



UNIVERSIDAD DE TALCA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA CIVIL EN COMPUTACIÓN

Documento de especificación de casos de prueba

**HERRAMIENTA PARA LA OPTIMIZACIÓN DE REDES DE
DISTRIBUCIÓN DE AGUA POTABLE**

EQUIPO DE DESARROLLO:

Nombre	Rol	Contacto
Gabriel Sanhueza Fuentes	Administrador, Analista, Diseñador, Implementador y Tester.	gsanhueza15@alumnos.uta.cl

CONTRAPARTE:

Nombre	Rol	Contacto
Jimmy H. Gutiérrez-Bahamondes	Cliente/Profesor guía	
Jimmy H. Gutiérrez-Bahamondes	Cliente/Profesor co-guía	

HISTORIAL DE CAMBIOS

Version	Fecha	Modificaciones
0.1	07/09/2019	
1	15/06/2020	

TABLA DE CONTENIDOS

página

Tabla de Contenidos	I
1. Casos de pruebas automatizadas	2
1.1. <i>Pump</i>	2
1.2. <i>GeneticAlgorithm</i>	8
1.3. <i>IntegerRangeRandomMutation</i>	10
1.4. <i>ReflectionUtils</i>	13
1.5. <i>ResultSimulation</i>	22
1.6. <i>JsonSimpleReader</i>	24
2. Casos de pruebas manuales	32

1. Casos de pruebas automatizadas

En este capítulo se presenta la especificación formal de los casos de prueba automatizados.

1.1. *Pump*

En esta sección se presentan las pruebas realizadas para la clase *Pump*.

Test ID:	AT001
Título:	Envío de parámetro inválido al método <code>setProperty</code> de la clase <i>Pump</i> cuando se usa la clave <i>PumpProperty.HEAD</i> .
Característica:	Validar parámetro recibido por el método <i>setProperty</i> .
Objetivo:	Comprobar que si se pasa un parámetro como valor de un tipo distinto a <i>String</i> , cuando se usa la clave <i>PumpProperty.HEAD</i> , el método lanza una excepción.
Configuración:	Instancia de la clase <i>Pump</i> inicializada.
Datos de prueba:	Clave: <i>PumpProperty.HEAD</i> Value: Un objeto diferente a un <i>String</i>
Acciones de prueba:	1. Inicializar instancia. 2. Pasar una instancia de un objeto distinto de un <i>String</i> al método cuando se usa la llave <i>PumpProperty.HEAD</i> .
Resultados esperados:	Excepción indicando que el tipo de instancia pasada no es válida.

Test ID:	AT002
Título:	Envío de parámetro invalido al método <code>setProperty</code> de la clase <i>Pump</i> cuando se usa la clave <i>PumpProperty.PATTERN</i> .
Característica:	Validar parametro recibido por el método <i>setProperty</i> .
Objetivo:	Comprobar que si se pasa un parametro como valor de un tipo distinto a <i>String</i> , cuando se usa la clave <i>PumpProperty.PATTERN</i> , el método lanza una excepción.
Configuración:	Instancia de la clase <i>Pump</i> inicializada.
Datos de prueba:	Clave: <i>PumpProperty.PATTERN</i> Value: Un objeto diferente a un <i>String</i>
Acciones de prueba:	1. Inicializar instancia. 2. Pasar una instancia de un objeto distinto de un <i>String</i> al método cuando se usa la llave <i>PumpProperty.HEAD</i> .
Resultados esperados:	Excepción indicando que el tipo de instancia pasada no es valida.

Test ID:	AT003
Título:	Envío de parámetro invalido al método <code>setProperty</code> de la clase <i>Pump</i> cuando se usa la clave <i>PumpProperty.SPEED</i> .
Característica:	Validar parametro recibido por el método <i>setProperty</i> .
Objetivo:	Comprobar que si se pasa un parametro como valor de un tipo distinto a <i>Double</i> , cuando se usa la clave <i>PumpProperty.SPEED</i> , el método lanza una excepción.
Configuración:	Instancia de la clase <i>Pump</i> inicializada.
Datos de prueba:	Clave: <i>PumpProperty.SPEED</i> Value: Un <i>Integer</i> o alguna instancia de otro objeto distinto de <i>Double</i> .
Acciones de prueba:	1. Inicializar instancia. 2. Pasar una instancia de un objeto distinto de un <i>Double</i> al método cuando se usa la llave <i>PumpProperty.SPEED</i> .
Resultados esperados:	Excepción indicando que el tipo de instancia pasada no es valida.

Test ID:	AT004
Título:	Envío de parámetro invalido al método <code>setProperty</code> de la clase <i>Pump</i> cuando se usa la clave <i>PumpProperty.POWER</i> .
Característica:	Validar parametro recibido por el método <i>setProperty</i> .
Objetivo:	Comprobar que si se pasa un parametro como valor de un tipo distinto a <i>Double</i> , cuando se usa la clave <i>PumpProperty.POWER</i> , el método lanza una excepción.
Configuración:	Instancia de la clase <i>Pump</i> inicializada.
Datos de prueba:	Clave: <i>PumpProperty.POWER</i> Value: Un <i>Integer</i> o alguna instancia de otro objeto distinto de <i>Double</i> .
Acciones de prueba:	1. Inicializar instancia. 2. Pasar una instancia de un objeto distinto de un <i>Double</i> al método cuando se usa la llave <i>PumpProperty.POWER</i> .
Resultados esperados:	Excepción indicando que el tipo de instancia pasada no es valida.

Test ID:	AT005
Título:	Envío de parámetro válido al método <code>setProperty</code> de la clase <i>Pump</i> cuando se usa la clave <i>PumpProperty.HEAD</i> .
Característica:	Validar parametro recibido por el método <i>setProperty</i> .
Objetivo:	Comprobar que si se pasa un parametro de tipo <i>String</i> como valor, cuando se usa la clave <i>PumpProperty.HEAD</i> , el método finaliza sin error.
Configuración:	Instancia de la clase <i>Pump</i> inicializada.
Datos de prueba:	Clave: <i>PumpProperty.HEAD</i> Value: Un <i>String</i> no vacío.
Acciones de prueba:	1. Inicializar instancia. 2. Pasar una instancia de un <i>String</i> al método cuando se usa la llave <i>PumpProperty.HEAD</i> .
Resultados esperados:	Método ejecutado sin errores.

Test ID:	AT006
Título:	Envío de parámetro válido al método <code>setProperty</code> de la clase <i>Pump</i> cuando se usa la clave <i>PumpProperty.PATTERN</i> .
Característica:	Validar parametro recibido por el método <i>setProperty</i> .
Objetivo:	Comprobar que si se pasa un parametro de tipo <i>String</i> como valor, cuando se usa la clave <i>PumpProperty.PATTERN</i> , el método finaliza sin error.
Configuración:	Instancia de la clase <i>Pump</i> inicializada.
Datos de prueba:	Clave: <i>PumpProperty.PATTERN</i> Value: Un <i>String</i> no vacío.
Acciones de prueba:	1. Inicializar instancia. 2. Pasar una instancia de un <i>String</i> al método cuando se usa la llave <i>PumpProperty.HEAD</i> .
Resultados esperados:	Método ejecutado sin errores.

Test ID:	AT007
Título:	Envío de parámetro válido al método <code>setProperty</code> de la clase <i>Pump</i> cuando se usa la clave <i>PumpProperty.SPEED</i> .
Característica:	Validar parametro recibido por el método <i>setProperty</i> .
Objetivo:	Comprobar que si se pasa un parametro de tipo <i>Double</i> como valor, cuando se usa la clave <i>PumpProperty.SPEED</i> , el método finaliza sin error.
Configuración:	Instancia de la clase <i>Pump</i> inicializada.
Datos de prueba:	Clave: <i>PumpProperty.SPEED</i> Value: Un valor <i>Double</i> .
Acciones de prueba:	1. Inicializar instancia. 2. Pasar una instancia de un <i>Double</i> al método cuando se usa la llave <i>PumpProperty.SPEED</i> .
Resultados esperados:	Método ejecutado sin errores.

Test ID:	AT008
Título:	Envío de parámetro válido al método <code>setProperty</code> de la clase <i>Pump</i> cuando se usa la clave <i>PumpProperty.POWER</i> .
Característica:	Validar parametro recibido por el método <i>setProperty</i> .
Objetivo:	Comprobar que si se pasa un parametro de tipo <i>Double</i> como valor, cuando se usa la clave <i>PumpProperty.POWER</i> , el método finaliza sin error.
Configuración:	Instancia de la clase <i>Pump</i> inicializada.
Datos de prueba:	Clave: <i>PumpProperty.POWER</i> Value: Un valor <i>Double</i> .
Acciones de prueba:	1. Inicializar instancia. 2. Pasar una instancia de un <i>Double</i> al método cuando se usa la llave <i>PumpProperty.SPEED</i> .
Resultados esperados:	Método ejecutado sin errores.

1.2. *GeneticAlgorithm*

En esta sección se especifican las pruebas realizadas para la clase *GeneticAlgorithm*.

Test ID:	AT009
Título:	Número máximo de evaluaciones no válido.
Característica:	Validar parámetro <i>maxEvaluations</i> .
Objetivo:	Validar que el parámetro <i>maxEvaluations</i> no sea negativo. Si el parámetro es negativo debe lanzarse una excepción.
Configuración:	Instancia de la clase <i>GeneticAlgorithm</i> inicializada.
Datos de prueba:	Cualquier entero menor que 0.
Acciones de prueba:	Llamar al método <i>setMaxEvaluations</i> con un argumento negativo.
Resultados esperados:	Una excepción

Test ID:	AT010
Título:	Número máximo de evaluaciones sin mejora no válido.
Característica:	Validar parámetro <i>maxNumberOfEvaluationsWithoutImprovement</i> .
Objetivo:	Validar que el parámetro <i>maxNumberOfEvaluationsWithoutImprovement</i> no sea negativo. Si el parámetro es negativo debe lanzarse una excepción.
Configuración:	Instancia de la clase <i>GeneticAlgorithm</i> inicializada.
Datos de prueba:	Cualquier entero menor que 0.
Acciones de prueba:	Llamar al método <i>setMaxNumberOfEvaluationsWithoutImprovement</i> con un argumento negativo.
Resultados esperados:	Una excepción

Test ID:	AT011
Título:	Deshabilitar número máximo de evaluaciones sin mejoras cuando se modifica el numero máximo de evaluaciones.
Característica:	Validar parámetro <i>maxEvaluations</i> .
Objetivo:	Validar que al modificar el parámetro <i>maxEvaluations</i> el parámetro <i>maxNumberOfEvaluationsWithoutImprovement</i> cambie a 0 (0 indica que esta deshabilitado).
Configuración:	Instancia de la clase <i>GeneticAlgorithm</i> inicializada.
Datos de prueba:	Cualquier entero positivo.
Acciones de prueba:	Llamar al método <i>setMaxEvaluations</i> con un entero positivo mayor a 0.
Resultados esperados:	Parámetro <i>maxNumberOfEvaluationsWithoutImprovement</i> igual a 0

Test ID:	AT012
Título:	Deshabilitar número máximo de evaluaciones cuando se modifica el numero máximo de evaluaciones sin mejoras.
Característica:	Validar parámetro <i>maxNumberOfEvaluationsWithoutImprovement</i> .
Objetivo:	Validar que al modificar el parámetro <i>maxNumberOfEvaluationsWithoutImprovement</i> el parámetro <i>maxNumberOfEvaluationsWithoutImprovement</i> cambie a 0 (0 indica que esta deshabilitado).
Configuración:	Instancia de la clase <i>GeneticAlgorithm</i> inicializada.
Datos de prueba:	Cualquier entero positivo.
Acciones de prueba:	Llamar al método <i>setMaxNumberOfEvaluationsWithoutImprovement</i> con un entero positivo mayor a 0.
Resultados esperados:	Parámetro <i>maxEvaluations</i> igual a 0

1.3. *IntegerRangeRandomMutation*

En esta sección se especifican las pruebas realizadas sobre la clase *IntegerRangeRandomMutation*.

Test ID:	AT013
Título:	Probabilidad de mutación no valida.
Característica:	Validar parámetro <i>mutationProbability</i> .
Objetivo:	Validar que el parámetro <i>mutationProbability</i> no sea negativo. Si el parámetro es negativo debe lanzarse una excepción.
Configuración:	Instancia de la clase <i>IntegerRangeRandomMutation</i> inicializada.
Datos de prueba:	Cualquier entero menor que 0.
Acciones de prueba:	Crear instancia de la clase <i>IntegerRangeRandomMutation</i> pasando como argumento para el parámetro <i>mutationProbability</i> un valor negativo.
Resultados esperados:	Una excepción

Test ID:	AT014
Título:	Rango no valido.
Característica:	Validar parámetro <i>range</i> .
Objetivo:	Validar que el parámetro <i>range</i> no sea negativo. Si el parámetro es negativo debe lanzarse una excepción.
Configuración:	Instancia de la clase <i>IntegerRangeRandomMutation</i> inicializada.
Datos de prueba:	Cualquier entero menor que 0.
Acciones de prueba:	Crear instancia de la clase <i>IntegerRangeRandomMutation</i> pasando como argumento para el parámetro <i>range</i> un valor negativo.
Resultados esperados:	Una excepción

Test ID:	AT015
Título:	Mutar variables.
Característica:	Mutar variables cuando el número generado por el <i>randomGenerator</i> es menor que la probabilidad de mutación.
Objetivo:	Validar que la mutación suceda cuando un número generado aleatoriamente sea menor que la probabilidad de mutación.
Configuración:	Solucion con variables preestablecidas. Operador <i>IntegerRangeRandomMutation</i> inicializado.
Datos de prueba:	<i>mutationProbability</i> : 0.3 <i>range</i> : 2 <i>solution</i> : [0 ,2, 4] <i>randomGenerator</i> : [0.2, 0.2, 0.4] <i>boundedRandomGenerator</i> : [2, 0, 5]
Acciones de prueba:	Pasar la solución al método <i>execute</i> .
Resultados esperados:	Las variables en la solución despues de realizar la mutación debe ser [2, 0, 4]

Test ID:	AT016
Título:	No mutar variables.
Característica:	No mutar variables cuando los números generado por el <i>randomGenerator</i> sean mayores que la probabilidad de mutación.
Objetivo:	Validar que la mutación no sucede si el <i>randomGenerator</i> devuelve valores mayores a <i>mutationProbability</i> .
Configuración:	Solucion con variables preestablecidas. Operador <i>IntegerRangeRandomMutation</i> inicializado.
Datos de prueba:	<i>mutationProbability</i> : 0.3 <i>range</i> : 2 <i>solution</i> : [0 ,2, 4] <i>randomGenerator</i> : [0.5, 0.4, 0.4] <i>boundedRandomGenerator</i> : [2, 0, 5]
Acciones de prueba:	Pasar la solución al método <i>execute</i> .
Resultados esperados:	Las variables en la solución despues de realizar la llamada al método <i>execute</i> debe ser los valores originales [0, 2, 4].

Test ID:	AT017
Título:	No mutar variables cuando no hay un rango de mutación.
Característica:	No mutar variables cuando el rango de las variables a generar es 0.
Objetivo:	Validar que la mutación no sucede si el <i>range</i> tiene el valor 0.
Configuración:	Solucion con variables preestablecidas. Operador <i>IntegerRangeRandomMutation</i> inicializado.
Datos de prueba:	<i>mutationProbability</i> : 0.3 <i>range</i> : 0 <i>solution</i> : [0 ,2, 4] <i>randomGenerator</i> : [0.1, 0.2, 0.3]
Acciones de prueba:	Pasar la solución al método <i>execute</i> .
Resultados esperados:	Las variables en la solución despues de realizar la llamada al método <i>execute</i> debe ser los valores originales [0, 2, 4].

1.4. *ReflectionUtils*

En esta sección se especifican las pruebas realizadas sobre la clase *ReflectionUtils*.

Durante esta sección nos referiremos a cualquier implementación de las interfaz *Registrable* o sus subinterfaces unicamente como *Registrable*. Es por ello, que cuando se mencione implementar *Registrable* se refiere a implementar ya sea *SingleObjectiveRegistrable* o *MultiobjectiveRegistrable*.

Test ID:	AT018
Título:	Nombre del problema en anotación <i>@NewProblem</i> .
Característica:	Obtener nombre del problema de la anotación <i>@NewProblem</i> .
Objetivo:	Validar que el método <i>getNameOfProblem</i> retorna el nombre asignado en la anotación <i>@NewProblem</i> .
Configuración:	Implementación de <i>Registrable</i> con la anotación <i>@NewProblem</i> en su constructor.
Datos de prueba:	Nombre del problema: "Test"
Acciones de prueba:	Pasar el objeto <i>Class<?></i> que hace referencia al tipo <i>Registrable</i> al método <i>getNameOfProblem</i> .
Resultados esperados:	"Test"

Test ID:	AT019
Título:	Nombre del algoritmo en anotación <i>@NewProblem</i> .
Característica:	Obtener nombre del algoritmo de la anotación <i>@NewProblem</i> .
Objetivo:	Validar que el método <i>getNameOfAlgorithm</i> retorna el nombre asignado en la anotación <i>@NewProblem</i> .
Configuración:	Implementación <i>Registrable</i> con la anotación <i>@NewProblem</i> en su constructor.
Datos de prueba:	Nombre del algoritmo: "NSGAI"
Acciones de prueba:	Pasar el objeto <i>Class<?></i> que hace referencia al tipo <i>Registrable</i> al método <i>getNameOfAlgorithm</i> .
Resultados esperados:	"NSGAI"

Test ID:	AT020
Título:	<i>Registrable</i> sin anotaciones.
Característica:	Validar las anotaciones y el tipo de parámetros utilizados en el constructor de la clase <i>Registrable</i> .
Objetivo:	Validar que si en el constructor público no tiene ni la anotación <i>@NewProblem</i> ni la anotación <i>Parameters</i> se lanza una excepción.
Configuración:	Implementación de <i>Registrable</i> sin anotaciones.
Datos de prueba:	Objeto <i>Class<?></i> que referencia al tipo <i>Registrable</i> .
Acciones de prueba:	Pasar el objeto <i>Class</i> que hace referencia al tipo <i>Registrable</i> al método <i>validateRegistrableProblem</i> .
Resultados esperados:	Una excepción.

Test ID:	AT021
Título:	<i>Registrable</i> con la anotación <i>@NewProblem</i> .
Característica:	Validar las anotaciones y el tipo de parámetros utilizados en el constructor de la clase <i>Registrable</i> .
Objetivo:	Validar si en el constructor público tiene ni la anotación <i>@NewProblem</i>
Configuración:	Implementación de <i>Registrable</i> con la anotación <i>@NewProblem</i> .
Datos de prueba:	Objeto <i>Class<?></i> que referencia al tipo <i>Registrable</i> .
Acciones de prueba:	Pasar el objeto <i>Class</i> que hace referencia al tipo <i>Registrable</i> al método <i>validateRegistrableProblem</i> .
Resultados esperados:	Método ejecutado sin errores.

Test ID:	AT022
Título:	<i>Registrable</i> cuyo constructor recibe los parámetros en el orden correcto dependiendo de su tipo.
Característica:	Validar las anotaciones y el tipo de parámetros utilizados en el constructor de la clase <i>Registrable</i> .
Objetivo:	Validar que el constructor público con las anotaciones recibe los parametros en el siguiente orden: <i>Object</i> , <i>File</i> , (<i>int</i> <i>Integer</i> <i>double</i> <i>Double</i>).
Configuración:	Implementación de <i>Registrable</i> con las anotaciones <i>@New-Problem</i> y <i>@Parameters</i> , y con los parámetros recibidos en el orden esperado.
Datos de prueba:	Objeto <i>Class<?></i> que referencia al tipo <i>Registrable</i> .
Acciones de prueba:	Pasar el objeto <i>Class</i> que hace referencia al tipo <i>Registrable</i> al método <i>validateRegistrableProblem</i> .
Resultados esperados:	Método ejecutado sin errores.

Test ID:	AT023
Título:	<i>Registrable</i> cuyo constructor recibe los parámetros en un orden incorrecto.
Característica:	Validar las anotaciones y el tipo de parámetros utilizados en el constructor de la clase <i>Registrable</i> .
Objetivo:	Validar que el constructor público con las anotaciones recibe los parametros en un orden distinto a: <i>Object</i> , <i>File</i> , (<i>int</i> <i>Integer</i> <i>double</i> <i>Double</i>).
Configuración:	Implementación de <i>Registrable</i> con las anotaciones <i>@New-Problem</i> y <i>@Parameters</i> , y con los parámetros recibidos en un orden distinto al especificado.
Datos de prueba:	Objeto <i>Class<?></i> que referencia al tipo <i>Registrable</i> .
Acciones de prueba:	Pasar el objeto <i>Class</i> que hace referencia al tipo <i>Registrable</i> al método <i>validateRegistrableProblem</i> .
Resultados esperados:	Una excepción.

Test ID:	AT024
Título:	<i>Registrable</i> cuyo constructor recibe una cantidad de parámetros distintas a la esperada de acuerdo a la anotación <i>@Parameters</i> .
Característica:	Validar las anotaciones y el tipo de parámetros utilizados en el constructor de la clase <i>Registrable</i> .
Objetivo:	Validar que si el constructor público recibe más parámetros que los indicados en <i>@Parameters</i> , lanza una excepción.
Configuración:	Implementación de <i>Registrable</i> con las anotaciones <i>@New-Problem</i> y <i>@Parameters</i> , y con un parámetro extra en el constructor al indicado en la anotación <i>@Parameters</i> .
Datos de prueba:	Objeto <i>Class<?></i> que referencia al tipo <i>Registrable</i> .
Acciones de prueba:	Pasar el objeto <i>Class</i> que hace referencia al tipo <i>Registrable</i> al método <i>validateRegistrableProblem</i> .
Resultados esperados:	Una excepción.

Test ID:	AT025
Título:	<i>Registrable</i> con una anotación extra en <i>@Parameters</i> .
Característica:	Validar las anotaciones y el tipo de parámetros utilizados en el constructor de la clase <i>Registrable</i> .
Objetivo:	Validar que si <i>@Parameters</i> tiene mas anotaciones que el número de parámetros en el constructor, se lanza una excepción.
Configuración:	Implementación de <i>Registrable</i> con las anotaciones <i>@NewProblem</i> y <i>@Parameters</i> , ésta última con una anotación extra.
Datos de prueba:	Objeto <i>Class<?></i> que referencia al tipo <i>Registrable</i> .
Acciones de prueba:	Pasar el objeto <i>Class</i> que hace referencia al tipo <i>Registrable</i> al método <i>validateRegistrableProblem</i> .
Resultados esperados:	Una excepción.

Test ID:	AT026
Título:	<i>Registrable</i> con dos constructores.
Característica:	Validar las anotaciones y el tipo de parámetros utilizados en el constructor de la clase <i>Registrable</i> .
Objetivo:	Validar que si <i>Registrable</i> tiene dos constructores se lanza una excepción.
Configuración:	Implementación de <i>Registrable</i> con dos constructor, uno de ellos con la anotaciones <i>@NewProblem</i> y <i>@Parameters</i> .
Datos de prueba:	Objeto <i>Class<?></i> que referencia al tipo <i>Registrable</i> .
Acciones de prueba:	Pasar el objeto <i>Class</i> que hace referencia al tipo <i>Registrable</i> al método <i>validateRegistrableProblem</i> .
Resultados esperados:	Una excepción.

Test ID:	AT027
Título:	Id del grupo usado por <i>@NumberToggleInput</i> secuencial.
Característica:	Validar las anotaciones y el tipo de parámetros utilizados en el constructor de la clase <i>Registrable</i> .
Objetivo:	Validar que si <i>Registrable</i> , en el elemento <i>numberToggle</i> de la anotación <i>@Parameters</i> recibe las anotaciones <i>@NumberToggleInput</i> de manera secuencial no se lanza una excepción. Con secuencial se refiere a que <i>@NumberToggleInput</i> con el mismo <i>groupID</i> deben estar juntos.
Configuración:	Implementación de <i>Registrable</i> las anotaciones <i>@NewProblem</i> y <i>@Parameters</i> . El elemento <i>numberToggle</i> de <i>@Parameters</i> tiene las anotaciones <i>@NumberToggleInput</i> con el mismo <i>groupID</i> juntos.
Datos de prueba:	Objeto <i>Class<?></i> que referencia al tipo <i>Registrable</i> .
Acciones de prueba:	Pasar el objeto <i>Class</i> que hace referencia al tipo <i>Registrable</i> al método <i>validateRegistrableProblem</i> .
Resultados esperados:	Método termina sin error.

Test ID:	AT028
Título:	Id del grupo usado por <i>@NumberToggleInput</i> no secuencial.
Característica:	Validar las anotaciones y el tipo de parámetros utilizados en el constructor de la clase <i>Registrable</i> .
Objetivo:	Validar que si <i>Registrable</i> , en el elemento <i>numberToggle</i> de la anotación <i>@Parameters</i> recibe las anotaciones <i>@NumberToggleInput</i> de manera no secuencial se lanza una excepción. Con secuencial se refiere a que <i>@NumberToggleInput</i> con el mismo <i>groupID</i> deben estar juntos.
Configuración:	Implementación de <i>Registrable</i> las anotaciones <i>@NewProblem</i> y <i>@Parameters</i> . El elemento <i>numberToggle</i> de <i>@Parameters</i> no tiene las anotaciones <i>@NumberToggleInput</i> con el mismo <i>groupID</i> juntos.
Datos de prueba:	Objeto <i>Class<?></i> que referencia al tipo <i>Registrable</i> .
Acciones de prueba:	Pasar el objeto <i>Class</i> que hace referencia al tipo <i>Registrable</i> al método <i>validateRegistrableProblem</i> .
Resultados esperados:	Método termina sin error.

Test ID:	AT029
Título:	<i>Registrable</i> con parámetros en el constructor que no están definidos en la anotación <i>@Parameters</i> .
Característica:	Validar las anotaciones y el tipo de parámetros utilizados en el constructor de la clase <i>Registrable</i> .
Objetivo:	Validar que si el constructor <i>Registrable</i> tiene un parámetro no correspondiente al indicado en la anotación <i>@Parameters</i> se lanza una excepción.
Configuración:	Implementación de <i>Registrable</i> las anotaciones <i>@NewProblem</i> y <i>@Parameters</i> . El constructor indica que recibirá un <i>File</i> mientras que la anotación <i>@Parameters</i> indica que en esa posición debería recibirse un <i>Object</i> , el cual hace referencia a un operador de la metaheurística.
Datos de prueba:	Objeto <i>Class<?></i> que referencia al tipo <i>Registrable</i> .
Acciones de prueba:	Pasar el objeto <i>Class</i> que hace referencia al tipo <i>Registrable</i> al método <i>validateRegistrableProblem</i> .
Resultados esperados:	Una excepción.

1.5. *ResultSimulation*

En esta sección se especifica los test automatizados realizados sobre la clase *ResultSimulation*.

Test ID:	AT030
Título:	<i>timeInSeconds</i> guardado correctamente.
Característica:	Validar que la clase abstracta guarda el parametro <i>timeInSeconds</i> correctamente.
Objetivo:	Validar que al pasar el tiempo en segundos a la superclase <i>ResultSimulation</i> se guarda el tiempo correctamente en el campo <i>timeInSeconds</i> .
Configuración:	Clase que hereda de <i>ResultSimulation</i> cuyo constructor delega el parámetro <i>timeInSeconds</i> para que sea guardado en la superclase.
Datos de prueba:	<i>timeInSeconds</i> : 72000
Acciones de prueba:	Pasar al constructor de la superclase abstracta <i>ResultSimulation</i> el tiempo en segundo de la simulación.
Resultados esperados:	<i>timeInSeconds</i> igual a 72000.

Test ID:	AT031
Título:	<i>String</i> que representa el valor de <i>timeInSeconds</i> en formato HH:mm:ss.
Característica:	Validar el método <i>getTimeString</i> .
Objetivo:	Validar que al pasar el tiempo en segundos a la superclase <i>ResultSimulation</i> y llamar al método <i>getTimeString</i> se retorna un <i>String</i> con la hora en formato HH:mm:ss.
Configuración:	Clase que hereda de <i>ResultSimulation</i> cuyo constructor delega el parámetro <i>timeInSeconds</i> para que sea guardado en la superclase.
Datos de prueba:	<i>timeInSeconds</i> : 72000, 0, 1, 1000, 86159, 86399.
Acciones de prueba:	Pasar al constructor de la superclase abstracta <i>ResultSimulation</i> el tiempo en segundo de la simulación.
Resultados esperados:	Los <i>String</i> "20:00:00", "00:00:00", "00:00:01", "00:30:00", "23:55:59", "23:59:59".

Test ID:	AT032
Título:	<i>timeInSeconds</i> fuera del rango de simulación.
Característica:	Validar el intervalo permitido para <i>timeInSeconds</i>
Objetivo:	Validar que si <i>timeInSeconds</i> esta fuera del rango de las 24 horas se lanza una excepción.
Configuración:	Clase que hereda de <i>ResultSimulation</i> cuyo constructor delega el parámetro <i>timeInSeconds</i> para que sea guardado en la superclase.
Datos de prueba:	<i>timeInSeconds</i> : -2, -1, 86400(24:00:00), 86401 (24:00:01).
Acciones de prueba:	Pasar al constructor de la superclase abstracta <i>ResultSimulation</i> el tiempo en segundo de la simulación.
Resultados esperados:	Una excepción para cualquiera de los valores de <i>timeInSeconds</i> usados.

1.6. *JsonSimpleReader*

En esta sección se especifican los casos de prueba para la clase *JsonSimpleReader*.

Test ID:	AT033
Título:	Leer un entero (<i>int</i>) desde un json.
Característica:	Leer valores desde un json.
Objetivo:	Validar que el método <i>getInt</i> retorna un <i>int</i> cuando se lee un número desde una propiedad en json.
Configuración:	Instancia de <i>JsonSimpleReader</i> inicializado y listo para leer.
Datos de prueba:	<p>Un json con los siguientes valores:</p> <pre> { "int": 5, "double": 2.5, "ints": [0,1,2,3,4,5], "doubles": [0.1,0.2,2.3,2.5,2.5], "doubleMatrix": [[0.2,0.3,0.4],[1.2,1.3,1.5]], "intMatrix": [[0,1,2],[3,4,5]], "string": "A simple string", "boolean": true } </pre>
Acciones de prueba:	Llamar al método <i>getInt</i> .
Resultados esperados:	El entero leído desde el json con clave " <i>int</i> ".

Test ID:	AT034
Título:	Leer un numero decimal (<i>double</i>) desde un json.
Característica:	Leer valores desde un json.
Objetivo:	Validar que el método <i>getDouble</i> retorna un <i>double</i> cuando se lee un número desde una propiedad en json.
Configuración:	Instancia de <i>JsonSimpleReader</i> inicializado y listo para leer.
Datos de prueba:	<p>Un json con los siguientes valores:</p> <pre> { "int": 5, "double": 2.5, "ints": [0,1,2,3,4,5], "doubles": [0.1,0.2,2.3,2.5,2.5], "doubleMatrix": [[0.2,0.3,0.4],[1.2,1.3,1.5]], "intMatrix": [[0,1,2],[3,4,5]], "string": "A simple string", "boolean": true } </pre>
Acciones de prueba:	Llamar al método <i>getDouble</i> .
Resultados esperados:	El double leído desde el json con clave " <i>double</i> ".

Test ID:	AT035
Título:	Leer un valor booleano (<i>boolean</i>) desde un json.
Característica:	Leer valores desde un json.
Objetivo:	Validar que el método <i>getBoolean</i> retorna un <i>boolean</i> cuando se lee una propiedad booleana desde el json.
Configuración:	Instancia de <i>JsonSimpleReader</i> inicializado y listo para leer.
Datos de prueba:	<p>Un json con los siguientes valores:</p> <pre> { "int": 5, "double": 2.5, "ints": [0,1,2,3,4,5], "doubles": [0.1,0.2,2.3,2.5,2.5], "doubleMatrix": [[0.2,0.3,0.4],[1.2,1.3,1.5]], "intMatrix": [[0,1,2],[3,4,5]], "string": "A simple string", "boolean": true } </pre>
Acciones de prueba:	Llamar al método <i>getBoolean</i> .
Resultados esperados:	El booleano leído desde el json con clave " <i>boolean</i> ".

Test ID:	AT036
Título:	Leer un arreglo de enteros (<i>int[]</i>) desde un json.
Característica:	Leer valores desde un json.
Objetivo:	Validar que el método <i>getIntegerArray</i> retorna un <i>int[]</i> cuando se lee un arreglo desde el json.
Configuración:	Instancia de <i>JsonSimpleReader</i> inicializado y listo para leer.
Datos de prueba:	<p>Un json con los siguientes valores:</p> <pre> { "int": 5, "double": 2.5, "ints": [0,1,2,3,4,5], "doubles": [0.1,0.2,2.3,2.5,2.5], "doubleMatrix": [[0.2,0.3,0.4],[1.2,1.3,1.5]], "intMatrix": [[0,1,2],[3,4,5]], "string": "A simple string", "boolean": true } </pre>
Acciones de prueba:	Llamar al método <i>getIntegerArray</i> .
Resultados esperados:	El arreglo de enteros desde el json con clave “ <i>ints</i> ”.

Test ID:	AT037
Título:	Leer un arreglo de números decimales (<i>double</i> []) desde un json.
Característica:	Leer valores desde un json.
Objetivo:	Validar que el método <i>getDoubleArray</i> retorna un <i>double</i> [] cuando se lee un arreglo desde el json.
Configuración:	Instancia de <i>JsonSimpleReader</i> inicializado y listo para leer.
Datos de prueba:	<p>Un json con los siguientes valores:</p> <pre> { "int": 5, "double": 2.5, "ints": [0,1,2,3,4,5], "doubles": [0.1,0.2,2.3,2.5,2.5], "doubleMatrix": [[0.2,0.3,0.4],[1.2,1.3,1.5]], "intMatrix": [[0,1,2],[3,4,5]], "string": "A simple string", "boolean": true } </pre>
Acciones de prueba:	Llamar al método <i>getDoubleArray</i> .
Resultados esperados:	El arreglo de valores decimales leído desde el json con clave “ <i>doubles</i> ”.

Test ID:	AT038
Título:	Leer una matriz de enteros (<i>int</i> []) desde un json.
Característica:	Leer valores desde un json.
Objetivo:	Validar que el método <i>getIntegerMatrix</i> retorna un <i>int</i> [] cuando se lee una matriz desde el json.
Configuración:	Instancia de <i>JsonSimpleReader</i> inicializado y listo para leer.
Datos de prueba:	<p>Un json con los siguientes valores:</p> <pre> { "int": 5, "double": 2.5, "ints": [0,1,2,3,4,5], "doubles": [0.1,0.2,2.3,2.5,2.5], "doubleMatrix": [[0.2,0.3,0.4],[1.2,1.3,1.5]], "intMatrix": [[0,1,2],[3,4,5]], "string": "A simple string", "boolean": true } </pre>
Acciones de prueba:	Llamar al método <i>getIntegerMatrix</i> .
Resultados esperados:	La matriz de valores enteros leído desde el json con clave <i>"intMatrix"</i> .

Test ID:	AT039
Título:	Leer una matriz de decimales (<i>double</i> []) desde un json.
Característica:	Leer valores desde un json.
Objetivo:	Validar que el método <i>getDoubleMatrix</i> retorna un <i>double</i> [] cuando se lee una matriz desde el json.
Configuración:	Instancia de <i>JsonSimpleReader</i> inicializado y listo para leer.
Datos de prueba:	<p>Un json con los siguientes valores:</p> <pre> { "int": 5, "double": 2.5, "ints": [0,1,2,3,4,5], "doubles": [0.1,0.2,2.3,2.5,2.5], "doubleMatrix": [[0.2,0.3,0.4],[1.2,1.3,1.5]], "intMatrix": [[0,1,2],[3,4,5]], "string": "A simple string", "boolean": true } </pre>
Acciones de prueba:	Llamar al método <i>getDoubleMatrix</i> .
Resultados esperados:	La matriz de valores decimales leída desde el json con clave " <i>doubleMatrix</i> ".

2. Casos de pruebas manuales

En este capítulo se presenta la especificación formal de los casos de prueba manuales llevados que son llevados a cabo sobre el software.

Test ID:	MT001
Título:	Visualización de la red.
Característica:	Mostrar visualización de la red.
Objetivo:	Confirmar que la red puede ser leída desde un archivo “.inp” y ser visualizada en la aplicación.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	inp: hanoi-Frankenstein.inp
Acciones de prueba:	1. Abrir JHawanetFramework 2. Cargar archivo de red.
Resultados esperados:	El sistema muestra la red leída desde un archivo inp gráficamente en la aplicación.

Test ID:	MT002
Título:	Optimización monoobjetivo realizada completamente.
Característica:	Realizar simulación monoobjetivo.
Objetivo:	Confirmar que se puede llevar a cabo la resolución del problema <i>PipeOptimizing</i> sobre la red abierta.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	independentRun = 10 Archivo inp: hanoi-Frankenstein.inp Archivo gama: hanoiHW.Gama
Acciones de prueba:	<ol style="list-style-type: none"> 1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Seleccionar el problema <i>PipeOptimizing del menú</i>. 4. Configurar el problema usando la ventana de configuración. 5. Realizar la optimización.
Resultados esperados:	Al terminar la optimización el sistema muestra una interfaz con las soluciones generadas por la optimización. Debe haber tantas soluciones como el numero de configuraciones independientes (<i>independentRun</i>).

Test ID:	MT003
Título:	Optimización monoobjetivo cancelada.
Característica:	Realizar simulación monoobjetivo.
Objetivo:	Confirmar que se puede llevar a cabo la resolución del problema <i>PipeOptimizing</i> sobre la red abierta y que se puede cancelar el proceso cerrando la ventana o pulsando cancelar.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: hanoi-Frankenstein.inp Archivo gama: hanoiHW.Gama
Acciones de prueba:	<ol style="list-style-type: none"> 1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Seleccionar el problema <i>PipeOptimizing</i> del menú. 4. Configurar el problema usando la ventana de configuración. 5. Realizar la optimización.
Resultados esperados:	Al cancelar la búsqueda de soluciones la ventana de estado indica que la optimización a sido detenida.

Test ID:	MT004
Título:	Optimización multiobjetivo realizada completamente.
Característica:	Realizar simulación multiobjetivo.
Objetivo:	Confirmar que se puede llevar a cabo la resolución del problema <i>PumpingScheduling</i> sobre la red abierta.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: Vanzyl.inp Archivo gama: VanzylConfiguration.json
Acciones de prueba:	<ol style="list-style-type: none"> 1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Seleccionar el problema <i>PumpingScheduling</i> del menú. 4. Seleccionar el operador NSGAIL. 5. Configurar el problema usando la ventana de configuración. 6. Realizar la optimización.
Resultados esperados:	Al terminar la optimización el sistema muestra una interfaz con las soluciones generadas por la optimización. Estas soluciones corresponden a la frontera de pareto del Experimento.

Test ID:	MT005
Título:	Optimización multiobjetivo cancelada.
Característica:	Realizar simulación multiobjetivo.
Objetivo:	Confirmar que se puede llevar a cabo la resolución del problema <i>PumpingScheduling</i> sobre la red abierta y que se puede cancelar el proceso cerrando la ventana o pulsando cancelar.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: Vanzyl.inp Archivo gama: VanzylConfiguration.json
Acciones de prueba:	<ol style="list-style-type: none"> 1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Seleccionar el problema <i>PumpingScheduling</i> del menú. 4. Seleccionar el operador NSGAIL. 5. Configurar el problema usando la ventana de configuración. 6. Realizar la optimización.
Resultados esperados:	Al cancelar la búsqueda de soluciones la ventana de estado indica que la optimización a sido detenida.

Test ID:	MT006
Título:	Ver soluciones de los problemas monoobjetivos gráficamente a medida que se ejecuta la optimización.
Característica:	Visualizar gráficamente las soluciones.
Objetivo:	Comprobar que a medida que algoritmo va generando soluciones estas pueden ser visualizadas. El grafico es Objetivo vs Numero de generaciones.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: hanoi-Frankenstein.inp Archivo gama: hanoiHW.Gama
Acciones de prueba:	1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Escoger el problema monoobjetivo 4. Configurar el problema monoobjetivo y ejecutar 5. Ir a la ventana del gráfico.
Resultados esperados:	Un gráfico de dos dimensiones en el que se muestre los resultados de las soluciones generadas por cada generación de cada una de las ejecuciones independientes.

Test ID:	MT007
Título:	Ver soluciones del problema multiobjetivos, con dos objetivos, gráficamente a medida que se ejecuta la optimización.
Característica:	Visualizar gráficamente las soluciones
Objetivo:	Comprobar que a medida que algoritmo va generando soluciones estas pueden ser visualizadas. El grafico es Objetivo1 vs Objetivo2.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: Vanzyl.inp Archivo gama: VanzylConfiguration.json
Acciones de prueba:	<ol style="list-style-type: none"> 1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Escoger el problema monoobjetivo 4. Configurar el problema monoobjetivo y ejecutar 5. Ir a la ventana del gráfico.
Resultados esperados:	Un gráfico de dos dimensiones en el que se muestre los resultados de las soluciones generadas por cada generación de cada una de las ejecuciones independientes.

Test ID:	MT008
Título:	Guardar la gráfica de las soluciones del problema mono-objetivo como png.
Característica:	Guardar gráfica de soluciones.
Objetivo:	Confirmar que se puede guardar el gráfico de soluciones como un png.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: hanoi-Frankenstein.inp Archivo gama: hanoiHW.Gama
Acciones de prueba:	<ol style="list-style-type: none"> 1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Escoger el problema monoobjetivo 4. Configurar el problema monoobjetivo y ejecutar 5. Ir a la ventana del gráfico. 6. Pulsar el boton para guardar una captura del gráfico. 7. Configurar donde guardar la captura.
Resultados esperados:	Generación de un archivo png en el equipo.

Test ID:	MT009
Título:	Guardar la gráfica de soluciones de problemas de 2 objetivos como png.
Característica:	Guardar gráfica de soluciones.
Objetivo:	Confirmar que se puede guardar el gráfico de soluciones como un png.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: Vanzyl.inp Archivo gama: VanzylConfiguration.json
Acciones de prueba:	<ol style="list-style-type: none"> 1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Escoger el problema monoobjetivo 4. Configurar el problema monoobjetivo y ejecutar 5. Ir a la ventana del gráfico. 6. Pulsar el boton para guardar una captura del gráfico. 7. Configurar donde guardar la captura.
Resultados esperados:	Generación de un archivo png en el equipo.

Test ID:	MT010
Título:	Guardar las solución seleccionada como inp.
Característica:	Guardar resultados de la optimización.
Objetivo:	Confirmar que se puede exportar los resultados de la solución sobre el archivo de red (inp).
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: hanoi-Frankenstein.inp Archivo gama: hanoiHW.Gama
Acciones de prueba:	<ol style="list-style-type: none"> 1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Escoger el problema monoobjetivo 4. Configurar el problema monoobjetivo y ejecutar 5. Esperar que la ejecución finalice. 6. Seleccionar una solución. 7. Pulsar el botón guardar como inp. 8. Configurar donde guardar el archivo.
Resultados esperados:	Generación del archivo de configuración de la red (inp) con la solución aplicada.

Test ID:	MT011
Título:	Guardar las soluciones en archivos tsv.
Característica:	Guardar resultados de la optimización.
Objetivo:	Confirmar que se puede exportar las soluciones a dos archivos tsv. Uno para las variables y el otro para los objetivos.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: hanoi-Frankenstein.inp Archivo gama: hanoiHW.Gama
Acciones de prueba:	<ol style="list-style-type: none"> 1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Escoger el problema monoobjetivo 4. Configurar el problema monoobjetivo y ejecutar 5. Esperar que la ejecucion finalice. 6. Pulsar el boton guardar tabla. 7. Configurar donde guardar los archivo.
Resultados esperados:	Generación del 2 archivos .tsv. Uno con el prefijo FUN_ y otro con el prefijo VAR_. El archivo FUN contiene el valor de los objetivos de las soluciones. El archivo VAR contiene las variables de la soluciones.

Test ID:	MT012
Título:	Guardar las soluciones en un excel.
Característica:	Guardar resultados de la optimización.
Objetivo:	Confirmar que se puede exportar la tabla de resultados completas a un excel.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: hanoi-Frankenstein.inp Archivo gama: hanoiHW.Gama
Acciones de prueba:	<ol style="list-style-type: none"> 1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Escoger el problema monoobjetivo 4. Configurar el problema monoobjetivo y ejecutar 5. Esperar que la ejecucion finalice. 6. Pulsar el boton guardar tabla como excel. 7. Configurar donde guardar el archivo.
Resultados esperados:	Un archivo excel con los mismos datos que la tabla de resultados de la aplicación.

Test ID:	MT013
Título:	Simulación hidráulica sobre red de un solo tiempo.
Característica:	Realizar simulación con configuración de archivo de red.
Objetivo:	Confirmar que se puede realizar la simulación utilizando los valores del archivo de red.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: hanoi-Frankenstein.inp Archivo gama: hanoiHW.Gama
Acciones de prueba:	1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Pulsar botón <i>Execute</i> de la ventana principal.
Resultados esperados:	Ejecución realizada sin ningun error.

Test ID:	MT014
Título:	Ver resultados de la simulación hidráulica de un solo tiempo
Característica:	Realizar simulación con configuración de archivo de red.
Objetivo:	Confirmar que se puede visualizar los resultados de la simulación realizada.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: hanoi-Frankenstein.inp Archivo gama: hanoiHW.Gama
Acciones de prueba:	1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Pulsar botón <i>Execute</i> de la ventana principal. 4. Pulsar botón <i>Results</i> de la ventana principal.
Resultados esperados:	Una interfaz que permite seleccionar si se quieren ver los resultados para los enlaces o los nodos.

Test ID:	MT015
Título:	Simulación hidráulica sobre red de mas de un tiempo.
Característica:	Realizar simulación con configuración de archivo de red.
Objetivo:	Confirmar que se puede realizar la simulación utilizando los valores del archivo de red.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: Vanzyl.inp Archivo gama: VanzylConfiguration.json
Acciones de prueba:	1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Pulsar boton <i>Execute</i> de la ventana principal.
Resultados esperados:	Ejecución realizada sin ningun error.

Test ID:	MT016
Título:	Ver resultados de la simulación hidráulica de mas de un tiempo
Característica:	Realizar simulación con configuración de archivo de red.
Objetivo:	Confirmar que se puede visualizar los resultados de la simulación realizada.
Configuración:	El equipo tiene la aplicación JHawanetFramework lista para ejecutar.
Datos de prueba:	Archivo inp: Vanzyl.inp Archivo gama: VanzylConfiguration.json
Acciones de prueba:	1. Abrir JHawanetFramework 2. Cargar archivo de red. 3. Pulsar botón <i>Execute</i> de la ventana principal. 4. Pulsar botón <i>Results</i> de la ventana principal.
Resultados esperados:	Una interfaz que permite seleccionar si se quieren ver los resultados para los enlaces o los nodos. Adicionalmente, permite escoger el tiempo de simulación y listar los resultados para todos los tiempos de un elemento de la red específico.