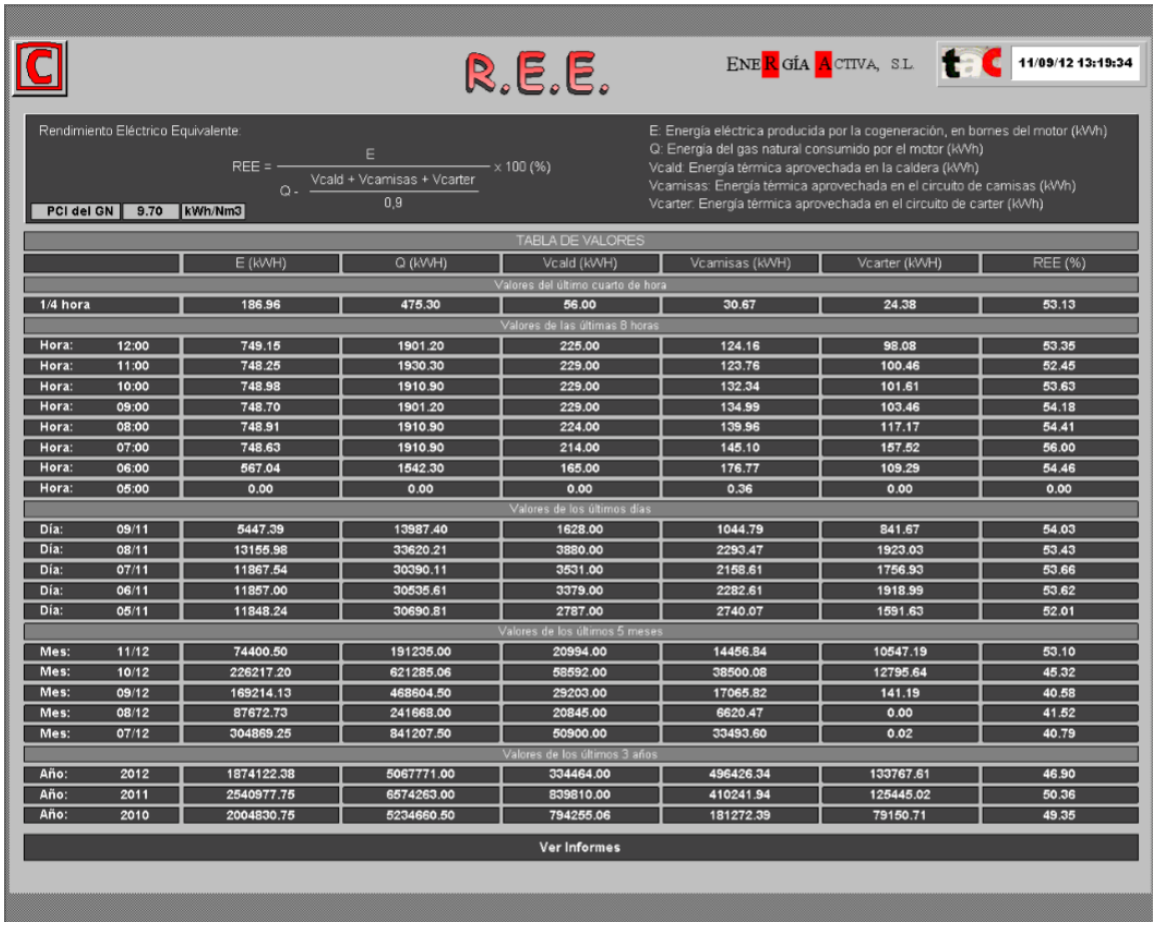
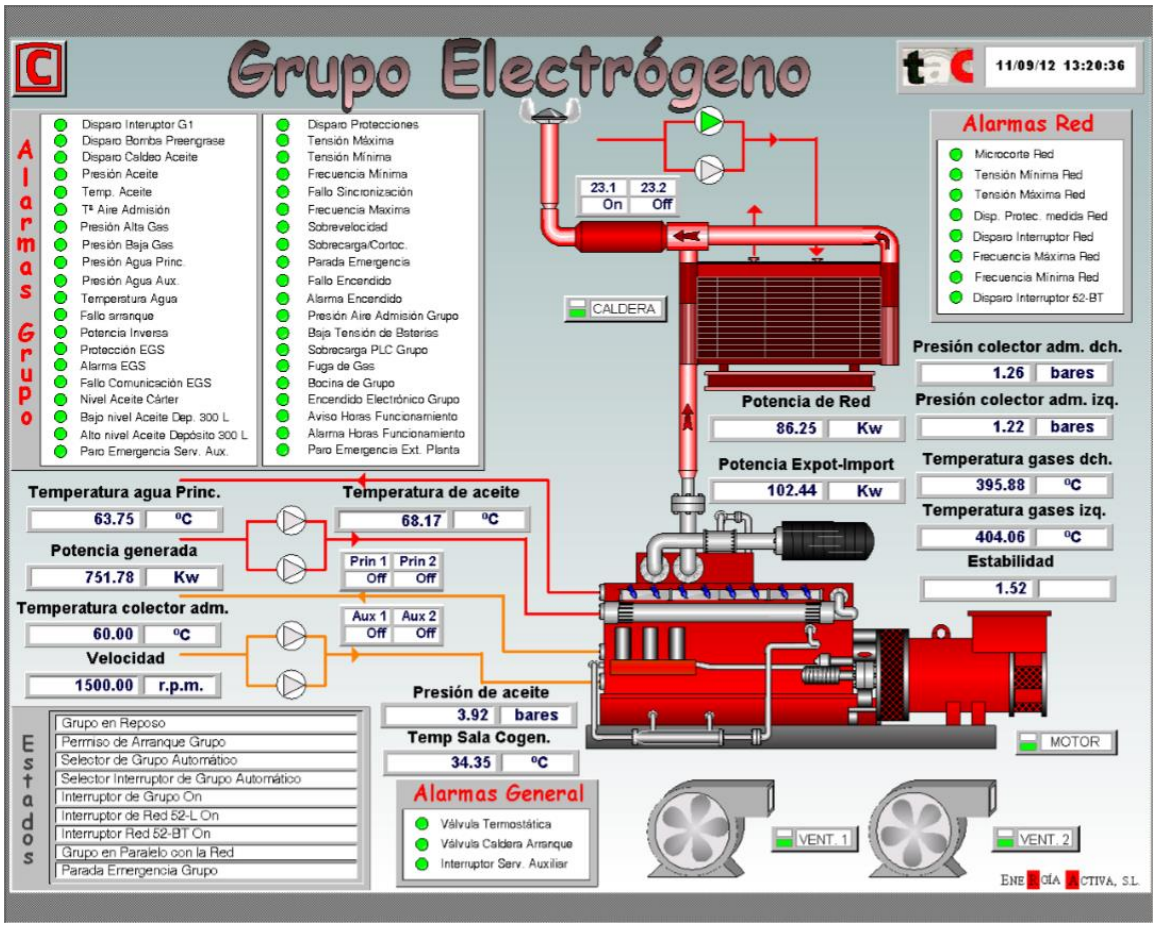
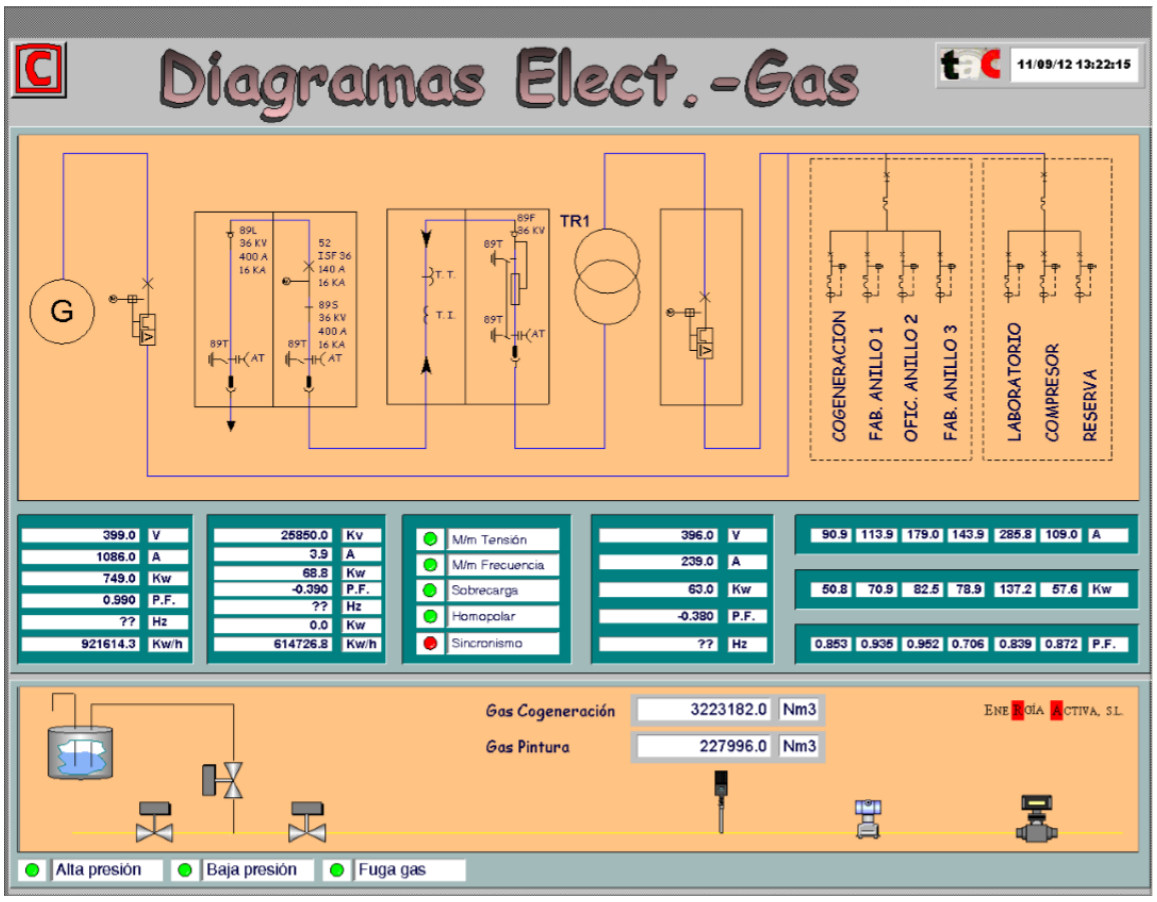


TABLA DE VALORES

		E (kWH)	Q (kWH)	Vcald (kWH)	Vcamisas (kWH)	Vcarter (kWH)	REE (%)
Valores del último cuarto de hora							
1/4 hora		186.96	475.30	56.00	30.67	24.38	53.13
Valores de las últimas 8 horas							
Hora:	12:00	749.15	1901.20	225.00	124.16	98.08	53.35
Hora:	11:00	748.25	1930.30	229.00	123.76	100.46	52.45
Hora:	10:00	748.98	1910.90	229.00	132.34	101.61	53.63
Hora:	09:00	748.70	1901.20	229.00	134.99	103.46	54.18
Hora:	08:00	748.91	1910.90	224.00	139.96	117.17	54.41
Hora:	07:00	748.63	1910.90	214.00	145.10	157.52	56.00
Hora:	06:00	567.04	1542.30	165.00	176.77	109.29	54.46
Hora:	05:00	0.00	0.00	0.00	0.36	0.00	0.00
Valores de los últimos días							
Día:	09/11	5447.39	13987.40	1628.00	1044.79	841.67	54.03
Día:	08/11	13155.98	33620.21	3880.00	2293.47	1923.03	53.43
Día:	07/11	11867.54	30390.11	3531.00	2158.61	1756.93	53.66
Día:	06/11	11857.00	30535.61	3379.00	2282.61	1918.99	53.62
Día:	05/11	11848.24	30690.81	2787.00	2740.07	1591.63	52.01
Valores de los últimos 5 meses							
Mes:	11/12	74400.50	191235.00	20994.00	14456.84	10547.19	53.10
Mes:	10/12	226217.20	621285.06	58592.00	38500.08	12795.64	45.32
Mes:	09/12	169214.13	468604.50	29203.00	17065.82	141.19	40.58
Mes:	08/12	87672.73	241668.00	20845.00	6620.47	0.00	41.52
Mes:	07/12	304869.25	841207.50	50900.00	33493.60	0.02	40.79
Valores de los últimos 3 años							
Año:	2012	1874122.38	5067771.00	334464.00	496426.34	133767.61	46.90
Año:	2011	2540977.75	6574263.00	839810.00	410241.94	125445.02	50.36
Año:	2010	2004830.75	5234660.50	794255.06	181272.39	79150.71	49.35

Ver Informes


Grupo Electrógeno
11/09/12 13:20:36
Alarmas Grupo
Disparo Interuptor G1
Disparo Bomba Preengrase
Disparo Caldeo Aceite
Presión Aceite
Temp. Aceite
Tª Aire Admisión
Presión Alta Gas
Presión Baja Gas
Presión Agua Princ.
Presión Agua Aux.
Temperatura Agua
Fallo arranque
Potencia Inversa
Protección EGS
Alarma EGS
Fallo Comunicación EGS
Nivel Aceite Cárter
Bajo nivel Aceite Dep. 300 L
Alto nivel Aceite Depósito 300 L
Paro Emergencia Serv. Aux.
Disparo Protecciones
Tensión Máxima
Tensión Mínima
Frecuencia Mínima
Fallo Sincronización
Frecuencia Maxima
Sobrevelocidad
Sobrecarga/Cortoc.
Parada Emergencia
Fallo Encendido
Alarma Encendido
Presión Aire Admisión Grupo
Baja Tensión de Baterías
Sobrecarga PLC Grupo
Fuga de Gas
Bocina de Grupo
Encendido Electrónico Grupo
Aviso Horas Funcionamiento
Alarma Horas Funcionamiento
Paro Emergencia Ext. Planta
Alarmas Red
Microcorte Red
Tensión Mínima Red
Tensión Máxima Red
Disp. Protec. medida Red
Disparo Interruptor Red
Frecuencia Máxima Red
Frecuencia Mínima Red
Disparo Interruptor 52-BT
23.1 On
23.2 Off
CALDERA
Presión colector adm. dch.
1.26 bares
Presión colector adm. izq.
1.22 bares
Potencia de Red
86.25 Kw
Potencia Expot-Import
102.44 Kw
Temperatura gases dch.
395.88 ºC
Temperatura gases izq.
404.06 ºC
Estabilidad
1.52
Temperatura agua Princ.
63.75 ºC
Temperatura de aceite
68.17 ºC
Potencia generada
751.78 Kw
Prin 1 Off
Prin 2 Off
Temperatura colector adm.
60.00 ºC
Aux 1 Off
Aux 2 Off
Velocidad
1500.00 r.p.m.
Presión de aceite
3.92 bares
Temp Sala Cogen.
34.35 ºC
MOTOR
Estados
Grupo en Reposo
Permiso de Arranque Grupo
Selector de Grupo Automático
Selector Interruptor de Grupo Automático
Interruptor de Grupo On
Interruptor de Red 52-L On
Interruptor Red 52-BT On
Grupo en Paralelo con la Red
Parada Emergencia Grupo
Alarmas General
Válvula Termostática
Válvula Caldera Arranque
Interruptor Serv. Auxiliar
VENT. 1
VENT. 2
ENERGÍA ACTIVA, S.L.


Diagramas Elect.-Gas
11/09/12 13:22:15
G
89L 36 KV 400 A 16 KA
52 I.SF 36 140 A 16 KA
89S 36 KV 400 A 16 KA
89T
AT
T. T.
T. I.
89F 36 KV
TR1
COGENERACION
FAB. ANILLO 1
OFIC. ANILLO 2
FAB. ANILLO 3
LABORATORIO
COMPRESOR
RESERVA
399.0 V
1086.0 A
749.0 Kw
0.990 P.F.
?? Hz
921614.3 Kw/h
25850.0 Kv
3.9 A
68.8 Kw
-0.390 P.F.
?? Hz
0.0 Kw
614726.8 Kw/h
M/m Tensión
M/m Frecuencia
Sobrecarga
Homopolar
Sincronismo
396.0 V
239.0 A
63.0 Kw
-0.380 P.F.
?? Hz
90.9 113.9 179.0 143.9 285.8 109.0 A
50.8 70.9 82.5 78.9 137.2 57.6 Kw
0.853 0.935 0.952 0.706 0.839 0.872 P.F.
Gas Cogeneración 3223182.0 Nm3
Gas Pintura 227996.0 Nm3
ENERGÍA ACTIVA, S.L.
Alta presión
Baja presión
Fuga gas

GUASCOR	CAPITULO 00	GENERALIDADES	INFORMACION DE PRODUCTO G-00-278	FECHA feb-03 INDICE B

PRESTACIONES / BALANCE TERMICO ENGINE PERFORMANCE DATA / HEAT BALANCES

MOTOR/ ENGINE **SFGLD 480** **1500 rpm**

MOTOR/ ENGINE TYPE:	SFGLD 480		ENCENDIDO/ IGNITION SYSTEM:	ALTRONIC
REGIMEN/ SPEED:	1500 rpm		REGULACION / REGULATION	Electrónica / Electronic
COMPRESION/ COMPRESSION RATIO:	11,8:1		AVANCE ENCENDIDO/ IGNITION TIMING (6)	17° BTDC
COLECTOR ESCAPE/ EXHAUST MANIFOLD TYPE:	WATER COOLED		COMBUSTIBLE/ FUEL TYPE:	NATURAL GAS
REFRIGERACION/ COOLING SYSTEM:	TWO CIRCUIT		NUMERO DE METANO MINIMO/ MINIMUM M.N. (6)	77
C. PRINCIPAL T AGUA SALIDA/ JACKET WATER TEMPERATURE (°C):	90		MAX.CONTRAPRESION/ MAX. BACK PRESSURE:	450 mm w.c.
C. AUXILIAR T AGUA ENTRADA/ INTERCOOLER WATER TEMP. (°C):	55		CONDICIONES AMBIENTALES/ AMBIENT CONDITIONS ISO 3046/1:	
CAUDAL MINIMO C. P./ MINIMUM WATER JACKET FLOW (m^3/h):	60		Atmospheric pressure (kPa)=	100
CAUDAL MINIMO C. A./ MIN. WATER FLOW 2ND CIRCUIT (m^3/h):	20		Ambient temperature (°C)=	25
PERDIDA DE CARGA C.P./ PRESSURE DROP MAIN CIRCUIT (bar): (8)	0,4	(8)	Relative humidity (%)=	30
PERDIDA DE CARGA C.A./ PRESSURE DROP 2ND CIRCUIT (bar): (8)	0,72	(8)	EMISIONES/ EMISSIONS:	500 mg/nm3 NOx

BALANCE TERMICO/ THERMAL BALANCE (4)			NOMINAL	CARGAS PARCIALES/ PART LOADS		
CARGA/ LOAD		%	100%	80%	60%	40%
POTENCIA MECANICA / MECHANICAL POWER	(3, 4, 5)	kWb	838	670	503	335
PME/ BMEP		bar	14,0	11,2	8,4	5,6
CONSUMO/ FUEL CONSUMPTION	(1)	kW	2160	1770	1380	995
RENDIMIENTO TERMICO/ THERMAL EFFICIENCY		%	38,8	37,9	36,4	33,7
POTENCIA C. PRINCIPAL AGUA/ HEAT IN MAIN WATER CIRCUIT	(1)	kW	585	500	410	320
POTENCIA C. AUXILIAR AGUA/ HEAT IN SECONDARY WATER CIRCUIT	(1)	kW	204	149	103	66
POTENCIA INTERCOOLER/ HEAT IN CHARGE COOLER	(1)	kW	127	76	35	3
POTENCIA INTERCAMBIADOR ACEITE/ HEAT IN OIL COOLER	(1)	kW	77	73	68	63
POTENCIA GASES DE ESCAPE/ HEAT IN EXHAUST GASES (25 °C)	(1)	kW	505	425	342	255
POTENCIA GASES DE ESCAPE/ HEAT IN EXHAUST GASES (120°C)	(1)	kW	366	312	254	191
TEMPERATURA GASES ESCAPE/ EXHAUST GAS TEMPERATURE	(1)	°C	370	382	393	403
PERDIDAS/ HEAT TO RADIATION	(1)	kW	28	26	22	19
AJUSTE CARBURACION/ CARBURATION SETTINGS (2)						
O_2 SECO EN ESCAPE/ O_2 TO EXHAUST (DRY)		%	9,01	8,88	8,73	8,44
LAMBDA			1,67	1,64	1,63	1,61
CAUDALES MASICOS/ MASS FLOWS						
CAUDAL AIRE ADMISION/ INTAKE AIR FLOW	(1)	kg/h	4424	3582	2786	1980
CAUDAL GASES DE ESCAPE (HUMEDOS)/ EXHAUST GAS FLOW (WET)	(1)	kg/h	4591	3719	2893	2057

Anexo B: Text View Colectores modificados

El código modificado correspondiente a:

Path: ThermoSysPro.WaterSteam.Junctions.Mixer8

para que su funcionamiento sea como separador de flujo (splitter) es el siguiente:

model Colectorret:

```
model Colectorret "Splitter with eight outlets"
  parameter Integer fluid = 1 "1: water/steam - 2: C3H3F5";
  parameter Integer mode = 0 "IF97 region. 1:liquid - 2:steam -
4:saturation line - 0:automatic";
public
  Modelica.SIunits.AbsolutePressure P(start = 10e5) "Fluid
pressure";
  Modelica.SIunits.SpecificEnthalpy h(start = 10e5) "Fluid
specific enthalpy";
  Modelica.SIunits.Temperature T "Fluid temperature";
  ThermoSysPro.WaterSteam.Connectors.FluidOutlet Cs5 annotation(
    layer = "icon",
    Placement(transformation(extent = {{-110, -50}, {-90, -30}},
rotation = 0)));
  ThermoSysPro.WaterSteam.Connectors.FluidOutlet Cs6 annotation(
    layer = "icon",
    Placement(transformation(extent = {{-112, -109}, {-92, -
89}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Connectors.FluidOutlet Cs7 annotation(
    layer = "icon",
    Placement(transformation(extent = {{-40, -109}, {-20, -89}},
rotation = 0)));
  ThermoSysPro.WaterSteam.Connectors.FluidOutlet Cs3 annotation(
    layer = "icon",
    Placement(transformation(extent = {{-112, 90}, {-92, 110}},
rotation = 0)));
  ThermoSysPro.WaterSteam.Connectors.FluidOutlet Cs2 annotation(
    layer = "icon",
    Placement(transformation(extent = {{-40, 90}, {-20, 110}},
rotation = 0)));
  ThermoSysPro.WaterSteam.Connectors.FluidOutlet Cs1 annotation(
    layer = "icon",
    Placement(transformation(extent = {{20, 92}, {40, 112}},
rotation = 0)));
  ThermoSysPro.WaterSteam.Connectors.FluidInlet Ce annotation(
    layer = "icon",
    Placement(transformation(extent = {{90, -10}, {110, 10}},
rotation = 0)));
  ThermoSysPro.WaterSteam.Connectors.FluidOutlet Cs8 annotation(
    layer = "icon",
    Placement(transformation(extent = {{20, -109}, {40, -89}},
rotation = 0)));
```

```
  ThermoSysPro.WaterSteam.Connectors.FluidOutlet Cs4 annotation(
    layer = "icon",
    Placement(transformation(extent = {{-112, 30}, {-92, 50}},
rotation = 0)));
  ThermoSysPro.Properties.WaterSteam.Common.ThermoProperties_ph
pro "Propriétés de l'eau" annotation(
    Placement(transformation(extent = {{-80, 80}, {-60, 100}},
rotation = 0)));
equation
/* Unconnected connectors */
  if cardinality(Cs1) == 0 then
    Cs1.Q = 0;
    Cs1.h = 1.e5;
    Cs1.a = true;
  end if;
  if cardinality(Cs2) == 0 then
    Cs2.Q = 0;
    Cs2.h = 1.e5;
    Cs2.a = true;
  end if;
  if cardinality(Cs3) == 0 then
    Cs3.Q = 0;
    Cs3.h = 1.e5;
    Cs3.a = true;
  end if;
  if cardinality(Cs4) == 0 then
    Cs4.Q = 0;
    Cs4.h = 1.e5;
    Cs4.a = true;
  end if;
  if cardinality(Cs5) == 0 then
    Cs5.Q = 0;
    Cs5.h = 1.e5;
    Cs5.a = true;
  end if;
  if cardinality(Cs6) == 0 then
    Cs6.Q = 0;
    Cs6.h = 1.e5;
    Cs6.a = true;
  end if;
  if cardinality(Cs7) == 0 then
    Cs7.Q = 0;
    Cs7.h = 1.e5;
    Cs7.a = true;
  end if;
  if cardinality(Cs8) == 0 then
    Cs8.Q = 0;
    Cs8.h = 1.e5;
    Cs8.a = true;
  end if;
  if cardinality(Ce) == 0 then
    Ce.Q = 0;
    Ce.h = 1.e5;
    Ce.b = true;
  end if;
/* Fluid pressure */
```

```
  P = Cs1.P;
  P = Cs2.P;
  P = Cs3.P;
  P = Cs4.P;
  P = Cs5.P;
  P = Cs6.P;
  P = Cs7.P;
  P = Cs8.P;
  P = Ce.P;
/* Fluid specific enthalpy (singular if all flows = 0) */
  Cs1.h_vol = h;
  Cs2.h_vol = h;
  Cs3.h_vol = h;
  Cs4.h_vol = h;
  Cs5.h_vol = h;
  Cs6.h_vol = h;
  Cs7.h_vol = h;
  Cs8.h_vol = h;
  Ce.h_vol = h;
/* Mass balance equation */
  0 = Cs1.Q - Cs2.Q - Cs3.Q - Cs4.Q - Cs5.Q - Cs6.Q - Cs7.Q -
Cs8.Q + Ce.Q;
/* Energy balance equation */
  0 = Cs1.Q * Cs1.h - Cs2.Q * Cs2.h - Cs3.Q * Cs3.h - Cs4.Q *
Cs4.h - Cs5.Q * Cs5.h - Cs6.Q * Cs6.h - Cs7.Q * Cs7.h - Cs8.Q *
Cs8.h + Ce.Q * Ce.h;
/* Fluid thermodynamic properties */
  pro = ThermoSysPro.Properties.Fluid.Ph(P, h, mode, fluid);
  T = pro.T;
  annotation(
    Diagram(coordinateSystem(preserveAspectRatio = false, extent
= {{-100, -100}, {100, 100}}, grid = {2, 2}), graphics =
{Ellipse(extent = {{-40, 80}, {40, 0}}, lineColor = {0, 0, 255},
fillColor = {255, 255, 0}, fillPattern = FillPattern.Solid),
Ellipse(extent = {{-40, 2}, {40, -80}}, lineColor = {0, 0, 255},
fillColor = {255, 255, 0}, fillPattern = FillPattern.Solid),
Rectangle(extent = {{-40, 40}, {40, -40}}, lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid), Line(points = {{40, 0}, {92, 0}}),
Line(points = {{-92, -40}, {-40, -40}}), Line(points = {{-30,
90}, {-30, 66}}, color = {0, 0, 255}), Line(points = {{30, 92},
{30, 66}}, color = {0, 0, 255}), Line(points = {{-30, -66}, {-
30, -90}}, color = {0, 0, 255}), Line(points = {{30, -66}, {30,
-90}}, color = {0, 0, 255}), Line(points = {{-92, 40}, {-40,
40}}), Line(points = {{-38, -52}, {-92, -90}}, color = {0, 0,
255}), Line(points = {{-38, 54}, {-92, 90}}, color = {0, 0,
255})}),
    Icon(coordinateSystem(preserveAspectRatio = false, extent =
{{-100, -100}, {100, 100}}, grid = {2, 2}), graphics =
{Ellipse(extent = {{-40, 80}, {40, 0}}, lineColor = {0, 0, 255},
fillColor = {255, 255, 0}, fillPattern = FillPattern.Solid),
Ellipse(extent = {{-40, 0}, {40, -80}}, lineColor = {0, 0, 255},
fillColor = {255, 255, 0}, fillPattern = FillPattern.Solid),
Rectangle(extent = {{-40, 40}, {40, -40}}, lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid), Line(points = {{-30, 90}, {-30, 66}}, color
```

```
= {0, 0, 255}), Line(points = {{-30, -66}, {-30, -90}}, color =
{0, 0, 255}), Line(points = {{30, 92}, {30, 66}}, color = {0, 0,
255}), Line(points = {{30, -66}, {30, -90}}, color = {0, 0,
255}), Line(points = {{-92, 40}, {-40, 40}}), Line(points = {{-
92, -40}, {-40, -40}}), Line(points = {{40, 0}, {92, 0}}),
Line(points = {{-38, -52}, {-92, -90}}, color = {0, 0, 255}),
Line(points = {{-38, 54}, {-92, 90}}, color = {0, 0, 255})}),
    Window(x = 0.05, y = 0.07, width = 0.74, height = 0.85),
    Documentation(info = "<html>
<p><b>Copyright &copy; EDF 2002 - 2010</b></p>
</HTML>
<html>
<p><b>ThermoSysPro Version 2.0</b></p>
</HTML>
    ", revisions = "<html>
<u><p><b>Authors</u> : </p></b>
<ul style='margin-top:0cm' type=disc>
<li>
    Baligh El Hefni</li>
<li>
    Daniel Bouskela</li>
</ul>
</html>
    "));

    end Colectorret;
```

El código del siguiente modelo también ha sido modificado por si se quieren utilizar más entradas y salidas de flujo. En este caso he añadido dos entradas/salidas más a las del modelo por defecto, teniendo un total de cinco. Para la construcción de este modelo es necesario modificar unos modelos además que se encuentran dentro de este. Esta es una de las peculiaridades del lenguaje modélica: herencia y jerarquía de clases. Para extender una clase se emplea la palabra extends. El modelo final es union_separador. Sin cargar union_separador_2 ni union_separador_1, no funciona el modelo union_separador.

Path: Buildings.Fluid.FixedResistances.SplitterFixedResistanceDpM

model union_separador_2:

```
record union_separador_2
  replaceable package Medium =
Modelica.Media.Interfaces.PartialMedium "Medium in the
component" annotation(
    choicesAllMatching = true);
  // Assumptions
  parameter Modelica.Fluid.Types.Dynamics energyDynamics =
Modelica.Fluid.Types.Dynamics.DynamicFreeInitial "Type of energy
balance: dynamic (3 initialization options) or steady state"
annotation(
    Evaluate = true,
    Dialog(tab = "Dynamics", group = "Equations"));
```

```
  parameter Modelica.Fluid.Types.Dynamics massDynamics =
energyDynamics "Type of mass balance: dynamic (3 initialization
options) or steady state" annotation(
    Evaluate = true,
    Dialog(tab = "Dynamics", group = "Equations"));
  final parameter Modelica.Fluid.Types.Dynamics
substanceDynamics = energyDynamics "Type of independent mass
fraction balance: dynamic (3 initialization options) or steady
state" annotation(
    Evaluate = true,
    Dialog(tab = "Dynamics", group = "Equations"));
  final parameter Modelica.Fluid.Types.Dynamics traceDynamics =
energyDynamics "Type of trace substance balance: dynamic (3
initialization options) or steady state" annotation(
    Evaluate = true,
    Dialog(tab = "Dynamics", group = "Equations"));
  // Initialization
  parameter Medium.AbsolutePressure p_start = Medium.p_default
"Start value of pressure" annotation(
    Dialog(tab = "Initialization"));
  parameter Medium.Temperature T_start = Medium.T_default "Start
value of temperature" annotation(
    Dialog(tab = "Initialization"));
  parameter Medium.MassFraction X_start[Medium.nX](quantity =
Medium.substanceNames) = Medium.X_default "Start value of mass
fractions m_i/m" annotation(
    Dialog(tab = "Initialization", enable = Medium.nXi > 0));
  parameter Medium.ExtraProperty C_start[Medium.nC](quantity =
Medium.extraPropertiesNames) = fill(0, Medium.nC) "Start value
of trace substances" annotation(
    Dialog(tab = "Initialization", enable = Medium.nC > 0));
  parameter Medium.ExtraProperty C_nominal[Medium.nC](quantity =
Medium.extraPropertiesNames) = fill(1E-2, Medium.nC) "Nominal
value of trace substances. (Set to typical order of magnitude.)"
annotation(
    Dialog(tab = "Initialization", enable = Medium.nC > 0));
  parameter Real mSenFac(min = 1) = 1 "Factor for scaling the
sensible thermal mass of the volume" annotation(
    Dialog(tab = "Dynamics"));
  annotation(
    preferredView = "info",
    Documentation(info = "<html>
<p>
This class contains parameters and medium properties
that are used in the lumped  volume model, and in models that
extend the
lumped volume model.
</p>
<p>
These parameters are used for example by
<a
href=\"modelica://Buildings.Fluid.Interfaces.ConservationEquatio
n\">
Buildings.Fluid.Interfaces.ConservationEquation</a>,
<a
href=\"modelica://Buildings.Fluid.MixingVolumes.MixingVolume\">
```

```
Buildings.Fluid.MixingVolumes.MixingVolume</a> and
<a
href=\"modelica://Buildings.Fluid.HeatExchangers.Radiators.Radia
torEN442_2\">
Buildings.Fluid.HeatExchangers.Radiators.RadiatorEN442_2</a>.
</p>
</html>", revisions = "<html>
<ul>
<li>
April 11, 2016 by Michael Wetter:<br/>
Corrected wrong hyperlink in documentation for
<a href=\"https://github.com/iea-annex60/modelica-
annex60/issues/450\">issue 450</a>.
</li>
<li>
January 26, 2016, by Michael Wetter:<br/>
Added <code>quantity=Medium.substanceNames</code> for
<code>X_start</code>.
</li>
<li>
October 21, 2014, by Filip Jorissen:<br/>
Added parameter <code>mFactor</code> to increase the thermal
capacity.
</li>
<li>
August 2, 2011, by Michael Wetter:<br/>
Set <code>substanceDynamics</code> and
<code>traceDynamics</code> to final
and equal to <code>energyDynamics</code>,
as there is no need to make them different from
<code>energyDynamics</code>.
</li>
<li>
August 1, 2011, by Michael Wetter:<br/>
Changed default value for <code>energyDynamics</code> to
<code>Modelica.Fluid.Types.Dynamics.DynamicFreeInitial</code>
because
<code>Modelica.Fluid.Types.Dynamics.SteadyStateInitial</code>
leads
to high order DAE that Dymola cannot reduce.
</li>
<li>
July 31, 2011, by Michael Wetter:<br/>
Changed default value for <code>energyDynamics</code> to
<code>Modelica.Fluid.Types.Dynamics.SteadyStateInitial</code>.
</li>
<li>
April 13, 2009, by Michael Wetter:<br/>
First implementation.
</li>
</ul>
</html>"));

    end union_separador_2;
```

model union_separador_1:

```
partial model union_separador_1 "Flow splitter with partial
resistance model at each port"
  extends union_separador_2(final mSenFac = 1);
  Modelica.Fluid.Interfaces.FluidPort_a port_1(redeclare package
Medium = Medium, m_flow(min = if portFlowDirection_1 ==
Modelica.Fluid.Types.PortFlowDirection.Entering then 0.0 else -
Modelica.Constants.inf, max = if portFlowDirection_1 ==
Modelica.Fluid.Types.PortFlowDirection.Leaving then 0.0 else
Modelica.Constants.inf)) "First port, typically inlet"
annotation(
    Placement(transformation(extent = {{-110, -10}, {-90,
10}})));
  Modelica.Fluid.Interfaces.FluidPort_b port_2(redeclare package
Medium = Medium, m_flow(min = if portFlowDirection_2 ==
Modelica.Fluid.Types.PortFlowDirection.Entering then 0.0 else -
Modelica.Constants.inf, max = if portFlowDirection_2 ==
Modelica.Fluid.Types.PortFlowDirection.Leaving then 0.0 else
Modelica.Constants.inf)) "Second port, typically outlet"
annotation(
    Placement(transformation(extent = {{90, -10}, {110, 10}})));
  Modelica.Fluid.Interfaces.FluidPort_a port_3(redeclare package
Medium = Medium, m_flow(min = if portFlowDirection_3 ==
Modelica.Fluid.Types.PortFlowDirection.Entering then 0.0 else -
Modelica.Constants.inf, max = if portFlowDirection_3 ==
Modelica.Fluid.Types.PortFlowDirection.Leaving then 0.0 else
Modelica.Constants.inf)) "Third port, can be either inlet or
outlet" annotation(
    Placement(visible = true, transformation(extent = {{-48, -
110}, {-28, -90}}, rotation = 0), iconTransformation(extent =
{{-50, -110}, {-30, -90}}, rotation = 0)));
  Modelica.Fluid.Interfaces.FluidPort_a port_4(redeclare package
Medium = Medium, m_flow(min = if portFlowDirection_4 ==
Modelica.Fluid.Types.PortFlowDirection.Entering then 0.0 else -
Modelica.Constants.inf, max = if portFlowDirection_4 ==
Modelica.Fluid.Types.PortFlowDirection.Leaving then 0.0 else
Modelica.Constants.inf)) "Fourth port, can be either inlet or
outlet" annotation(
    Placement(visible = true, transformation(extent = {{28, -
110}, {48, -90}}, rotation = 0), iconTransformation(extent =
{{30, -110}, {50, -90}}, rotation = 0)));
  Modelica.Fluid.Interfaces.FluidPort_a port_5(redeclare package
Medium = Medium, m_flow(min = if portFlowDirection_5 ==
Modelica.Fluid.Types.PortFlowDirection.Entering then 0.0 else -
Modelica.Constants.inf, max = if portFlowDirection_5 ==
Modelica.Fluid.Types.PortFlowDirection.Leaving then 0.0 else
Modelica.Constants.inf)) "Fifth port, can be either inlet or
outlet" annotation(
    Placement(transformation(extent = {{-10, -110}, {10, -
90}})));
  parameter Modelica.SIunits.Time tau = 10 "Time constant at
nominal flow for dynamic energy and momentum balance"
annotation(
```

```
    Dialog(tab = "Dynamics", group = "Nominal condition", enable
= not energyDynamics ==
Modelica.Fluid.Types.Dynamics.SteadyState));
  parameter Modelica.SIunits.MassFlowRate mDyn_flow_nominal
"Nominal mass flow rate for dynamic momentum and energy balance"
annotation(
    Dialog(tab = "Dynamics", group = "Equations", enable = not
energyDynamics == Modelica.Fluid.Types.Dynamics.SteadyState));
  parameter Boolean from_dp = true "= true, use m_flow = f(dp)
else dp = f(m_flow)" annotation(
    Evaluate = true,
    Dialog(tab = "Advanced"));
  parameter Modelica.Fluid.Types.PortFlowDirection
portFlowDirection_1 =
Modelica.Fluid.Types.PortFlowDirection.Bidirectional "Flow
direction for port_1" annotation(
    Dialog(tab = "Advanced"));
  parameter Modelica.Fluid.Types.PortFlowDirection
portFlowDirection_2 =
Modelica.Fluid.Types.PortFlowDirection.Bidirectional "Flow
direction for port_2" annotation(
    Dialog(tab = "Advanced"));
  parameter Modelica.Fluid.Types.PortFlowDirection
portFlowDirection_3 =
Modelica.Fluid.Types.PortFlowDirection.Bidirectional "Flow
direction for port_3" annotation(
    Dialog(tab = "Advanced"));
  parameter Modelica.Fluid.Types.PortFlowDirection
portFlowDirection_4 =
Modelica.Fluid.Types.PortFlowDirection.Bidirectional "Flow
direction for port_4" annotation(
    Dialog(tab = "Advanced"));
  parameter Modelica.Fluid.Types.PortFlowDirection
portFlowDirection_5 =
Modelica.Fluid.Types.PortFlowDirection.Bidirectional "Flow
direction for port_5" annotation(
    Dialog(tab = "Advanced"));
  replaceable Buildings.Fluid.Interfaces.PartialTwoPortInterface
res1 constrainedby
Buildings.Fluid.Interfaces.PartialTwoPortInterface(redeclare
final package Medium = Medium, allowFlowReversal =
portFlowDirection_1 ==
Modelica.Fluid.Types.PortFlowDirection.Bidirectional) "Partial
model, to be replaced with a fluid component" annotation(
     Placement(transformation(extent = {{-60, -10}, {-40,
10}})));
  replaceable Buildings.Fluid.Interfaces.PartialTwoPortInterface
res2 constrainedby
Buildings.Fluid.Interfaces.PartialTwoPortInterface(redeclare
final package Medium = Medium, allowFlowReversal =
portFlowDirection_2 ==
Modelica.Fluid.Types.PortFlowDirection.Bidirectional) "Partial
model, to be replaced with a fluid component" annotation(
     Placement(transformation(extent = {{60, -10}, {40, 10}})));
  replaceable Buildings.Fluid.Interfaces.PartialTwoPortInterface
res3 annotation(
```

```
    Placement(visible = true, transformation(origin = {-38, -
50}, extent = {{-10, 10}, {10, -10}}, rotation = 90)))
constrainedby
Buildings.Fluid.Interfaces.PartialTwoPortInterface(redeclare
final package Medium = Medium, allowFlowReversal =
portFlowDirection_3 ==
Modelica.Fluid.Types.PortFlowDirection.Bidirectional) "Partial
model, to be replaced with a fluid component" annotation(
     Placement(transformation(origin = {0, -50}, extent = {{-10,
10}, {10, -10}}, rotation = 90)));
  replaceable Buildings.Fluid.Interfaces.PartialTwoPortInterface
res4 annotation(
    Placement(visible = true, transformation(origin = {38, -50},
extent = {{-10, 10}, {10, -10}}, rotation = 90))) constrainedby
Buildings.Fluid.Interfaces.PartialTwoPortInterface(redeclare
final package Medium = Medium, allowFlowReversal =
portFlowDirection_4 ==
Modelica.Fluid.Types.PortFlowDirection.Bidirectional) "Partial
model, to be replaced with a fluid component" annotation(
     Placement(transformation(origin = {0, -50}, extent = {{-10,
10}, {10, -10}}, rotation = 90)));
  replaceable Buildings.Fluid.Interfaces.PartialTwoPortInterface
res5 constrainedby
Buildings.Fluid.Interfaces.PartialTwoPortInterface(redeclare
final package Medium = Medium, allowFlowReversal =
portFlowDirection_5 ==
Modelica.Fluid.Types.PortFlowDirection.Bidirectional) "Partial
model, to be replaced with a fluid component" annotation(
     Placement(transformation(origin = {0, -50}, extent = {{-10,
10}, {10, -10}}, rotation = 90)));
  Buildings.Fluid.Delays.DelayFirstOrder vol(redeclare final
package Medium = Medium, final nPorts = 5, final tau = tau,
final m_flow_nominal = mDyn_flow_nominal, final energyDynamics =
energyDynamics, final massDynamics = massDynamics, final p_start
= p_start, final T_start = T_start, final X_start = X_start,
final C_start = C_start, final allowFlowReversal = true, final
prescribedHeatFlowRate = false) if have_controlVolume "Fluid
volume to break algebraic loop" annotation(
    Placement(transformation(extent = {{-10, 0}, {10, 20}})));
protected
  parameter Boolean have_controlVolume = energyDynamics <>
Modelica.Fluid.Types.Dynamics.SteadyState or massDynamics <>
Modelica.Fluid.Types.Dynamics.SteadyState "Boolean flag used to
remove conditional components";
  Modelica.Fluid.Interfaces.FluidPort_a port_internal(redeclare
package Medium = Medium) if not have_controlVolume "Internal
dummy port for easier connection of conditional connections"
annotation(
    Placement(transformation(extent = {{-10, 50}, {10, 70}})));
equation
  if portFlowDirection_1 ==
Modelica.Fluid.Types.PortFlowDirection.Leaving then
    if not have_controlVolume then
      connect(res1.port_a, port_internal) annotation(
        Line(points = {{-60, 0}, {-60, 60}, {0, 60}}, color =
{0, 127, 255}));
```

```
    else
      connect(res1.port_a, vol.ports[1]) annotation(
        Line(points = {{-60, 0}, {-2.66667, 0}}, color = {0,
127, 255}));
    end if;
    connect(port_1, res1.port_b) annotation(
      Line(points = {{-100, 0}, {-100, 0}, {-40, 0}}, color =
{0, 127, 255}));
  else
    if not have_controlVolume then
      connect(res1.port_b, port_internal) annotation(
        Line(points = {{-40, 0}, {-40, 60}, {0, 60}}, color =
{0, 127, 255}));
    else
      connect(res1.port_b, vol.ports[1]) annotation(
        Line(points = {{-40, 0}, {-2.66667, 0}}, color = {0,
127, 255}));
    end if;
    connect(port_1, res1.port_a) annotation(
      Line(points = {{-100, 0}, {-100, 0}, {-60, 0}}, color =
{0, 127, 255}));
  end if;
  if portFlowDirection_2 ==
Modelica.Fluid.Types.PortFlowDirection.Leaving then
    if not have_controlVolume then
      connect(res2.port_a, port_internal) annotation(
        Line(points = {{60, 0}, {60, 60}, {0, 60}}, color = {0,
127, 255}));
    else
      connect(res2.port_a, vol.ports[2]) annotation(
        Line(points = {{60, 0}, {2.22045e-16, 0}}, color = {0,
127, 255}));
    end if;
    connect(port_2, res2.port_b) annotation(
      Line(points = {{100, 0}, {100, 0}, {40, 0}}, color = {0,
127, 255}));
  else
    if not have_controlVolume then
      connect(res2.port_b, port_internal) annotation(
        Line(points = {{40, 0}, {40, 60}, {0, 60}}, color = {0,
127, 255}));
    else
      connect(res2.port_b, vol.ports[2]) annotation(
        Line(points = {{40, 0}, {2.22045e-16, 0}}, color = {0,
127, 255}));
    end if;
    connect(port_2, res2.port_a) annotation(
      Line(points = {{100, 0}, {100, 0}, {60, 0}}, color = {0,
127, 255}));
  end if;
  if portFlowDirection_3 ==
Modelica.Fluid.Types.PortFlowDirection.Leaving then
    if not have_controlVolume then
      connect(res3.port_a, port_internal) annotation(
        Line(points = {{-4.44089e-16, -60}, {20, -60}, {20, 60},
{0, 60}}, color = {0, 127, 255}));
```

```
    else
      connect(res3.port_a, vol.ports[3]) annotation(
        Line(points = {{-6.66134e-16, -60}, {0, -60}, {0, 0},
{2.66667, 0}}, color = {0, 127, 255}));
    end if;
    connect(port_3, res3.port_b) annotation(
      Line(points = {{0, -100}, {0, -100}, {0, -40}}, color =
{0, 127, 255}));
  else
    if not have_controlVolume then
      connect(res3.port_b, port_internal) annotation(
        Line(points = {{4.44089e-16, -40}, {20, -40}, {20, 60},
{0, 60}}, color = {0, 127, 255}));
    else
      connect(res3.port_b, vol.ports[3]) annotation(
        Line(points = {{4.44089e-16, -40}, {0, -40}, {0, 0},
{2.66667, 0}}, color = {0, 127, 255}));
    end if;
    connect(port_3, res3.port_a) annotation(
      Line(points = {{0, -100}, {0, -100}, {0, -60}}, color =
{0, 127, 255}));
  end if;
  if portFlowDirection_4 ==
Modelica.Fluid.Types.PortFlowDirection.Leaving then
    if not have_controlVolume then
      connect(res4.port_a, port_internal) annotation(
        Line(points = {{-4.44089e-16, -60}, {20, -60}, {20, 60},
{0, 60}}, color = {0, 127, 255}));
    else
      connect(res4.port_a, vol.ports[4]) annotation(
        Line(points = {{-6.66134e-16, -60}, {0, -60}, {0, 0},
{2.66667, 0}}, color = {0, 127, 255}));
    end if;
    connect(port_4, res4.port_b) annotation(
      Line(points = {{0, -100}, {0, -100}, {0, -40}}, color =
{0, 127, 255}));
  else
    if not have_controlVolume then
      connect(res4.port_b, port_internal) annotation(
        Line(points = {{4.44089e-16, -40}, {20, -40}, {20, 60},
{0, 60}}, color = {0, 127, 255}));
    else
      connect(res4.port_b, vol.ports[4]) annotation(
        Line(points = {{4.44089e-16, -40}, {0, -40}, {0, 0},
{2.66667, 0}}, color = {0, 127, 255}));
    end if;
    connect(port_4, res4.port_a) annotation(
      Line(points = {{0, -100}, {0, -100}, {0, -60}}, color =
{0, 127, 255}));
  end if;
  if portFlowDirection_5 ==
Modelica.Fluid.Types.PortFlowDirection.Leaving then
    if not have_controlVolume then
      connect(res5.port_a, port_internal) annotation(
        Line(points = {{-4.44089e-16, -60}, {20, -60}, {20, 60},
{0, 60}}, color = {0, 127, 255}));
```

```
    else
      connect(res5.port_a, vol.ports[5]) annotation(
        Line(points = {{-6.66134e-16, -60}, {0, -60}, {0, 0},
{2.66667, 0}}, color = {0, 127, 255}));
    end if;
    connect(port_5, res5.port_b) annotation(
      Line(points = {{0, -100}, {0, -100}, {0, -40}}, color =
{0, 127, 255}));
  else
    if not have_controlVolume then
      connect(res5.port_b, port_internal) annotation(
        Line(points = {{4.44089e-16, -40}, {20, -40}, {20, 60},
{0, 60}}, color = {0, 127, 255}));
    else
      connect(res5.port_b, vol.ports[5]) annotation(
        Line(points = {{4.44089e-16, -40}, {0, -40}, {0, 0},
{2.66667, 0}}, color = {0, 127, 255}));
    end if;
    connect(port_5, res5.port_a) annotation(
      Line(points = {{0, -100}, {0, -100}, {0, -60}}, color =
{0, 127, 255}));
  end if;
  annotation(
    Documentation(info = "<html>
<p>
Partial model for flow resistances with three ports such as a
flow mixer/splitter or a three way valve.
</p>
<p>
If <code>energyDynamics &ne;
Modelica.Fluid.Types.Dynamics.SteadyState</code>,
then at the junction of the three flows,
a mixing volume will be present. This will introduce a dynamic
energy and momentum
balance, which often breaks algebraic loops.
The time constant of the mixing volume is determined by the
parameter <code>tau</code>.
</p>
</html>", revisions = "<html>
<ul>
<li>
February 22, 2016, by Michael Wetter:<br/>
Conditionally removed control volume <code>vol</code>, and added
the conditional connnector
<code>port_internal</code>.
This was already done when the parameter
<code>dynamicBalance</code> was present, but
was updated wrong when this parameter was removed.
Without these conditional components, the regression test for
<code>Buildings.Fluid.Examples.ResistanceVolumeFlowReversal</cod
e> fails to simulate.
</li>
<li>
December 17, 2015, by Michael Wetter:<br/>
Added assignment <code>redeclare final package
Medium=Medium</code>
```

```
as this is required for OpenModelica.
This is for
<a href=\"https://github.com/lbl-srg/modelica-
buildings/issues/475\">
https://github.com/lbl-srg/modelica-buildings/issues/475</a>.
</li>
<li>February 20, 2016, by Ruben Baetens:<br/>
Removal of <code>dynamicBalance</code> as parameter for
<code>massDynamics</code> and <code>energyDynamics</code>.
</li>
<li>
April 13 2015, by Filip Jorissen:<br/>
Exposed options for flow reversal to users and added
corresponding implementation.
</li>
<li>
March 23 2010, by Michael Wetter:<br/>
Changed start values from <code>system.p_start</code> or (code
<code>T_start</code>)
to <code>Medium.p_default</code>.
</li>
<li>
September 18, 2008 by Michael Wetter:<br/>
Replaced splitter model with a fluid port since the
splitter model in Modelica.Fluid 1.0 beta does not transport
<code>mC_flow</code>.
<li>
June 11, 2008 by Michael Wetter:<br/>
First implementation.
</li>
</ul>
</html>"));
end union_separador_1;
```

model union_separador:

```
model union_separador
  "Flow splitter with fixed resistance at each port"
  extends union_separador_1(mDyn_flow_nominal =
sum(abs(m_flow_nominal[:])/5),
      redeclare
Buildings.Fluid.FixedResistances.FixedResistanceDpM res1(
            final allowFlowReversal=true,
            from_dp=from_dp,
            final m_flow_nominal=m_flow_nominal[1],
            final dp_nominal=dp_nominal[1],
            final ReC=ReC[1],
            final dh=dh[1],
            linearized=linearized,
            homotopyInitialization=homotopyInitialization,
            deltaM=deltaM),
      redeclare
Buildings.Fluid.FixedResistances.FixedResistanceDpM res2(
            final allowFlowReversal=true,
```

```
            from_dp=from_dp,
            final m_flow_nominal=m_flow_nominal[2],
            final dp_nominal=dp_nominal[2],
            final ReC=ReC[2],
            final dh=dh[2],
            linearized=linearized,
            homotopyInitialization=homotopyInitialization,
            deltaM=deltaM),
      redeclare
Buildings.Fluid.FixedResistances.FixedResistanceDpM res3(
            final allowFlowReversal=true,
            from_dp=from_dp,
            final m_flow_nominal=m_flow_nominal[3],
            final dp_nominal=dp_nominal[3],
            final ReC=ReC[3],
            final dh=dh[3],
            linearized=linearized,
            homotopyInitialization=homotopyInitialization,
            deltaM=deltaM),
      redeclare
Buildings.Fluid.FixedResistances.FixedResistanceDpM res4(
            final allowFlowReversal=true,
            from_dp=from_dp,
            final m_flow_nominal=m_flow_nominal[4],
            final dp_nominal=dp_nominal[4],
            final ReC=ReC[4],
            final dh=dh[4],
            linearized=linearized,
            homotopyInitialization=homotopyInitialization,
            deltaM=deltaM),
       redeclare
Buildings.Fluid.FixedResistances.FixedResistanceDpM res5(
            final allowFlowReversal=true,
            from_dp=from_dp,
            final m_flow_nominal=m_flow_nominal[5],
            final dp_nominal=dp_nominal[5],
            final ReC=ReC[5],
            final dh=dh[5],
            linearized=linearized,
            homotopyInitialization=homotopyInitialization,
            deltaM=deltaM));

  parameter Modelica.SIunits.MassFlowRate[5] m_flow_nominal
    "Mass flow rate. Set negative at outflowing ports."
annotation(Dialog(group = "Nominal condition"));

  parameter Modelica.SIunits.Pressure[5] dp_nominal(each
displayUnit = "Pa")
    "Pressure drop at nominal mass flow rate, set to zero or
negative number at outflowing ports."
    annotation(Dialog(group = "Nominal condition"));

  parameter Boolean use_dh = false
    "= true, use dh and ReC, otherwise use deltaM"
    annotation(Evaluate=true,
               Dialog(group = "Transition to laminar",
```

```
                        enable = not linearized));

  parameter Real deltaM(min=0) = 0.3
    "Fraction of nominal mass flow rate where transition to
turbulent occurs"
       annotation(Dialog(group = "Transition to laminar",
                         enable = not use_dh and not
linearized));

  parameter Modelica.SIunits.Length[5] dh={1, 1, 1, 1, 1}
"Hydraulic diameter"
    annotation(Dialog(group = "Transition to laminar",
                      enable = use_dh and not linearized));

  parameter Real[5] ReC(each min=0)={4000, 4000, 4000, 4000,
4000}
    "Reynolds number where transition to turbulent starts"
      annotation(Dialog(group = "Transition to laminar",
                        enable = use_dh and not linearized));
  parameter Boolean linearized = false
    "= true, use linear relation between m_flow and dp for any
flow rate"
    annotation(Dialog(tab="Advanced"));

  parameter Boolean homotopyInitialization = true "= true, use
homotopy method"
    annotation(Evaluate=true, Dialog(tab="Advanced"));

  annotation (Icon(coordinateSystem(initialScale = 0.1),
graphics={Polygon(fillColor = {175, 175, 175}, fillPattern =
FillPattern.Solid, points = {{-100, -46}, {-32, -40}, {-32, -
100}, {30, -100}, {30, -36}, {100, -30}, {100, 38}, {-100, 52},
{-100, -46}}), Polygon( fillColor = {0, 128, 255}, fillPattern =
FillPattern.HorizontalCylinder, points = {{-100, -34}, {-18, -
28}, {-18, -100}, {18, -100}, {18, -26}, {100, -20}, {100, 22},
{-100, 38}, {-100, -34}}), Ellipse(lineColor = {0, 0, 127},
fillColor = {0, 0, 127}, fillPattern = FillPattern.Solid, extent
= {{-38, 36}, {40, -40}}, endAngle = 360), Text(lineColor = {0,
0, 255}, extent = {{-151, 142}, {149, 102}}, textString =
"%name", fontName = "MS Shell Dlg 2")}),
defaultComponentName="spl",
    Documentation(info="<html>
<p>
Model of a flow splitter or mixer with a fixed resistance in
each flow leg.
In each flow lag, a pressure drop can be modeled, and at the
fluid junction,
a mixing volume can be modeled.
</p>
<p>
The pressure drop is implemented using the model
<a
href=\"modelica://Buildings.Fluid.FixedResistances.FixedResistan
ceDpM\">
Buildings.Fluid.FixedResistances.FixedResistanceDpM</a>.
```

```
If its nominal pressure drop is set to zero, then the pressure
drop
model will be removed.
For example, the pressure drop declaration
</p>
<pre>
  m_flow_nominal={ 0.1, 0.1,  -0.2},
  dp_nominal =   {500,    0, -6000}
</pre>
<p>
would model a mixer that has the nominal flow rates and
associated pressure drops
as shown in the figure below. Note that <code>port_3</code> is
set to negative values.
The negative values indicate that at the nominal conditions,
fluid is leaving the component.
</p>
<p align=\"center\">
<img alt=\"image\"
src=\"modelica://Buildings/Resources/Images/Fluid/FixedResistanc
es/SplitterFixedResistanceDpM.png\"/>
</p>
<p>
Optionally, at the fluid junction, a control volume can be
modeled.
This is implemented using the model
<a href=\"modelica://Buildings.Fluid.Delays.DelayFirstOrder\">
Buildings.Fluid.Delays.DelayFirstOrder</a>.
The fluid volume is modeled if
<code>energyDynamics &lt;&gt;
Modelica.Fluid.Types.Dynamics.SteadyState</code>.
The control volume has the size
</p>
<pre>
  V = sum(abs(m_flow_nominal[:])/3)*tau/rho_nominal
</pre>
<p>
where <code>tau</code> is a parameter and
<code>rho_nominal</code> is the density
of the medium in the volume at nominal condition.
Setting
<code>energyDynamics=Modelica.Fluid.Types.Dynamics.FixedInitial<
/code>
can help reducing the size of the nonlinear
system of equations.
</p>
</html>", revisions="<html>
<ul>
<li>
October 14, 2016 by Michael Wetter:<br/>
Added to Annex 60 library.<br/>
Updated comment for parameter <code>use_dh</code>.<br/>
This is for
<a href=\"https://github.com/iea-annex60/modelica-
annex60/issues/451\">issue 451</a>.
</li>
```

```
<li>
Removed parameter <code>dynamicBalance</code> that overwrote the
setting
of <code>energyDynamics</code> and <code>massDynamics</code>.
This is for
<a href=\"https://github.com/iea-annex60/modelica-
annex60/issues/411\">
Annex 60, issue 411</a>.
</li>
<li>
February 1, 2012 by Michael Wetter:<br/>
Expanded documentation.
</li>
<li>
August 4, 2011 by Michael Wetter:<br/>
Added <code>final allowFlowReversal=true</code> to all
resistances since it is impractical
to avoid flow reversal in large flow networks where such a
setting may be useful.
</li>
<li>
June 11, 2008 by Michael Wetter:<br/>
Based class on
<a
href=\"modelica://Buildings.Fluid.BaseClasses.PartialThreeWayFix
edResistance\">
PartialThreeWayFixedResistance</a>.
</li>
<li>
July 20, 2007 by Michael Wetter:<br/>
First implementation.
</li>
</ul>
</html>"));


    end union_separador;
```

Anexo C: Text View Bombas de calor y Máquina de absorción

En este anexo se presenta el código (Text View) de los modelos creados basados en curvas de comportamiento de la máquina de absorción y de las bombas de calor (sistema de apoyo).

model Maquina_absorcion:

```
model Maquina_absorcion
  parameter Real a = 0.723412;
  parameter Real b = 0.079006;
  parameter Real c = -0.000897;
  parameter Real d = -0.025285;
  parameter Real e = -0.000048;
  parameter Real f = 0.000276;
  parameter Real a1 = 0.652273;
  parameter Real b1 = 0.000000;
  parameter Real c1 = 0.000000;
  parameter Real d1 = -0.000545;
  parameter Real e1 = 0.000055;
  parameter Real f1 = 0.000000;
  parameter Real a2 = 0.087773;
  parameter Real b2 = 0.744921;
  parameter Real c2 = 0.167306;
  parameter kW POT_NOM;
  parameter Real EER_NOM;
  ThermoSysPro.WaterSteam.Connectors.FluidInlet EntTher
annotation(
    Placement(visible = true, transformation(extent = {{-8, -
68}, {12, -48}}, rotation = 0), iconTransformation(extent = {{-
8, -70}, {12, -50}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Connectors.FluidOutlet SalTher
annotation(
    Placement(visible = true, transformation(extent = {{-8, 52},
{12, 72}}, rotation = 0)));
  Modelica.SIunits.Conversions.NonSIunits.Temperature_degF
T_ent;
  Modelica.SIunits.Conversions.NonSIunits.Temperature_degF
T_sal;
protected
  Real POT_NOM_T;
  Real EER_TERM_T;
  Real EER_TERM_FCP;
  Real PLR;
public
  kW POT;
  Real EER;
```

```
  ThermoSysPro.InstrumentationAndControl.Connectors.InputReal
TEva annotation(
    Placement(visible = true, transformation(origin = {-120,
60}, extent = {{20, -20}, {-20, 20}}, rotation = 180),
iconTransformation(extent = {{20, -20}, {-20, 20}}, rotation =
180)));
  ThermoSysPro.Properties.WaterSteam.Common.ThermoProperties_ph
proce annotation(
    Placement(visible = true, transformation(origin = {-80, 84},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.Properties.WaterSteam.Common.ThermoProperties_pT
procs annotation(
    Placement(visible = true, transformation(origin = {-54, 84},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
equation
  TEva.signal = T_sal;
  EntTher.P = SalTher.P;
  0 = if EntTher.Q > 0 then EntTher.h - EntTher.h_vol else
SalTher.h - SalTher.h_vol;
  EntTher.Q = SalTher.Q;

    POT_NOM_T = a + b * T_sal + c * T_sal ^ 2 + d * T_ent + e *

    EER_TERM_T = a1 + b1 * T_sal + c1 * T_sal ^ 2 + d1 * T_ent +
e1 * T_ent ^ 2 + f1 * T_sal * T_ent;

  PLR = 1;
  EER_TERM_FCP = a2 + b2 * PLR + c2 * PLR ^ 2;
  proce =
ThermoSysPro.Properties.WaterSteam.IF97.Water_Ph(EntTher.P,
EntTher.h, 0);
  procs =
ThermoSysPro.Properties.WaterSteam.IF97.Water_PT(SalTher.P,
Modelica.SIunits.Conversions.from_degF(T_sal), 0);
  T_ent = Modelica.SIunits.Conversions.to_degF(proce.T);
  SalTher.h = procs.h;
  POT = POT_NOM * POT_NOM_T;
  EER = EER_NOM * EER_TERM_T * EER_TERM_FCP;
  annotation(
    Icon(coordinateSystem(preserveAspectRatio = false,
initialScale = 0.1), graphics = {Rectangle(lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid, extent = {{-100, 60}, {100, -60}}),
Line(points = {{-80, 60}, {-80, -60}}), Line(points = {{80, 60},
{80, -60}}), Line(points = {{-80, 0}, {-60, 0}, {-40, 20}, {40,
-20}, {60, 0}, {80, 0}}, color = {0, 0, 255})}),
    Diagram(coordinateSystem(preserveAspectRatio = false, extent
= {{-100, -100}, {100, 100}}, grid = {2, 2}), graphics =
{Rectangle(extent = {{-100, 60}, {100, -60}}, lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid), Line(points = {{-80, 60}, {-80, -60}}),
Line(points = {{80, 60}, {80, -60}}), Line(points = {{-80, 0},
{-60, 0}, {-40, 20}, {40, -20}, {60, 0}, {80, 0}}, color = {0,
0, 255})}),
    Window(x = 0.05, y = 0.01, width = 0.93, height = 0.87),
    Documentation(info = "<html>
```

```
<p><b>Copyright &copy; EDF 2002 - 2014</b> </p>
<p><b>ThermoSysPro Version 3.1</b> </p>
</html>", revisions = "<html>
<u><p><b>Authors</u> : </p></b>
<ul style='margin-top:0cm' type=disc>
<li>
    Daniel Bouskela</li>
</ul>
</html>
    "),
    uses(ThermoSysPro(version = "3.1")));
end Maquina_absorcion;
```

model Bombacalor_ref:

```
model Bombacalor_ref
 parameter Real a = 0.453776;
 parameter Real b = 0.015342;
 parameter Real c = 0.000182;
 parameter Real d = -0.001328;
 parameter Real e = 0.000021;
 parameter Real f = -0.000140;
 parameter Real a1 = 0.951894;
 parameter Real b1 = -0.010518;
 parameter Real c1 = 0.000126;
 parameter Real d1 = -0.003399;
 parameter Real e1 = 0.000183;
 parameter Real f1 = -0.000206;
 parameter Real a2 = 0.088065;
 parameter Real b2 = 1.137742;
 parameter Real c2 = -0.225806;
 parameter Real d2 = 0.000000;
 parameter kW POT_NOM;
 parameter Real EER_NOM;

  ThermoSysPro.WaterSteam.Connectors.FluidInlet EntTher
                          annotation (Placement(visible = true,
transformation(extent = {{-8, -68}, {12, -48}}, rotation = 0),
iconTransformation(extent = {{-8, -70}, {12, -50}}, rotation =
0)));
  ThermoSysPro.WaterSteam.Connectors.FluidOutlet SalTher
annotation (Placement(visible = true, transformation(extent =
{{-8, 52}, {12, 72}}, rotation = 0)));

Modelica.SIunits.Conversions.NonSIunits.Temperature_degF T_ent;
Modelica.SIunits.Conversions.NonSIunits.Temperature_degF T_sal;
protected
Real POT_NOM_T;
Real EER_ELEC_T;
Real EER_ELEC_FCP;
Real PLR;
public
kW POT;
Real EER;
```

```
ThermoSysPro.InstrumentationAndControl.Connectors.InputReal TEva

    annotation (Placement(visible = true, transformation(origin
= {-120, 60}, extent = {{20, -20}, {-20, 20}}, rotation = 180),
iconTransformation(extent = {{20, -20}, {-20, 20}}, rotation =
180)));


ThermoSysPro.Properties.WaterSteam.Common.ThermoProperties_ph
proce annotation(
    Placement(visible = true, transformation(origin = {-80, 84},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.Properties.WaterSteam.Common.ThermoProperties_pT
procs annotation(
    Placement(visible = true, transformation(origin = {-54, 84},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));

equation

  TEva.signal = T_sal;

  EntTher.P = SalTher.P;

  0 = if (EntTher.Q > 0) then EntTher.h - EntTher.h_vol else
SalTher.h - SalTher.h_vol;

  EntTher.Q = SalTher.Q;

    POT_NOM_T = a + b * T_sal + c * (T_sal)^2 + d * T_ent + e *
(T_ent)^2 + f * T_sal * T_ent;

    EER_ELEC_T = a1 + b1 * T_sal + c1 * (T_sal)^2 + d1 * T_ent +
e1 * (T_ent)^2 + f1 * T_sal * T_ent;

    PLR = 1;
    EER_ELEC_FCP = a2 + b2 * PLR + c2 * (PLR)^2 + d2 * (PLR)^3;

  proce =
ThermoSysPro.Properties.WaterSteam.IF97.Water_Ph(EntTher.P,
EntTher.h, 0);
  procs =
ThermoSysPro.Properties.WaterSteam.IF97.Water_PT(SalTher.P,
Modelica.SIunits.Conversions.from_degF(T_sal), 0);

  T_ent = Modelica.SIunits.Conversions.to_degF(proce.T);
  SalTher.h = procs.h;

  POT = POT_NOM * POT_NOM_T;
  EER = EER_NOM * EER_ELEC_T * EER_ELEC_FCP;

  annotation(
    Icon(coordinateSystem(preserveAspectRatio = false,
initialScale = 0.1), graphics = {Rectangle(lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid, extent = {{-100, 60}, {100, -60}}),
```

```
Line(points = {{-80, 60}, {-80, -60}}), Line(points = {{80, 60},
{80, -60}}), Line(points = {{-80, 0}, {-60, 0}, {-40, 20}, {40,
-20}, {60, 0}, {80, 0}}, color = {0, 0, 255})}),
    Diagram(coordinateSystem(preserveAspectRatio = false, extent
= {{-100, -100}, {100, 100}}, grid = {2, 2}), graphics =
{Rectangle(extent = {{-100, 60}, {100, -60}}, lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid), Line(points = {{-80, 60}, {-80, -60}}),
Line(points = {{80, 60}, {80, -60}}), Line(points = {{-80, 0},
{-60, 0}, {-40, 20}, {40, -20}, {60, 0}, {80, 0}}, color = {0,
0, 255})}),
    Window(x = 0.05, y = 0.01, width = 0.93, height = 0.87),
    Documentation(info = "<html>
<p><b>Copyright &copy; EDF 2002 - 2014</b> </p>
<p><b>ThermoSysPro Version 3.1</b> </p>
</html>", revisions = "<html>
<u><p><b>Authors</u> : </p></b>
<ul style='margin-top:0cm' type=disc>
<li>
    Daniel Bouskela</li>
</ul>
</html>
    "),
 uses(ThermoSysPro(version = "3.1")));


    end Bombacalor_ref;
```

model Bombacalor_cal:

```
model Bombacalor_cal
 parameter Real a = 0.421783;
 parameter Real b = -0.000674;
 parameter Real c = 0.000007;
 parameter Real d = 0.013689;
 parameter Real e = 0.000040;
 parameter Real f = -0.000039;
 parameter Real a1 = 0.132733;
 parameter Real b1 = 0.012322;
 parameter Real c1 = 0.000032;
 parameter Real d1 = -0.011109;
 parameter Real e1 = 0.000125;
 parameter Real f1 = -0.000123;
 parameter Real a2 = 0.088065;
 parameter Real b2 = 1.137742;
 parameter Real c2 = -0.225806;
 parameter Real d2 = 0.000000;
 parameter kW POT_CAL_NOM;
 parameter Real COP_NOM;


  ThermoSysPro.WaterSteam.Connectors.FluidInlet EntTher
```

```
                          annotation (Placement(visible = true,
transformation(extent = {{-8, -68}, {12, -48}}, rotation = 0),
iconTransformation(extent = {{-8, -70}, {12, -50}}, rotation =
0)));
  ThermoSysPro.WaterSteam.Connectors.FluidOutlet SalTher
annotation (Placement(visible = true, transformation(extent =
{{-8, 52}, {12, 72}}, rotation = 0)));

Modelica.SIunits.Conversions.NonSIunits.Temperature_degF T_ent;
Modelica.SIunits.Conversions.NonSIunits.Temperature_degF T_sal;
protected
Real POT_NOM_CAL_T;
Real COP_ELEC_T;
Real COP_ELEC_FCP;
Real PLR;
public
kW POT_CAL;
Real COP;
ThermoSysPro.InstrumentationAndControl.Connectors.InputReal TCon

    annotation (Placement(visible = true, transformation(origin
= {-120, 60}, extent = {{20, -20}, {-20, 20}}, rotation = 180),
iconTransformation(extent = {{20, -20}, {-20, 20}}, rotation =
180)));

ThermoSysPro.Properties.WaterSteam.Common.ThermoProperties_ph
proce annotation(
    Placement(visible = true, transformation(origin = {-80, 84},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.Properties.WaterSteam.Common.ThermoProperties_pT
procs annotation(
    Placement(visible = true, transformation(origin = {-54, 84},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));

equation

  TCon.signal = T_sal;

  EntTher.P = SalTher.P;


  0 = if (EntTher.Q > 0) then EntTher.h - EntTher.h_vol else
SalTher.h - SalTher.h_vol;

  EntTher.Q = SalTher.Q;

    POT_NOM_CAL_T = a + b * T_sal + c * (T_sal)^2 + d * T_ent +
e * (T_ent)^2 + f * T_sal * T_ent;

    COP_ELEC_T = a1 + b1 * T_sal + c1 * (T_sal)^2 + d1 * T_ent +
e1 * (T_ent)^2 + f1 * T_sal * T_ent;

    PLR = 1;
    COP_ELEC_FCP = a2 + b2 * PLR + c2 * (PLR)^2 + d2 * (PLR)^3;
```

```
  proce =
ThermoSysPro.Properties.WaterSteam.IF97.Water_Ph(EntTher.P,
EntTher.h, 0);
  procs =
ThermoSysPro.Properties.WaterSteam.IF97.Water_PT(SalTher.P,
Modelica.SIunits.Conversions.from_degF(T_sal), 0);

  T_ent = Modelica.SIunits.Conversions.to_degF(proce.T);
  SalTher.h = procs.h;

  POT_CAL = POT_CAL_NOM * POT_NOM_CAL_T;
  COP = COP_NOM * COP_ELEC_T * COP_ELEC_FCP;

  annotation(
    Icon(coordinateSystem(preserveAspectRatio = false,
initialScale = 0.1), graphics = {Rectangle(lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid, extent = {{-100, 60}, {100, -60}}),
Line(points = {{-80, 60}, {-80, -60}}), Line(points = {{80, 60},
{80, -60}}), Line(points = {{-80, 0}, {-60, 0}, {-40, 20}, {40,
-20}, {60, 0}, {80, 0}}, color = {0, 0, 255})}),
    Diagram(coordinateSystem(preserveAspectRatio = false, extent
= {{-100, -100}, {100, 100}}, grid = {2, 2}), graphics =
{Rectangle(extent = {{-100, 60}, {100, -60}}, lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid), Line(points = {{-80, 60}, {-80, -60}}),
Line(points = {{80, 60}, {80, -60}}), Line(points = {{-80, 0},
{-60, 0}, {-40, 20}, {40, -20}, {60, 0}, {80, 0}}, color = {0,
0, 255})}),
    Window(x = 0.05, y = 0.01, width = 0.93, height = 0.87),
    Documentation(info = "<html>
<p><b>Copyright &copy; EDF 2002 - 2014</b> </p>
<p><b>ThermoSysPro Version 3.1</b> </p>
</html>", revisions = "<html>
<u><p><b>Authors</u> : </p></b>
<ul style='margin-top:0cm' type=disc>
<li>
    Daniel Bouskela</li>
</ul>
</html>
    "),
 uses(ThermoSysPro(version = "3.1")));

    end Bombacalor_cal;
```

Anexo D: Text View Adaptadores

model AdapRealMT:

```
model AdapWaterMT
replaceable package Medium =
Modelica.Media.Water.StandardWaterOnePhase constrainedby
Modelica.Media.Interfaces.PartialMedium;
  Modelica.Fluid.Interfaces.FluidPort_a EntMod(redeclare package
Medium = Medium) annotation(
    Placement(transformation(extent = {{-110, -10}, {-90, 10}},
rotation = 0)));
  ThermoSysPro.WaterSteam.Connectors.FluidOutlet SalTher
annotation(
    Placement(visible = true, transformation(extent = {{90, -
10}, {110, 10}}, rotation = 0), iconTransformation(extent =
{{90, -12}, {110, 8}}, rotation = 0)));

equation

  EntMod.p = SalTher.P;

  EntMod.h_outflow = SalTher.h_vol;

  0 = EntMod.m_flow + SalTher.Q;

  0 = EntMod.m_flow*EntMod.h_outflow + SalTher.Q*SalTher.h;

  annotation(
    Icon(coordinateSystem(preserveAspectRatio = false,
initialScale = 0.1), graphics = {Rectangle(lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid, extent = {{-100, 60}, {100, -60}}),
Line(points = {{-80, 60}, {-80, -60}}), Line(points = {{80, 60},
{80, -60}}), Line(points = {{-80, 0}, {-60, 0}, {-40, 20}, {40,
-20}, {60, 0}, {80, 0}}, color = {0, 0, 255})}),
    Diagram(coordinateSystem(preserveAspectRatio = false, extent
= {{-100, -100}, {100, 100}}, grid = {2, 2}), graphics =
{Rectangle(extent = {{-100, 60}, {100, -60}}, lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid), Line(points = {{-80, 60}, {-80, -60}}),
Line(points = {{80, 60}, {80, -60}}), Line(points = {{-80, 0},
{-60, 0}, {-40, 20}, {40, -20}, {60, 0}, {80, 0}}, color = {0,
0, 255})}),
    Window(x = 0.05, y = 0.01, width = 0.93, height = 0.87),
    Documentation(info = "<html>
<p><b>Copyright &copy; EDF 2002 - 2014</b> </p>
<p><b>ThermoSysPro Version 3.1</b> </p>
</html>", revisions = "<html>
<u><p><b>Authors</u> : </p></b>
<ul style='margin-top:0cm' type=disc>
<li>
    Daniel Bouskela</li>
```

```
</ul>
</html>
    "));

end AdapWaterMT;
```

model AdapRealTM:

```
model AdapWaterTM
replaceable package Medium =
Modelica.Media.Water.StandardWaterOnePhase constrainedby
Modelica.Media.Interfaces.PartialMedium;
  Modelica.Fluid.Interfaces.FluidPort_b SalMod(redeclare package
Medium = Medium) annotation(
    Placement(visible = true, transformation(extent = {{90, -
10}, {110, 10}}, rotation = 0), iconTransformation(extent =
{{90, -10}, {110, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Connectors.FluidInlet EntTher
annotation(
    Placement(visible = true, transformation(extent = {{-110, -
10}, {-90, 10}}, rotation = 0), iconTransformation(extent = {{-
110, -8}, {-90, 12}}, rotation = 0)));

equation

  EntTher.P = SalMod.p;

  EntTher.h_vol = SalMod.h_outflow;

  0 = EntTher.Q*EntTher.h + SalMod.m_flow*SalMod.h_outflow;

  0 = EntTher.Q + SalMod.m_flow;

  annotation(
    Icon(coordinateSystem(preserveAspectRatio = false,
initialScale = 0.1), graphics = {Rectangle(lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid, extent = {{-100, 60}, {100, -60}}),
Line(points = {{-80, 60}, {-80, -60}}), Line(points = {{80, 60},
{80, -60}}), Line(points = {{-80, 0}, {-60, 0}, {-40, 20}, {40,
-20}, {60, 0}, {80, 0}}, color = {0, 0, 255})}),
    Diagram(coordinateSystem(preserveAspectRatio = false, extent
= {{-100, -100}, {100, 100}}, grid = {2, 2}), graphics =
{Rectangle(extent = {{-100, 60}, {100, -60}}, lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid), Line(points = {{-80, 60}, {-80, -60}}),
Line(points = {{80, 60}, {80, -60}}), Line(points = {{-80, 0},
{-60, 0}, {-40, 20}, {40, -20}, {60, 0}, {80, 0}}, color = {0,
0, 255})}),
    Window(x = 0.05, y = 0.01, width = 0.93, height = 0.87),
    Documentation(info = "<html>
<p><b>Copyright &copy; EDF 2002 - 2014</b> </p>
<p><b>ThermoSysPro Version 3.1</b> </p>
</html>", revisions = "<html>
```

```
<u><p><b>Authors</u> : </p></b>
<ul style='margin-top:0cm' type=disc>
<li>
    Daniel Bouskela</li>
</ul>
</html>
    "));

    end AdapWaterTM;
```

model AdapRealMT:

```
model AdapRealMT
  ThermoSysPro.InstrumentationAndControl.Connectors.OutputReal
SalidaTher annotation (Placement(transformation(
          extent={{100,-10},{120,10}}, rotation=0)));
  Modelica.Blocks.Interfaces.RealInput EntradaMod annotation
(Placement(
        visible = true, transformation(extent = {{-140, -20}, {-
100, 20}}, rotation = 0), iconTransformation(origin = {-112, 0},
extent = {{-12, -12}, {12, 12}}, rotation = 0)));
 equation

EntradaMod = SalidaTher.signal;

  annotation(
    Icon(coordinateSystem(preserveAspectRatio = false,
initialScale = 0.1), graphics = {Rectangle(lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid, extent = {{-100, 60}, {100, -60}}),
Line(points = {{-80, 60}, {-80, -60}}), Line(points = {{80, 60},
{80, -60}}), Line(points = {{-80, 0}, {-60, 0}, {-40, 20}, {40,
-20}, {60, 0}, {80, 0}}, color = {0, 0, 255})}),
    Diagram(coordinateSystem(preserveAspectRatio = false,
initialScale = 0.1), graphics = {Rectangle( lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid, extent = {{-100, 60}, {100, -60}}),
Line(points = {{-80, 60}, {-80, -60}}), Line(points = {{80, 60},
{80, -60}}), Line(points = {{-80, 0}, {-60, 0}, {-40, 20}, {40,
-20}, {60, 0}, {80, 0}}, color = {0, 0, 255})}),
    Window(x = 0.05, y = 0.01, width = 0.93, height = 0.87),
    Documentation(info = "<html>
<p><b>Copyright &copy; EDF 2002 - 2014</b> </p>
<p><b>ThermoSysPro Version 3.1</b> </p>
</html>", revisions = "<html>
<u><p><b>Authors</u> : </p></b>
<ul style='margin-top:0cm' type=disc>
<li>
    Daniel Bouskela</li>
</ul>
</html>
    "));
```

```
    end AdapRealMT;
```

model AdapRealTM:

```
model AdapRealTM

ThermoSysPro.InstrumentationAndControl.Connectors.InputReal
EntradaTher    annotation (Placement(transformation(
          extent={{-120,-10},{-100,10}}, rotation=0)));
  Modelica.Blocks.Interfaces.RealOutput SalidaMod annotation
(Placement(
        visible = true, transformation(origin = {119, -1},
extent = {{-19, -19}, {19, 19}}, rotation = 0),
iconTransformation(extent = {{-18, -8}, {2, 12}}, rotation =
0)));

equation

EntradaTher.signal = SalidaMod;

  annotation(
    Icon(coordinateSystem(preserveAspectRatio = false,
initialScale = 0.1), graphics = {Rectangle(lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid, extent = {{-100, 60}, {100, -60}}),
Line(points = {{-80, 60}, {-80, -60}}), Line(points = {{80, 60},
{80, -60}}), Line(points = {{-80, 0}, {-60, 0}, {-40, 20}, {40,
-20}, {60, 0}, {80, 0}}, color = {0, 0, 255})}),
    Diagram(coordinateSystem(preserveAspectRatio = false,
initialScale = 0.1), graphics = {Rectangle( lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid, extent = {{-100, 60}, {100, -60}}),
Line(points = {{-80, 60}, {-80, -60}}), Line(points = {{80, 60},
{80, -60}}), Line(points = {{-80, 0}, {-60, 0}, {-40, 20}, {40,
-20}, {60, 0}, {80, 0}}, color = {0, 0, 255})}),
    Window(x = 0.05, y = 0.01, width = 0.93, height = 0.87),
    Documentation(info = "<html>
<p><b>Copyright &copy; EDF 2002 - 2014</b> </p>
<p><b>ThermoSysPro Version 3.1</b> </p>
</html>", revisions = "<html>
<u><p><b>Authors</u> : </p></b>
<ul style='margin-top:0cm' type=disc>
<li>
    Daniel Bouskela</li>
</ul>
</html>
    "));

end AdapRealTM;
```

model AdapBooleanMT:

```
model AdapBooleanMT

ThermoSysPro.InstrumentationAndControl.Connectors.OutputLogical
SalidaTher annotation (Placement(transformation(
          extent={{100,-10},{120,10}}, rotation=0)));
  Modelica.Blocks.Interfaces.BooleanInput EntradaMod annotation
(Placement(
        visible = true, transformation(extent = {{-140, -20}, {-
100, 20}}, rotation = 0), iconTransformation(origin = {-111, 1},
extent = {{-11, -11}, {11, 11}}, rotation = 0)));
  equation

EntradaMod = SalidaTher.signal;

  annotation(
    Icon(coordinateSystem(preserveAspectRatio = false,
initialScale = 0.1), graphics = {Rectangle(lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid, extent = {{-100, 60}, {100, -60}}),
Line(points = {{-80, 60}, {-80, -60}}), Line(points = {{80, 60},
{80, -60}}), Line(points = {{-80, 0}, {-60, 0}, {-40, 20}, {40,
-20}, {60, 0}, {80, 0}}, color = {0, 0, 255})}),
    Diagram(coordinateSystem(preserveAspectRatio = false,
initialScale = 0.1), graphics = {Rectangle( lineColor = {0, 0,
255}, fillColor = {255, 255, 0}, fillPattern =
FillPattern.Solid, extent = {{-100, 60}, {100, -60}}),
Line(points = {{-80, 60}, {-80, -60}}), Line(points = {{80, 60},
{80, -60}}), Line(points = {{-80, 0}, {-60, 0}, {-40, 20}, {40,
-20}, {60, 0}, {80, 0}}, color = {0, 0, 255})}),
    Window(x = 0.05, y = 0.01, width = 0.93, height = 0.87),
    Documentation(info = "<html>
<p><b>Copyright &copy; EDF 2002 - 2014</b> </p>
<p><b>ThermoSysPro Version 3.1</b> </p>
</html>", revisions = "<html>
<u><p><b>Authors</u> : </p></b>
<ul style='margin-top:0cm' type=disc>
<li>
    Daniel Bouskela</li>
</ul>
</html>
    "));

    end AdapBooleanMT;
```

Anexo E: Text View Ejemplos

Ejemplo 1:

```
model Test_MotoIntgases_1
  ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Rampe
rampe1(Duration = 500, Finalvalue = 1300, Initialvalue = 800)
annotation(
    Placement(visible = true, transformation(origin = {40, -10},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.FlueGases.BoundaryConditions.SourcePQ
sourcePQ2(T0 = 298.15, Xh2o = 0.2) annotation(
    Placement(visible = true, transformation(origin = {-28, -6},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.FlueGases.BoundaryConditions.Sink sink2
annotation(
    Placement(visible = true, transformation(origin = {24, 34},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe1 annotation(
    Placement(visible = true, transformation(origin = {40, -54},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Sensors.SensorT sensorT1 annotation(
    Placement(visible = true, transformation(origin = {-36, 24},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Sensors.SensorT sensorT2 annotation(
    Placement(visible = true, transformation(origin = {54, 24},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.WaterSteam.HeatExchangers.StaticWaterWaterExchanger
staticWaterWaterExchanger1 annotation(
    Placement(visible = true, transformation(origin = {-10, -
54}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.BoundaryConditions.SourceP sourceP1(P0
= 300000, T0 = 293.15) annotation(
    Placement(visible = true, transformation(origin = {-26, -
76}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.BoundaryConditions.SinkP sinkP1
annotation(
    Placement(visible = true, transformation(origin = {16, -80},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.LoopBreakers.LoopBreakerQ
loopBreakerQ1 annotation(
    Placement(visible = true, transformation(origin = {-78, -
28}, extent = {{10, -10}, {-10, 10}}, rotation = -90)));
  ThermoSysPro.WaterSteam.BoundaryConditions.RefP refP1
annotation(
    Placement(visible = true, transformation(origin = {-68, 16},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Machines.StaticCentrifugalPump
staticCentrifugalPump1 annotation(
    Placement(visible = true, transformation(origin = {84, 16},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
```

```
  ThermoSysPro.MultiFluids.Machines.AlternatingEngine
alternatingEngine1 annotation(
    Placement(visible = true, transformation(origin = {-8, 16},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.Combustion.BoundaryConditions.FuelSourcePQ
fuelSourcePQ1(Q0 = 6)  annotation(
    Placement(visible = true, transformation(origin = {-22, -
22}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.MultiFluids.HeatExchangers.StaticExchangerWaterStea
mFlueGases staticExchangerWaterSteamFlueGases1 annotation(
    Placement(visible = true, transformation(origin = {10, 60},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.BoundaryConditions.Sink sink1
annotation(
    Placement(visible = true, transformation(origin = {42, 60},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.BoundaryConditions.SourcePQ
sourcePQ1(h0 = 200000)  annotation(
    Placement(visible = true, transformation(origin = {-26, 60},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
equation
  connect(staticExchangerWaterSteamFlueGases1.Cws2, sink1.C)
annotation(
    Line(points = {{20, 60}, {32, 60}, {32, 60}, {32, 60}},
color = {0, 0, 255}));
  connect(sourcePQ1.C, staticExchangerWaterSteamFlueGases1.Cws1)
annotation(
    Line(points = {{-16, 60}, {0, 60}, {0, 60}, {0, 60}}, color
= {0, 0, 255}));
  connect(staticExchangerWaterSteamFlueGases1.Cfg2, sink2.C)
annotation(
    Line(points = {{10, 50}, {10, 50}, {10, 34}, {14, 34}, {14,
34}}));
  connect(alternatingEngine1.Cfg,
staticExchangerWaterSteamFlueGases1.Cfg1) annotation(
    Line(points = {{-8, 26}, {-8, 80}, {10, 80}, {10, 69}}));
  connect(sourceP1.C, staticWaterWaterExchanger1.Ef) annotation(
    Line(points = {{-16, -76}, {-4, -76}, {-4, -60}}, color =
{0, 0, 255}));
  connect(refP1.C2, sensorT1.C1) annotation(
    Line(points = {{-58, 16}, {-46, 16}}, color = {0, 0, 255}));
  connect(loopBreakerQ1.C2, refP1.C1) annotation(
    Line(points = {{-78, -18}, {-78, 16}}, color = {0, 0,
255}));
  connect(rampe1.y, staticCentrifugalPump1.rpm_or_mpower)
annotation(
    Line(points = {{52, -10}, {84, -10}, {84, 6}, {84, 6}},
color = {0, 0, 255}));
  connect(fuelSourcePQ1.C, alternatingEngine1.Cfuel) annotation(
    Line(points = {{-12, -22}, {-12, -22}, {-12, 6}, {-12,
6}}));
  connect(sourcePQ2.C, alternatingEngine1.Cair) annotation(
    Line(points = {{-18, -6}, {-4, -6}, {-4, 6}, {-4, 6}}));
  connect(alternatingEngine1.Cws2, sensorT2.C1) annotation(
```

```
    Line(points = {{2, 16}, {44, 16}, {44, 16}, {44, 16}}, color
= {0, 0, 255}));
  connect(sensorT1.C2, alternatingEngine1.Cws1) annotation(
    Line(points = {{-26, 16}, {-16, 16}, {-16, 16}, {-18, 16}},
color = {0, 0, 255}));
  connect(sensorT2.C2, staticCentrifugalPump1.C1) annotation(
    Line(points = {{64, 16}, {74, 16}}, color = {0, 0, 255}));
  connect(staticCentrifugalPump1.C2, lumpedStraightPipe1.C1)
annotation(
    Line(points = {{94, 16}, {94, -54}, {50, -54}}, color = {0,
0, 255}));
  connect(staticWaterWaterExchanger1.Sc, loopBreakerQ1.C1)
annotation(
    Line(points = {{-20, -54}, {-78, -54}, {-78, -38}}, color =
{0, 0, 255}));
  connect(staticWaterWaterExchanger1.Sf, sinkP1.C) annotation(
    Line(points = {{-16, -60}, {-14, -60}, {-14, -70}, {6, -70},
{6, -80}, {6, -80}, {6, -80}}, color = {0, 0, 255}));
  connect(lumpedStraightPipe1.C2, staticWaterWaterExchanger1.Ec)
annotation(
    Line(points = {{30, -54}, {0, -54}, {0, -54}, {0, -54}},
color = {0, 0, 255}));
  annotation(
    uses(ThermoSysPro(version = "3.1"),
Modelica_StateGraph2(version = "2.0.3")));

end Test_MotoIntgases_1;
```

Ejemplo 2:

```
model Test_absorcion_torre_1
  Modelica.Blocks.Logical.OnOffController onOffController1
annotation(
    Placement(visible = true, transformation(origin = {-38, 72},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Sources.Constant const1(k = 0) annotation(
    Placement(visible = true, transformation(origin = {-78, 78},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Logical.TriggeredTrapezoid
triggeredTrapezoid1(amplitude = 5, offset = 273.15 + 25)
annotation(
    Placement(visible = true, transformation(origin = {-2, 72},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Sources.CombiTimeTable
combiTimeTable1(smoothness =
Modelica.Blocks.Types.Smoothness.MonotoneContinuousDerivative1,
table = [0, 30; 50, -15; 200, 30; 250, -30; 300, 25; 500, -30])
annotation(
    Placement(visible = true, transformation(origin = {-78, 42},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Buildings.Fluid.HeatExchangers.CoolingTowers.FixedApproach
fixedApproach1(redeclare package Medium = Buildings.Media.Water,
dp_nominal = 500, m_flow_nominal = 500) annotation(
```

```
    Placement(visible = true, transformation(origin = {-32, 16},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  AdapWaterTM adapWaterTM1 annotation(
    Placement(visible = true, transformation(origin = {-28, -
38}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.BoundaryConditions.SourceP sourceP1(T0
= 308.15) annotation(
    Placement(visible = true, transformation(origin = {-60, -
38}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Fluid.Sensors.TemperatureTwoPort
temperature(redeclare package Medium = Buildings.Media.Water)
annotation(
    Placement(visible = true, transformation(origin = {26, 16},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Fluid.Pipes.StaticPipe pipe(redeclare package Medium
= Buildings.Media.Water, diameter = 0.2, length = 10)
annotation(
    Placement(visible = true, transformation(origin = {-2, 16},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Buildings.Fluid.HeatExchangers.HeaterCooler_T hea(redeclare
package Medium = Buildings.Media.Water, dp_nominal = 800,
m_flow_nominal = 800)  annotation(
    Placement(visible = true, transformation(origin = {60, 16},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Sources.Pulse pulse1(amplitude = 5, offset =
273.15 + 15, period = 200, width = 50)  annotation(
    Placement(visible = true, transformation(origin = {32, 50},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Fluid.Sensors.TemperatureTwoPort temperature1
(redeclare package Medium = Buildings.Media.Water) annotation(
    Placement(visible = true, transformation(origin = {94, 16},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  AdapWaterMT adapWaterMT1 annotation(
    Placement(visible = true, transformation(origin = {130, 16},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.BoundaryConditions.SinkP sinkP1
annotation(
    Placement(visible = true, transformation(origin = {164, 16},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
equation
  connect(adapWaterMT1.SalTher, sinkP1.C) annotation(
    Line(points = {{140, 16}, {154, 16}, {154, 16}, {154, 16}},
color = {0, 0, 255}));
  connect(temperature1.port_b, adapWaterMT1.EntMod) annotation(
    Line(points = {{104, 16}, {120, 16}, {120, 16}, {120, 16}},
color = {0, 127, 255}));
  connect(hea.port_b, temperature1.port_a) annotation(
    Line(points = {{70, 16}, {84, 16}, {84, 16}, {84, 16}},
color = {0, 127, 255}));
  connect(pulse1.y, hea.TSet) annotation(
    Line(points = {{44, 50}, {46, 50}, {46, 22}, {48, 22}},
color = {0, 0, 127}));
  connect(temperature.port_b, hea.port_a) annotation(
    Line(points = {{36, 16}, {50, 16}, {50, 16}, {50, 16}},
color = {0, 127, 255}));
  connect(pipe.port_b, temperature.port_a) annotation(
```

```
    Line(points = {{8, 16}, {16, 16}, {16, 16}, {16, 16}}, color
= {0, 127, 255}));
  connect(fixedApproach1.port_b, pipe.port_a) annotation(
    Line(points = {{-22, 16}, {-12, 16}, {-12, 16}, {-12, 16}},
color = {0, 127, 255}));
  connect(adapWaterTM1.SalMod, fixedApproach1.port_a)
annotation(
    Line(points = {{-18, -38}, {-12, -38}, {-12, -22}, {-50, -
22}, {-50, 16}, {-42, 16}}, color = {0, 127, 255}));
  connect(triggeredTrapezoid1.y, fixedApproach1.TAir)
annotation(
    Line(points = {{10, 72}, {18, 72}, {18, 36}, {-50, 36}, {-
50, 20}, {-44, 20}}, color = {0, 0, 127}));
  connect(combiTimeTable1.y[1], onOffController1.u) annotation(
    Line(points = {{-66, 42}, {-56, 42}, {-56, 66}, {-50, 66},
{-50, 66}}, color = {0, 0, 127}));
  connect(const1.y, onOffController1.reference) annotation(
    Line(points = {{-67, 78}, {-50, 78}}, color = {0, 0, 127}));
  connect(onOffController1.y, triggeredTrapezoid1.u) annotation(
    Line(points = {{-27, 72}, {-14, 72}}, color = {255, 0,
255}));
  connect(sourceP1.C, adapWaterTM1.EntTher) annotation(
    Line(points = {{-50, -38}, {-44, -38}, {-44, -38}, {-38, -
38}, {-38, -38}, {-38, -38}, {-38, -38}, {-38, -38}}, color =
{0, 0, 255}));
  annotation(
    uses(Buildings(version = "4.0.0"), ThermoSysPro(version =
"3.1"), Modelica(version = "3.2.2")),
    Diagram(coordinateSystem(extent = {{-200, -200}, {200,
200}})),
    Icon(coordinateSystem(extent = {{-200, -200}, {200, 200}})),
    version = "",
    __OpenModelica_commandLineOptions = "");

    end Test_absorcion_torre_1;
```

Ejemplo 3:

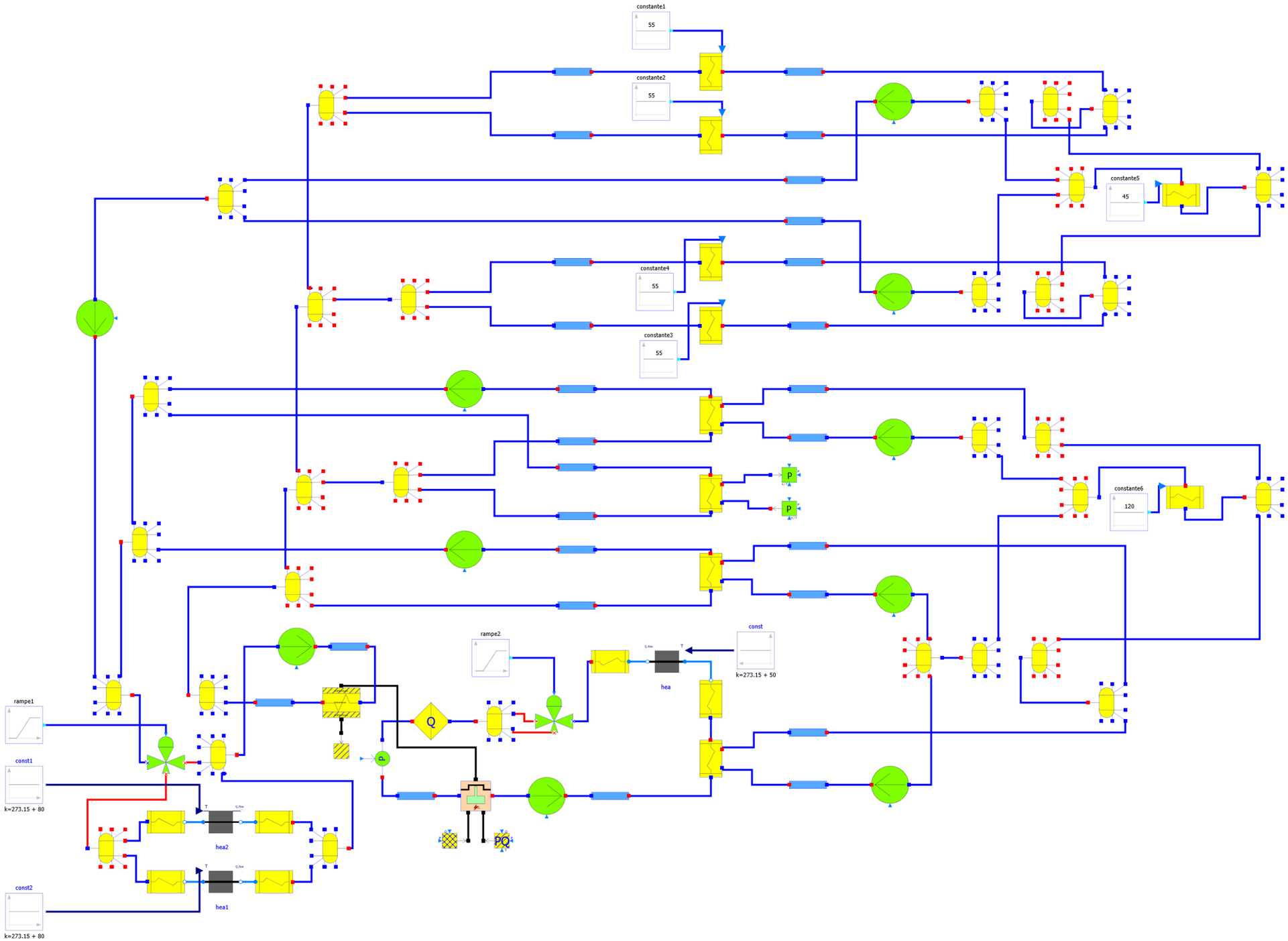
```
model Test_altern_valv
  ThermoSysPro.WaterSteam.PressureLosses.ThreeWayValve
threeWayValve1 annotation(
    Placement(visible = true, transformation(origin = {-6, 26},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.BoundaryConditions.SourceP sourceP1(T0
= 288.15) annotation(
    Placement(visible = true, transformation(origin = {-74, -
38}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Junctions.Mixer8 mixer81 annotation(
    Placement(visible = true, transformation(origin = {46, -10},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.BoundaryConditions.SinkP sinkP1
annotation(
    Placement(visible = true, transformation(origin = {74, -10},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
```

```
ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Constante
constante1(k = 100) annotation(
    Placement(visible = true, transformation(origin = {-90, -4},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Constante
constante2(k = 50) annotation(
    Placement(visible = true, transformation(origin = {-36, -
24}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Constante
constante3(k = 50) annotation(
    Placement(visible = true, transformation(origin = {-2, 58},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Echelon
echelon1(startTime = 250) annotation(
    Placement(visible = true, transformation(origin = {-34, 46},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe1 annotation(
    Placement(visible = true, transformation(origin = {52, 58},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe2 annotation(
    Placement(visible = true, transformation(origin = {-6, -34},
extent = {{-10, -10}, {10, 10}}, rotation = -90)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe3 annotation(
    Placement(visible = true, transformation(origin = {-38, 22},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Maquina_absorcion_1 maquina_absorcion_11(EER_NOM = 4.7,
POT_NOM = 100) annotation(
    Placement(visible = true, transformation(origin = {-6, -4},
extent = {{-10, 10}, {10, -10}}, rotation = 0)));
  Bombacalor_ref_1 bombacalor_ref_11(EER_NOM = 4.7, POT_NOM =
130) annotation(
    Placement(visible = true, transformation(origin = {28, 30},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Bombacalor_cal_1 bombacalor_cal_11(COP_NOM = 4.7, POT_CAL_NOM
= 130) annotation(
    Placement(visible = true, transformation(origin = {-60, -
10}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Sensors.SensorT sensorT1 annotation(
    Placement(visible = true, transformation(origin = {-68, 12},
extent = {{-10, -10}, {10, 10}}, rotation = 90)));
  ThermoSysPro.WaterSteam.Sensors.SensorT sensorT2 annotation(
    Placement(visible = true, transformation(origin = {24, -42},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Sensors.SensorT sensorT3 annotation(
    Placement(visible = true, transformation(origin = {70, 30},
extent = {{-10, -10}, {10, 10}}, rotation = -90)));
equation
  connect(sensorT3.C2, mixer81.Ce1) annotation(
    Line(points = {{62, 20}, {62, 20}, {62, 0}, {50, 0}, {50,
0}}, color = {0, 0, 255}));
```

```
  connect(lumpedStraightPipe1.C2, sensorT3.C1) annotation(
    Line(points = {{62, 58}, {62, 58}, {62, 40}, {62, 40}},
color = {0, 0, 255}));
  connect(sensorT2.C2, mixer81.Ce8) annotation(
    Line(points = {{34, -50}, {48, -50}, {48, -20}, {50, -20}},
color = {0, 0, 255}));
  connect(lumpedStraightPipe2.C2, sensorT2.C1) annotation(
    Line(points = {{-6, -44}, {-6, -44}, {-6, -50}, {14, -50},
{14, -50}}, color = {0, 0, 255}));
  connect(sensorT1.C2, lumpedStraightPipe3.C1) annotation(
    Line(points = {{-60, 22}, {-48, 22}, {-48, 22}, {-48, 22}},
color = {0, 0, 255}));
  connect(bombacalor_cal_11.SalTher, sensorT1.C1) annotation(
    Line(points = {{-60, -4}, {-60, -4}, {-60, 2}, {-60, 2}},
color = {0, 0, 255}));
  connect(constante1.y, bombacalor_cal_11.TCon) annotation(
    Line(points = {{-79, -4}, {-72, -4}}, color = {0, 0, 255}));
  connect(sourceP1.C, bombacalor_cal_11.EntTher) annotation(
    Line(points = {{-64, -38}, {-60, -38}, {-60, -16}}, color =
{0, 0, 255}));
  connect(constante3.y, bombacalor_ref_11.TEva) annotation(
    Line(points = {{10, 58}, {14, 58}, {14, 36}, {16, 36}},
color = {0, 0, 255}));
  connect(bombacalor_ref_11.SalTher, lumpedStraightPipe1.C1)
annotation(
    Line(points = {{28, 36}, {28, 36}, {28, 58}, {42, 58}, {42,
58}}, color = {0, 0, 255}));
  connect(threeWayValve1.C2, bombacalor_ref_11.EntTher)
annotation(
    Line(points = {{4, 22}, {28, 22}, {28, 24}, {28, 24}}, color
= {255, 0, 0}));
  connect(constante2.y, maquina_absorcion_11.TEva) annotation(
    Line(points = {{-24, -24}, {-18, -24}, {-18, -10}, {-18, -
10}}, color = {0, 0, 255}));
  connect(maquina_absorcion_11.SalTher, lumpedStraightPipe2.C1)
annotation(
    Line(points = {{-6, -10}, {-6, -10}, {-6, -24}, {-6, -24}},
color = {0, 0, 255}));
  connect(threeWayValve1.C3, maquina_absorcion_11.EntTher)
annotation(
    Line(points = {{-6, 16}, {-6, 16}, {-6, 2}, {-6, 2}}, color
= {255, 0, 0}));
  connect(lumpedStraightPipe3.C2, threeWayValve1.C1) annotation(
    Line(points = {{-28, 22}, {-18, 22}, {-18, 22}, {-16, 22}},
color = {0, 0, 255}));
  connect(echelon1.y, threeWayValve1.Ouv) annotation(
    Line(points = {{-22, 46}, {-16, 46}, {-16, 38}, {-6, 38}, {-
6, 38}}, color = {0, 0, 255}));
  connect(mixer81.Cs, sinkP1.C) annotation(
    Line(points = {{56, -10}, {64, -10}, {64, -10}, {64, -10}},
color = {0, 0, 255}));
  annotation(
    uses(Buildings(version = "4.0.0"), ThermoSysPro(version =
"3.1"), Modelica(version = "3.2.2")),
    Diagram(coordinateSystem(extent = {{-300, -300}, {300,
300}})),
```

```
Icon(coordinateSystem(extent = {{-300, -300}, {300, 300}})),
version = "",
__OpenModelica_commandLineOptions = "");

end Test_altern_valv;
```

Anexo F: Instalación de cogeneración en OpenModelica



Text View Instalación cogeneración:

```
model Planta_cogeneracion
  ThermoSysPro.MultiFluids.Machines.AlternatingEngine
alternatingEngine1 annotation(
    Placement(visible = true, transformation(origin = {-46, 32},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.MultiFluids.HeatExchangers.StaticExchangerWaterStea
mFlueGases staticExchangerWaterSteamFlueGases1 annotation(
    Placement(visible = true, transformation(origin = {-118,
82}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe1 annotation(
    Placement(visible = true, transformation(origin = {26, 32},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Machines.StaticCentrifugalPump
staticCentrifugalPump1 annotation(
    Placement(visible = true, transformation(origin = {-8, 32},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.WaterSteam.HeatExchangers.StaticWaterWaterExchanger
staticWaterWaterExchanger1 annotation(
    Placement(visible = true, transformation(origin = {80, 52},
extent = {{-10, -10}, {10, 10}}, rotation = 90)));
  Buildings.Fluid.HeatExchangers.HeaterCooler_T
hea(Q_flow_maxHeat = 0, dp_nominal = 1000, m_flow_nominal =
1000)  annotation(
    Placement(visible = true, transformation(origin = {56, 104},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  AdapWaterTM adapWaterTM1 annotation(
    Placement(visible = true, transformation(origin = {80, 84},
extent = {{-10, -10}, {10, 10}}, rotation = 90)));
  AdapWaterMT adapWaterMT1 annotation(
    Placement(visible = true, transformation(origin = {26, 104},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.ThreeWayValve
threeWayValve1 annotation(
    Placement(visible = true, transformation(origin = {-4, 76},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorida colectorida1 annotation(
    Placement(visible = true, transformation(origin = {-36, 72},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe2 annotation(
    Placement(visible = true, transformation(origin = {-78, 32},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.LoopBreakers.LoopBreakerQ
loopBreakerQ1 annotation(
    Placement(visible = true, transformation(origin = {-70, 72},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.BoundaryConditions.RefP refP1
annotation(
    Placement(visible = true, transformation(origin = {-96, 52},
extent = {{10, -10}, {-10, 10}}, rotation = 90)));
```

```
  ThermoSysPro.FlueGases.BoundaryConditions.Sink sink1
annotation(
    Placement(visible = true, transformation(origin = {-118,
56}, extent = {{-10, -10}, {10, 10}}, rotation = -90)));
  ThermoSysPro.FlueGases.BoundaryConditions.SourcePQ
sourcePQ1(T0 = 298.15, Xh2o = 0.2)  annotation(
    Placement(visible = true, transformation(origin = {-32, 8},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.Combustion.BoundaryConditions.FuelSourcePQ
fuelSourcePQ1 annotation(
    Placement(visible = true, transformation(origin = {-60, 8},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe3 annotation(
    Placement(visible = true, transformation(origin = {-154,
82}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorida colectorida2 annotation(
    Placement(visible = true, transformation(origin = {-190,
86}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Machines.StaticCentrifugalPump
staticCentrifugalPump2 annotation(
    Placement(visible = true, transformation(origin = {-142,
112}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe4 annotation(
    Placement(visible = true, transformation(origin = {-114,
112}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorida colectorida3 annotation(
    Placement(visible = true, transformation(origin = {-240,
86}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.ThreeWayValve
threeWayValve2 annotation(
    Placement(visible = true, transformation(origin = {-212,
54}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Buildings.Fluid.HeatExchangers.HeaterCooler_T
hea1(Q_flow_maxHeat = 0, dp_nominal = 1000, m_flow_nominal =
1000)  annotation(
    Placement(visible = true, transformation(origin = {-182, -
14}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Buildings.Fluid.HeatExchangers.HeaterCooler_T
hea2(Q_flow_maxHeat = 0, dp_nominal = 1000, m_flow_nominal =
1000)  annotation(
    Placement(visible = true, transformation(origin = {-182,
18}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  AdapWaterTM adapWaterTM2 annotation(
    Placement(visible = true, transformation(origin = {-212,
18}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  AdapWaterTM adapWaterTM3 annotation(
    Placement(visible = true, transformation(origin = {-212, -
14}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorret colectorret1 annotation(
    Placement(visible = true, transformation(origin = {-244, 4},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  AdapWaterMT adapWaterMT2 annotation(
    Placement(visible = true, transformation(origin = {-154,
18}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
```

```
  AdapWaterMT adapWaterMT3 annotation(
    Placement(visible = true, transformation(origin = {-154, -
14}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorida colectorida4 annotation(
    Placement(visible = true, transformation(origin = {-124, 4},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorida colectorida5 annotation(
    Placement(visible = true, transformation(origin = {-184,
54}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.WaterSteam.HeatExchangers.StaticWaterWaterExchanger
staticWaterWaterExchanger2 annotation(
    Placement(visible = true, transformation(origin = {80, 152},
extent = {{-10, -10}, {10, 10}}, rotation = 90)));

ThermoSysPro.WaterSteam.HeatExchangers.StaticWaterWaterExchanger
staticWaterWaterExchanger3 annotation(
    Placement(visible = true, transformation(origin = {80, 236},
extent = {{-10, -10}, {10, 10}}, rotation = 90)));

ThermoSysPro.WaterSteam.HeatExchangers.StaticWaterWaterExchanger
staticWaterWaterExchanger4 annotation(
    Placement(visible = true, transformation(origin = {80, 194},
extent = {{-10, -10}, {10, 10}}, rotation = 90)));
  Colectorret colectorret2 annotation(
    Placement(visible = true, transformation(origin = {-144,
144}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe5 annotation(
    Placement(visible = true, transformation(origin = {8, 134},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorret colectorret3 annotation(
    Placement(visible = true, transformation(origin = {-138,
196}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorret colectorret4 annotation(
    Placement(visible = true, transformation(origin = {-86,
200}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe6 annotation(
    Placement(visible = true, transformation(origin = {8, 182},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe7 annotation(
    Placement(visible = true, transformation(origin = {8, 222},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Maquina_absorcion_1 maquina_absorcion_11(EER_NOM = 4.7,
POT_NOM = 100)  annotation(
    Placement(visible = true, transformation(origin = {80, 284},
extent = {{-10, -10}, {10, 10}}, rotation = -90)));
  Colectorret colectorret5 annotation(
    Placement(visible = true, transformation(origin = {-132,
294}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorret colectorret6 annotation(
    Placement(visible = true, transformation(origin = {-82,
298}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
```

```
  Maquina_absorcion_1 maquina_absorcion_12(EER_NOM = 4.7,
POT_NOM = 100)  annotation(
    Placement(visible = true, transformation(origin = {80, 318},
extent = {{-10, -10}, {10, 10}}, rotation = -90)));
  Maquina_absorcion_1 maquina_absorcion_13(EER_NOM = 4.7,
POT_NOM = 100)  annotation(
    Placement(visible = true, transformation(origin = {80, 420},
extent = {{-10, -10}, {10, 10}}, rotation = -90)));
  Maquina_absorcion_1 maquina_absorcion_14(EER_NOM = 4.7,
POT_NOM = 100)  annotation(
    Placement(visible = true, transformation(origin = {80, 386},
extent = {{-10, -10}, {10, 10}}, rotation = -90)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe8 annotation(
    Placement(visible = true, transformation(origin = {6, 284},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe9 annotation(
    Placement(visible = true, transformation(origin = {6, 318},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorret colectorret7 annotation(
    Placement(visible = true, transformation(origin = {-126,
402}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe10 annotation(
    Placement(visible = true, transformation(origin = {6, 386},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe11 annotation(
    Placement(visible = true, transformation(origin = {6, 420},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe12 annotation(
    Placement(visible = true, transformation(origin = {132, 66},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe13 annotation(
    Placement(visible = true, transformation(origin = {132,
166}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorida colectorida6 annotation(
    Placement(visible = true, transformation(origin = {292, 82},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorret colectorret8 annotation(
    Placement(visible = true, transformation(origin = {256,
106}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe14 annotation(
    Placement(visible = true, transformation(origin = {132, 38},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Machines.StaticCentrifugalPump
staticCentrifugalPump3 annotation(
    Placement(visible = true, transformation(origin = {176, 38},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Machines.StaticCentrifugalPump
staticCentrifugalPump4 annotation(
```

```
    Placement(visible = true, transformation(origin = {178,
140}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe15 annotation(
    Placement(visible = true, transformation(origin = {132,
140}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.BoundaryConditions.SourceP sourceP1
annotation(
    Placement(visible = true, transformation(origin = {122,
186}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.BoundaryConditions.SinkP sinkP1(P0 =
300000)  annotation(
    Placement(visible = true, transformation(origin = {122,
204}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe16 annotation(
    Placement(visible = true, transformation(origin = {132,
250}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorret colectorret10 annotation(
    Placement(visible = true, transformation(origin = {258,
224}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorida colectorida7 annotation(
    Placement(visible = true, transformation(origin = {224,
106}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorret colectorret9 annotation(
    Placement(visible = true, transformation(origin = {194,
106}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorida colectorida8 annotation(
    Placement(visible = true, transformation(origin = {224,
224}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Machines.StaticCentrifugalPump
staticCentrifugalPump5 annotation(
    Placement(visible = true, transformation(origin = {178,
224}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe17 annotation(
    Placement(visible = true, transformation(origin = {132,
224}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe18 annotation(
    Placement(visible = true, transformation(origin = {132,
284}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe19 annotation(
    Placement(visible = true, transformation(origin = {130,
318}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorret colectorret11 annotation(
    Placement(visible = true, transformation(origin = {258,
302}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorida colectorida9 annotation(
    Placement(visible = true, transformation(origin = {294,
300}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorida colectorida10 annotation(
    Placement(visible = true, transformation(origin = {224,
302}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
```

```
  ThermoSysPro.WaterSteam.Machines.StaticCentrifugalPump
staticCentrifugalPump6 annotation(
    Placement(visible = true, transformation(origin = {178,
302}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe20 annotation(
    Placement(visible = true, transformation(origin = {130,
386}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe21 annotation(
    Placement(visible = true, transformation(origin = {130,
420}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorida colectorida11 annotation(
    Placement(visible = true, transformation(origin = {228,
404}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorret colectorret12 annotation(
    Placement(visible = true, transformation(origin = {262,
404}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorida colectorida12 annotation(
    Placement(visible = true, transformation(origin = {294,
400}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Machines.StaticCentrifugalPump
staticCentrifugalPump7 annotation(
    Placement(visible = true, transformation(origin = {178,
404}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe22 annotation(
    Placement(visible = true, transformation(origin = {130,
340}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe23 annotation(
    Placement(visible = true, transformation(origin = {130,
362}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorida colectorida13 annotation(
    Placement(visible = true, transformation(origin = {-180,
352}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe24 annotation(
    Placement(visible = true, transformation(origin = {8, 164},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Machines.StaticCentrifugalPump
staticCentrifugalPump8 annotation(
    Placement(visible = true, transformation(origin = {-52,
164}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorida colectorida14 annotation(
    Placement(visible = true, transformation(origin = {-226,
168}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorida colectorida15 annotation(
    Placement(visible = true, transformation(origin = {-220,
246}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe25 annotation(
    Placement(visible = true, transformation(origin = {8, 250},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.PressureLosses.LumpedStraightPipe
lumpedStraightPipe26 annotation(
```

```
    Placement(visible = true, transformation(origin = {8, 208},
extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Machines.StaticCentrifugalPump
staticCentrifugalPump9 annotation(
    Placement(visible = true, transformation(origin = {-52,
250}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  ThermoSysPro.WaterSteam.Machines.StaticCentrifugalPump
staticCentrifugalPump10 annotation(
    Placement(visible = true, transformation(origin = {-250,
288}, extent = {{10, -10}, {-10, 10}}, rotation = 90)));
  Bombacalor_ref_1 bombacalor_ref_11(EER_NOM = 4.5, POT_NOM =
200)  annotation(
    Placement(visible = true, transformation(origin = {332,
354}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Bombacalor_cal_1 bombacalor_cal_11(COP_NOM = 4.5, POT_CAL_NOM
= 200)  annotation(
    Placement(visible = true, transformation(origin = {334,
192}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorida colectorida16 annotation(
    Placement(visible = true, transformation(origin = {376,
358}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorret colectorret13 annotation(
    Placement(visible = true, transformation(origin = {276,
358}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Colectorida colectorida17 annotation(
    Placement(visible = true, transformation(origin = {376,
192}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Colectorret colectorret14 annotation(
    Placement(visible = true, transformation(origin = {278,
192}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Sources.Constant const(k = 273.15 + 50)
annotation(
    Placement(visible = true, transformation(origin = {104,
110}, extent = {{10, -10}, {-10, 10}}, rotation = 0)));
  Modelica.Blocks.Sources.Constant const1(k = 273.15 + 80)
annotation(
    Placement(visible = true, transformation(origin = {-288,
38}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  Modelica.Blocks.Sources.Constant const2(k = 273.15 + 80)
annotation(
    Placement(visible = true, transformation(origin = {-288, -
30}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Rampe
rampe1(Duration = 500)  annotation(
    Placement(visible = true, transformation(origin = {-288,
70}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
  ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Rampe
rampe2(Duration = 500) annotation(
    Placement(visible = true, transformation(origin = {-38,
106}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Constante
constante1(k = 55)  annotation(
    Placement(visible = true, transformation(origin = {48, 442},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));
```

```
ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Constante
constante2(k = 55)  annotation(
    Placement(visible = true, transformation(origin = {48, 404},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Constante
constante3(k = 55)  annotation(
    Placement(visible = true, transformation(origin = {52, 266},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Constante
constante4(k = 55)  annotation(
    Placement(visible = true, transformation(origin = {50, 302},
extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Constante
constante5(k = 45)  annotation(
    Placement(visible = true, transformation(origin = {302,
350}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));

ThermoSysPro.InstrumentationAndControl.Blocks.Sources.Constante
constante6(k = 120)  annotation(
    Placement(visible = true, transformation(origin = {304,
184}, extent = {{-10, -10}, {10, 10}}, rotation = 0)));
equation
  connect(constante6.y, bombacalor_cal_11.TCon) annotation(
    Line(points = {{316, 184}, {320, 184}, {320, 198}, {322,
198}}, color = {0, 0, 255}));
  connect(colectorret14.Cs6, colectorida7.Ce3) annotation(
    Line(points = {{268, 182}, {234, 182}, {234, 116}}, color =
{0, 0, 255}));
  connect(colectorret14.Cs3, colectorida8.Ce6) annotation(
    Line(points = {{268, 202}, {236, 202}, {236, 214}, {234,
214}}, color = {0, 0, 255}));
  connect(bombacalor_cal_11.SalTher, colectorret14.Ce)
annotation(
    Line(points = {{334, 198}, {334, 208}, {288, 208}, {288,
192}}, color = {0, 0, 255}));
  connect(constante5.y, bombacalor_ref_11.TEva) annotation(
    Line(points = {{314, 350}, {320, 350}, {320, 360}, {320,
360}}, color = {0, 0, 255}));
  connect(constante3.y, maquina_absorcion_11.TEva) annotation(
    Line(points = {{64, 266}, {68, 266}, {68, 296}, {86, 296},
{86, 296}}, color = {0, 0, 255}));
  connect(constante4.y, maquina_absorcion_12.TEva) annotation(
    Line(points = {{62, 302}, {66, 302}, {66, 330}, {86, 330},
{86, 330}}, color = {0, 0, 255}));
  connect(constante2.y, maquina_absorcion_14.TEva) annotation(
    Line(points = {{60, 404}, {86, 404}, {86, 398}, {86, 398}},
color = {0, 0, 255}));
  connect(constante1.y, maquina_absorcion_13.TEva) annotation(
    Line(points = {{60, 442}, {86, 442}, {86, 432}, {86, 432}},
color = {0, 0, 255}));
  connect(rampe2.y, threeWayValve1.Ouv) annotation(
```

```
    Line(points = {{-26, 106}, {-4, 106}, {-4, 88}, {-4, 88}},
color = {0, 0, 255}));
  connect(rampe1.y, threeWayValve2.Ouv) annotation(
    Line(points = {{-276, 70}, {-212, 70}, {-212, 66}, {-212,
66}}, color = {0, 0, 255}));
  connect(const2.y, hea1.TSet) annotation(
    Line(points = {{-276, -30}, {-194, -30}, {-194, -8}, {-194,
-8}}, color = {0, 0, 127}));
  connect(const1.y, hea2.TSet) annotation(
    Line(points = {{-276, 38}, {-194, 38}, {-194, 24}, {-194,
24}}, color = {0, 0, 127}));
  connect(const.y, hea.TSet) annotation(
    Line(points = {{92, 110}, {70, 110}, {70, 110}, {68, 110}},
color = {0, 0, 127}));
  connect(colectorida17.Cs, bombacalor_cal_11.EntTher)
annotation(
    Line(points = {{366, 192}, {354, 192}, {354, 180}, {334,
180}, {334, 186}, {334, 186}}, color = {0, 0, 255}));
  connect(colectorret8.Cs3, colectorida17.Ce8) annotation(
    Line(points = {{266, 116}, {374, 116}, {374, 182}, {374,
182}}, color = {0, 0, 255}));
  connect(colectorret10.Cs5, colectorida17.Ce1) annotation(
    Line(points = {{268, 220}, {374, 220}, {374, 202}, {374,
202}}, color = {0, 0, 255}));
  connect(colectorret13.Cs5, colectorida10.Ce3) annotation(
    Line(points = {{266, 354}, {234, 354}, {234, 312}, {234,
312}}, color = {0, 0, 255}));
  connect(colectorret13.Cs4, colectorida11.Ce6) annotation(
    Line(points = {{266, 362}, {238, 362}, {238, 394}, {238,
394}}, color = {0, 0, 255}));
  connect(bombacalor_ref_11.SalTher, colectorret13.Ce)
annotation(
    Line(points = {{332, 360}, {332, 360}, {332, 368}, {286,
368}, {286, 358}, {286, 358}}, color = {0, 0, 255}));
  connect(colectorida16.Cs, bombacalor_ref_11.EntTher)
annotation(
    Line(points = {{366, 358}, {346, 358}, {346, 344}, {332,
344}, {332, 348}, {332, 348}}, color = {0, 0, 255}));
  connect(colectorret11.Cs3, colectorida16.Ce8) annotation(
    Line(points = {{268, 312}, {268, 312}, {268, 332}, {374,
332}, {374, 348}, {374, 348}}, color = {0, 0, 255}));
  connect(colectorret12.Cs6, colectorida16.Ce1) annotation(
    Line(points = {{272, 394}, {272, 394}, {272, 376}, {374,
376}, {374, 368}, {374, 368}}, color = {0, 0, 255}));
  connect(staticCentrifugalPump10.C2, colectorida3.Ce3)
annotation(
    Line(points = {{-250, 278}, {-250, 278}, {-250, 96}, {-250,
96}}, color = {0, 0, 255}));
  connect(colectorida13.Cs, staticCentrifugalPump10.C1)
annotation(
    Line(points = {{-190, 352}, {-250, 352}, {-250, 298}, {-250,
298}}, color = {0, 0, 255}));
  connect(staticCentrifugalPump9.C2, colectorida15.Ce4)
annotation(
    Line(points = {{-62, 250}, {-210, 250}}, color = {0, 0,
255}));
```

```
  connect(lumpedStraightPipe25.C2, staticCentrifugalPump9.C1)
annotation(
    Line(points = {{-2, 250}, {-42, 250}}, color = {0, 0,
255}));
  connect(lumpedStraightPipe26.C2, colectorida15.Ce6)
annotation(
    Line(points = {{-2, 208}, {-18, 208}, {-18, 236}, {-210,
236}}, color = {0, 0, 255}));
  connect(colectorida15.Cs, colectorida14.Ce1) annotation(
    Line(points = {{-230, 246}, {-230, 178}, {-229, 178}}, color
= {0, 0, 255}));
  connect(colectorida14.Cs, colectorida3.Ce1) annotation(
    Line(points = {{-236, 168}, {-236, 96}}, color = {0, 0,
255}));
  connect(staticCentrifugalPump8.C2, colectorida14.Ce5)
annotation(
    Line(points = {{-62, 164}, {-216, 164}}, color = {0, 0,
255}));
  connect(staticWaterWaterExchanger3.Sc,
lumpedStraightPipe25.C1) annotation(
    Line(points = {{80, 246}, {80, 250}, {18, 250}}, color = {0,
0, 255}));
  connect(lumpedStraightPipe7.C2, staticWaterWaterExchanger3.Ec)
annotation(
    Line(points = {{18, 222}, {80, 222}, {80, 226}}, color = {0,
0, 255}));
  connect(colectorret4.Cs4, lumpedStraightPipe7.C1) annotation(
    Line(points = {{-76, 204}, {-36, 204}, {-36, 222}, {-2,
222}}, color = {0, 0, 255}));
  connect(staticWaterWaterExchanger4.Sc,
lumpedStraightPipe26.C1) annotation(
    Line(points = {{80, 204}, {80, 204}, {80, 208}, {18, 208},
{18, 208}}, color = {0, 0, 255}));
  connect(lumpedStraightPipe24.C2, staticCentrifugalPump8.C1)
annotation(
    Line(points = {{-2, 164}, {-42, 164}, {-42, 164}, {-42,
164}}, color = {0, 0, 255}));
  connect(staticWaterWaterExchanger2.Sc,
lumpedStraightPipe24.C1) annotation(
    Line(points = {{80, 162}, {80, 162}, {80, 164}, {18, 164},
{18, 164}}, color = {0, 0, 255}));
  connect(lumpedStraightPipe23.C2, colectorida13.Ce3)
annotation(
    Line(points = {{120, 362}, {-170, 362}}, color = {0, 0,
255}));
  connect(lumpedStraightPipe22.C2, colectorida13.Ce6)
annotation(
    Line(points = {{120, 340}, {-170, 340}, {-170, 342}}, color
= {0, 0, 255}));
  connect(staticCentrifugalPump6.C2, lumpedStraightPipe22.C1)
annotation(
    Line(points = {{168, 302}, {160, 302}, {160, 340}, {140,
340}, {140, 340}}, color = {0, 0, 255}));
  connect(staticCentrifugalPump7.C2, lumpedStraightPipe23.C1)
annotation(
```

```
    Line(points = {{168, 404}, {158, 404}, {158, 362}, {140,
362}, {140, 362}}, color = {0, 0, 255}));
  connect(colectorida12.Cs, colectorret12.Ce) annotation(
    Line(points = {{284, 400}, {278, 400}, {278, 390}, {252,
390}, {252, 404}}, color = {0, 0, 255}));
  connect(colectorida11.Cs, staticCentrifugalPump7.C1)
annotation(
    Line(points = {{218, 404}, {188, 404}}, color = {0, 0,
255}));
  connect(lumpedStraightPipe20.C2, colectorida12.Ce8)
annotation(
    Line(points = {{140, 386}, {292, 386}, {292, 390}}, color =
{0, 0, 255}));
  connect(maquina_absorcion_14.SalTher, lumpedStraightPipe20.C1)
annotation(
    Line(points = {{86, 386}, {120, 386}}, color = {0, 0,
255}));
  connect(lumpedStraightPipe10.C2, maquina_absorcion_14.EntTher)
annotation(
    Line(points = {{16, 386}, {74, 386}}, color = {0, 0, 255}));
  connect(colectorret7.Cs5, lumpedStraightPipe10.C1) annotation(
    Line(points = {{-116, 398}, {-38, 398}, {-38, 386}, {-4,
386}}, color = {0, 0, 255}));
  connect(lumpedStraightPipe21.C2, colectorida12.Ce1)
annotation(
    Line(points = {{140, 420}, {290, 420}, {290, 410}, {292,
410}}, color = {0, 0, 255}));
  connect(maquina_absorcion_13.SalTher, lumpedStraightPipe21.C1)
annotation(
    Line(points = {{86, 420}, {120, 420}}, color = {0, 0,
255}));
  connect(lumpedStraightPipe11.C2, maquina_absorcion_13.EntTher)
annotation(
    Line(points = {{16, 420}, {74, 420}}, color = {0, 0, 255}));
  connect(colectorret7.Cs4, lumpedStraightPipe11.C1) annotation(
    Line(points = {{-116, 406}, {-38, 406}, {-38, 420}, {-4,
420}}, color = {0, 0, 255}));
  connect(colectorida9.Cs, colectorret11.Ce) annotation(
    Line(points = {{284, 300}, {278, 300}, {278, 288}, {248,
288}, {248, 302}}, color = {0, 0, 255}));
  connect(colectorida10.Cs, staticCentrifugalPump6.C1)
annotation(
    Line(points = {{214, 302}, {188, 302}}, color = {0, 0,
255}));
  connect(lumpedStraightPipe18.C2, colectorida9.Ce8) annotation(
    Line(points = {{142, 284}, {290, 284}, {290, 290}, {292,
290}}, color = {0, 0, 255}));
  connect(maquina_absorcion_11.SalTher, lumpedStraightPipe18.C1)
annotation(
    Line(points = {{86, 284}, {122, 284}}, color = {0, 0,
255}));
  connect(lumpedStraightPipe8.C2, maquina_absorcion_11.EntTher)
annotation(
    Line(points = {{16, 284}, {74, 284}}, color = {0, 0, 255}));
  connect(colectorret6.Cs5, lumpedStraightPipe8.C1) annotation(
```

```
    Line(points = {{-72, 294}, {-38, 294}, {-38, 284}, {-4,
284}}, color = {0, 0, 255}));
  connect(lumpedStraightPipe9.C2, maquina_absorcion_12.EntTher)
annotation(
    Line(points = {{16, 318}, {74, 318}}, color = {0, 0, 255}));
  connect(colectorret6.Cs4, lumpedStraightPipe9.C1) annotation(
    Line(points = {{-72, 302}, {-38, 302}, {-38, 318}, {-4,
318}}, color = {0, 0, 255}));
  connect(maquina_absorcion_12.SalTher, lumpedStraightPipe19.C1)
annotation(
    Line(points = {{86, 318}, {120, 318}}, color = {0, 0,
255}));
  connect(lumpedStraightPipe19.C2, colectorida9.Ce1) annotation(
    Line(points = {{140, 318}, {291, 318}, {291, 310}}, color =
{0, 0, 255}));
  connect(colectorret2.Cs6, lumpedStraightPipe5.C1) annotation(
    Line(points = {{-134, 134}, {-2, 134}}, color = {0, 0,
255}));
  connect(lumpedStraightPipe5.C2, staticWaterWaterExchanger2.Ec)
annotation(
    Line(points = {{18, 134}, {80, 134}, {80, 142}}, color = {0,
0, 255}));
  connect(staticCentrifugalPump5.C2, lumpedStraightPipe17.C1)
annotation(
    Line(points = {{168, 224}, {142, 224}, {142, 224}, {142,
224}}, color = {0, 0, 255}));
  connect(lumpedStraightPipe17.C2,
staticWaterWaterExchanger3.Ef) annotation(
    Line(points = {{122, 224}, {108, 224}, {108, 232}, {86,
232}}, color = {0, 0, 255}));
  connect(lumpedStraightPipe16.C2, colectorret10.Ce) annotation(
    Line(points = {{142, 250}, {248, 250}, {248, 224}}, color =
{0, 0, 255}));
  connect(colectorida8.Cs, staticCentrifugalPump5.C1)
annotation(
    Line(points = {{214, 224}, {188, 224}}, color = {0, 0,
255}));
  connect(colectorret9.Cs8, staticCentrifugalPump3.C1)
annotation(
    Line(points = {{198, 96}, {198, 38}, {186, 38}}, color = {0,
0, 255}));
  connect(staticCentrifugalPump3.C2, lumpedStraightPipe14.C1)
annotation(
    Line(points = {{166, 38}, {142, 38}}, color = {0, 0, 255}));
  connect(colectorret9.Cs1, staticCentrifugalPump4.C1)
annotation(
    Line(points = {{198, 116}, {196, 116}, {196, 140}, {188,
140}, {188, 140}}, color = {0, 0, 255}));
  connect(staticCentrifugalPump4.C2, lumpedStraightPipe15.C1)
annotation(
    Line(points = {{168, 140}, {142, 140}}, color = {0, 0,
255}));
  connect(colectorida7.Cs, colectorret9.Ce) annotation(
    Line(points = {{214, 106}, {204, 106}, {204, 106}, {204,
106}}, color = {0, 0, 255}));
```

```
  connect(staticWaterWaterExchanger3.Sf,
lumpedStraightPipe16.C1) annotation(
    Line(points = {{86, 242}, {108, 242}, {108, 250}, {122,
250}, {122, 250}}, color = {0, 0, 255}));
  connect(staticWaterWaterExchanger4.Sf, sinkP1.C) annotation(
    Line(points = {{86, 200}, {98, 200}, {98, 204}, {112, 204},
{112, 204}}, color = {0, 0, 255}));
  connect(sourceP1.C, staticWaterWaterExchanger4.Ef) annotation(
    Line(points = {{112, 186}, {100, 186}, {100, 190}, {86,
190}, {86, 190}}, color = {0, 0, 255}));
  connect(lumpedStraightPipe15.C2,
staticWaterWaterExchanger2.Ef) annotation(
    Line(points = {{122, 140}, {102, 140}, {102, 148}, {86,
148}, {86, 148}}, color = {0, 0, 255}));
  connect(lumpedStraightPipe14.C2,
staticWaterWaterExchanger1.Ef) annotation(
    Line(points = {{122, 38}, {102, 38}, {102, 46}, {86, 46},
{86, 46}}, color = {0, 0, 255}));
  connect(lumpedStraightPipe12.C2, colectorida6.Ce6) annotation(
    Line(points = {{142, 66}, {302, 66}, {302, 72}, {302, 72}},
color = {0, 0, 255}));
  connect(colectorida6.Cs, colectorret8.Ce) annotation(
    Line(points = {{282, 82}, {246, 82}, {246, 106}}, color =
{0, 0, 255}));
  connect(lumpedStraightPipe13.C2, colectorida6.Ce3) annotation(
    Line(points = {{142, 166}, {302, 166}, {302, 92}}, color =
{0, 0, 255}));
  connect(staticWaterWaterExchanger2.Sf,
lumpedStraightPipe13.C1) annotation(
    Line(points = {{86, 158}, {103, 158}, {103, 166}, {122,
166}}, color = {0, 0, 255}));
  connect(staticWaterWaterExchanger1.Sf,
lumpedStraightPipe12.C1) annotation(
    Line(points = {{86, 58}, {103, 58}, {103, 66}, {122, 66}},
color = {0, 0, 255}));
  connect(colectorret5.Cs1, colectorret7.Ce) annotation(
    Line(points = {{-134, 304}, {-136, 304}, {-136, 402}}, color
= {0, 0, 255}));
  connect(colectorret5.Cs4, colectorret6.Ce) annotation(
    Line(points = {{-122, 298}, {-92, 298}}, color = {0, 0,
255}));
  connect(colectorret3.Cs1, colectorret5.Ce) annotation(
    Line(points = {{-142, 206}, {-142, 294}}, color = {0, 0,
255}));
  connect(lumpedStraightPipe6.C2, staticWaterWaterExchanger4.Ec)
annotation(
    Line(points = {{18, 182}, {80, 182}, {80, 184}, {80, 184}},
color = {0, 0, 255}));
  connect(colectorret4.Cs5, lumpedStraightPipe6.C1) annotation(
    Line(points = {{-76, 196}, {-36, 196}, {-36, 182}, {-2,
182}, {-2, 182}}, color = {0, 0, 255}));
  connect(colectorret3.Cs4, colectorret4.Ce) annotation(
    Line(points = {{-128, 200}, {-96, 200}, {-96, 200}, {-96,
200}}, color = {0, 0, 255}));
  connect(colectorret2.Cs1, colectorret3.Ce) annotation(
```

```
    Line(points = {{-148, 154}, {-148, 196}}, color = {0, 0,
255}));
  connect(colectorida2.Cs, colectorret2.Ce) annotation(
    Line(points = {{-200, 86}, {-200, 144}, {-154, 144}}, color
= {0, 0, 255}));
  connect(colectorida5.Cs, staticCentrifugalPump2.C1)
annotation(
    Line(points = {{-174, 54}, {-170, 54}, {-170, 112}, {-152,
112}, {-152, 112}}, color = {0, 0, 255}));
  connect(colectorida4.Cs, colectorida5.Ce8) annotation(
    Line(points = {{-114, 4}, {-112, 4}, {-112, 40}, {-182, 40},
{-182, 44}, {-180, 44}}, color = {0, 0, 255}));
  connect(threeWayValve2.C2, colectorida5.Ce5) annotation(
    Line(points = {{-202, 50}, {-194, 50}}, color = {255, 0,
0}));
  connect(staticCentrifugalPump2.C2, lumpedStraightPipe4.C1)
annotation(
    Line(points = {{-132, 112}, {-124, 112}}, color = {0, 0,
255}));
  connect(lumpedStraightPipe4.C2,
staticExchangerWaterSteamFlueGases1.Cws1) annotation(
    Line(points = {{-104, 112}, {-100, 112}, {-100, 82}, {-108,
82}}, color = {0, 0, 255}));
  connect(adapWaterMT3.SalTher, colectorida4.Ce6) annotation(
    Line(points = {{-144, -14}, {-134, -14}, {-134, -6}, {-134,
-6}}, color = {0, 0, 255}));
  connect(adapWaterMT2.SalTher, colectorida4.Ce3) annotation(
    Line(points = {{-144, 18}, {-134, 18}, {-134, 14}, {-134,
14}}, color = {0, 0, 255}));
  connect(hea2.port_b, adapWaterMT2.EntMod) annotation(
    Line(points = {{-172, 18}, {-164, 18}, {-164, 18}, {-164,
18}}, color = {0, 127, 255}));
  connect(hea1.port_b, adapWaterMT3.EntMod) annotation(
    Line(points = {{-172, -14}, {-164, -14}, {-164, -14}, {-164,
-14}}, color = {0, 127, 255}));
  connect(colectorret1.Cs5, adapWaterTM3.EntTher) annotation(
    Line(points = {{-234, 0}, {-228, 0}, {-228, -14}, {-222, -
14}, {-222, -14}}, color = {0, 0, 255}));
  connect(colectorret1.Cs4, adapWaterTM2.EntTher) annotation(
    Line(points = {{-234, 8}, {-226, 8}, {-226, 18}, {-222, 18},
{-222, 18}}, color = {0, 0, 255}));
  connect(threeWayValve2.C3, colectorret1.Ce) annotation(
    Line(points = {{-212, 44}, {-212, 44}, {-212, 30}, {-254,
30}, {-254, 4}, {-254, 4}}, color = {255, 0, 0}));
  connect(adapWaterTM3.SalMod, hea1.port_a) annotation(
    Line(points = {{-202, -14}, {-192, -14}, {-192, -14}, {-192,
-14}}, color = {0, 127, 255}));
  connect(adapWaterTM2.SalMod, hea2.port_a) annotation(
    Line(points = {{-202, 18}, {-192, 18}, {-192, 18}, {-192,
18}}, color = {0, 127, 255}));
  connect(colectorida3.Cs, threeWayValve2.C1) annotation(
    Line(points = {{-230, 86}, {-226, 86}, {-226, 50}, {-222,
50}, {-222, 50}}, color = {0, 0, 255}));
  connect(lumpedStraightPipe3.C2, colectorida2.Ce5) annotation(
    Line(points = {{-164, 82}, {-180, 82}}, color = {0, 0,
255}));
```

```
  connect(staticExchangerWaterSteamFlueGases1.Cws2,
lumpedStraightPipe3.C1) annotation(
    Line(points = {{-128, 82}, {-144, 82}}, color = {0, 0,
255}));
  connect(staticExchangerWaterSteamFlueGases1.Cfg2, sink1.C)
annotation(
    Line(points = {{-118, 73}, {-118, 66}}));
  connect(alternatingEngine1.Cfg,
staticExchangerWaterSteamFlueGases1.Cfg1) annotation(
    Line(points = {{-46, 42}, {-46, 58}, {-88, 58}, {-88, 91},
{-118, 91}}));
  connect(fuelSourcePQ1.C, alternatingEngine1.Cfuel) annotation(
    Line(points = {{-50, 8}, {-50, 8}, {-50, 22}, {-50, 22}}));
  connect(sourcePQ1.C, alternatingEngine1.Cair) annotation(
    Line(points = {{-42, 8}, {-42, 8}, {-42, 22}, {-42, 22}}));
  connect(refP1.C2, lumpedStraightPipe2.C1) annotation(
    Line(points = {{-96, 42}, {-96, 42}, {-96, 32}, {-88, 32},
{-88, 32}}, color = {0, 0, 255}));
  connect(loopBreakerQ1.C2, refP1.C1) annotation(
    Line(points = {{-80, 72}, {-96, 72}, {-96, 62}, {-96, 62}},
color = {0, 0, 255}));
  connect(colectorida1.Cs, loopBreakerQ1.C1) annotation(
    Line(points = {{-46, 72}, {-60, 72}, {-60, 72}, {-60, 72}},
color = {0, 0, 255}));
  connect(lumpedStraightPipe2.C2, alternatingEngine1.Cws1)
annotation(
    Line(points = {{-68, 32}, {-56, 32}, {-56, 32}, {-56, 32}},
color = {0, 0, 255}));
  connect(threeWayValve1.C3, colectorida1.Ce5) annotation(
    Line(points = {{-4, 66}, {-26, 66}, {-26, 68}, {-26, 68}},
color = {255, 0, 0}));
  connect(threeWayValve1.C2, colectorida1.Ce4) annotation(
    Line(points = {{-14, 72}, {-20, 72}, {-20, 76}, {-26, 76},
{-26, 76}}, color = {255, 0, 0}));
  connect(adapWaterMT1.SalTher, threeWayValve1.C1) annotation(
    Line(points = {{16, 104}, {14, 104}, {14, 72}, {6, 72}, {6,
72}}, color = {0, 0, 255}));
  connect(hea.port_b, adapWaterMT1.EntMod) annotation(
    Line(points = {{46, 104}, {36, 104}}, color = {0, 127,
255}));
  connect(adapWaterTM1.SalMod, hea.port_a) annotation(
    Line(points = {{80, 94}, {80, 104}, {66, 104}}, color = {0,
127, 255}));
  connect(staticWaterWaterExchanger1.Sc, adapWaterTM1.EntTher)
annotation(
    Line(points = {{80, 62}, {80, 74}}, color = {0, 0, 255}));
  connect(lumpedStraightPipe1.C2, staticWaterWaterExchanger1.Ec)
annotation(
    Line(points = {{36, 32}, {80, 32}, {80, 42}}, color = {0, 0,
255}));
  connect(staticCentrifugalPump1.C2, lumpedStraightPipe1.C1)
annotation(
    Line(points = {{2, 32}, {16, 32}, {16, 32}, {16, 32}}, color
= {0, 0, 255}));
  connect(alternatingEngine1.Cws2, staticCentrifugalPump1.C1)
annotation(
```

```
    Line(points = {{-36, 32}, {-18, 32}, {-18, 32}, {-18, 32}},
color = {0, 0, 255}));
  annotation(
    Diagram(coordinateSystem(extent = {{-1000, -1000}, {1000,
1000}})),
    Icon(coordinateSystem(extent = {{-1000, -1000}, {1000,
1000}})),
    version = "",
    uses(ThermoSysPro(version = "3.1"), Buildings(version =
"4.0.0"), Modelica(version = "3.2.2")),
    __OpenModelica_commandLineOptions = "");

end Planta_cogeneracion;
```