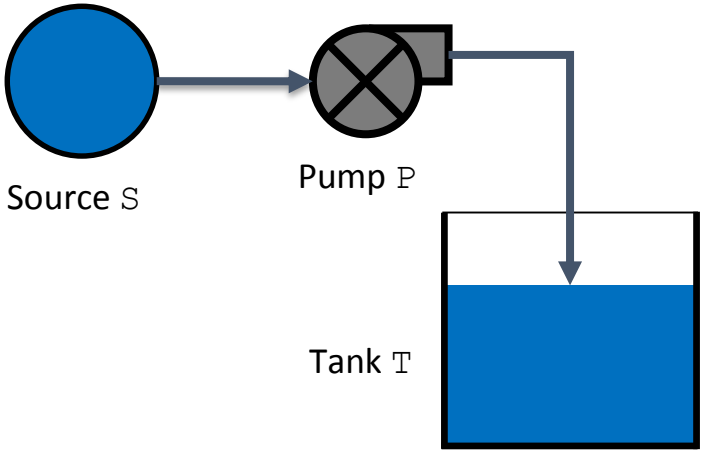
OpenAltaRica - Example

A (Simple) Water Supply System

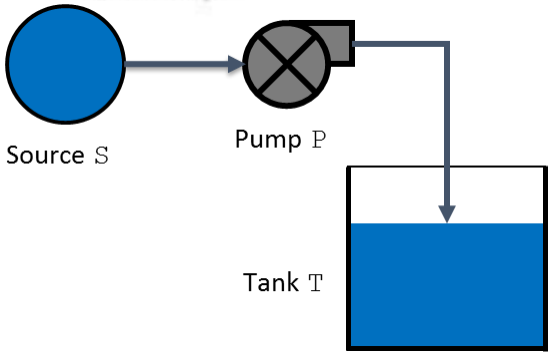
(Simple) Water Supply System



WHAT Feel the tank

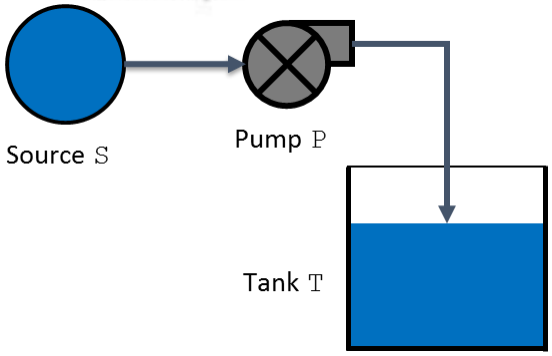
HOW Move the water from the source `S` to the tank `T` by means of a pump `P`

(Simple) Water Supply System



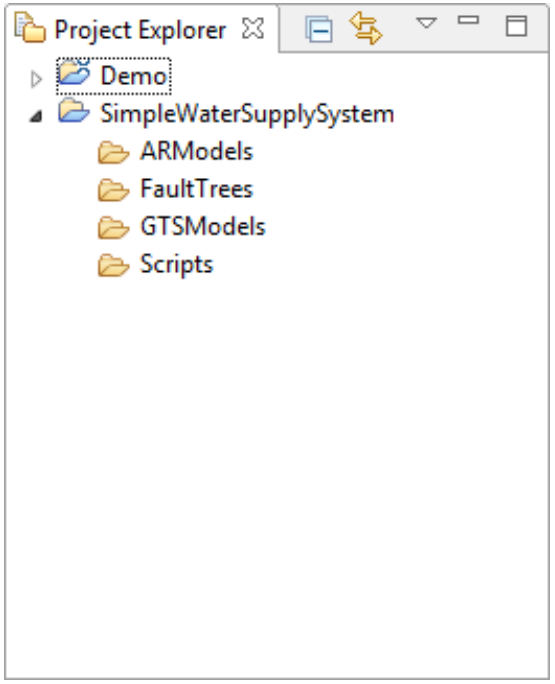
A. To design your model

(Simple) Water Supply System



A. To design your model

1. Create a new project into the AltaRica 3.0 Text Editor:
   a. ‘File’->‘New’->’Project’;
   b. ‘AltaRica’ folder -> ‘AltaRica Project’ -> ‘Next’;
   c. ‘NameOfYourProject’ -> ‘Finish’.



It also creates four sub-folders:

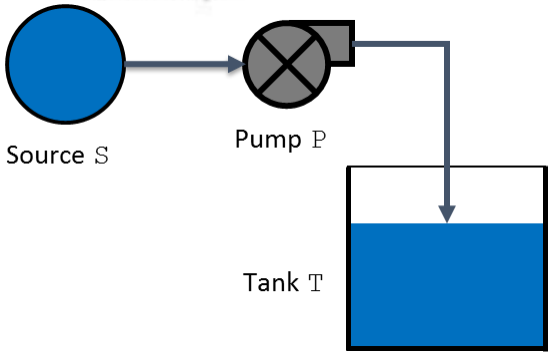
ARModels: for AltaRica 3.0 models;

FaultTrees: for generated fault trees (from AltaRica 3.0 models);

GTSModels: for flattened AltaRica 3.0 models;

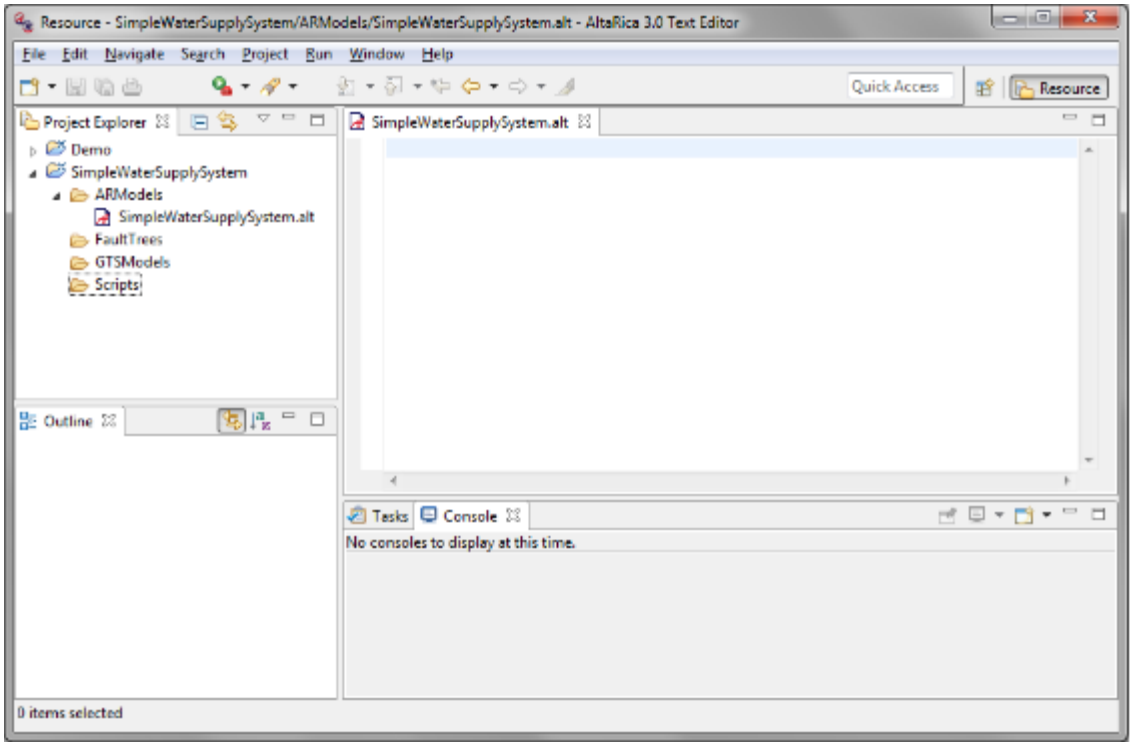
Scripts: for script simulations.

(Simple) Water Supply System

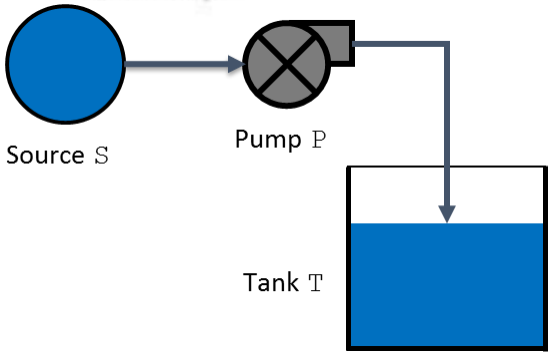


A. To design your model

2. Create a new AltaRica 3,0 model:
   a. Click on the folder ‘ARModels’;
   b. ‘File’ -> New’ -> ‘Other’;
   c. Folder ‘AltaRica’ -> ‘AltaRica Model’ -> ‘Next’;
   d. ‘NameOfYourModel.alt’ -> ‘Finish’.



(Simple) Water Supply System



A. To design your model

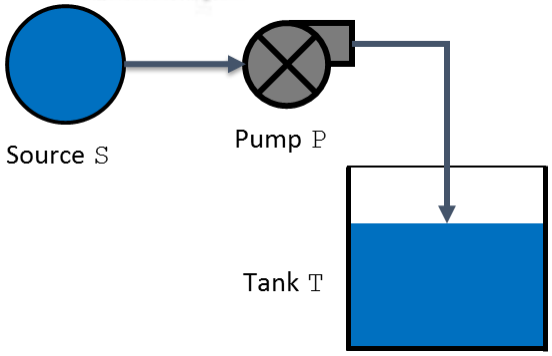
3. Design your AltaRica 3.0 model.

The source and the pump are repairable components

=> a generic class representing repairable components

```
class RepairableComponent
    Boolean working (init = true);
    parameter Real lambda = 0.0001;
    parameter Real mu = 0.01;
    event failure (delay = exponential(lambda));
    event repair (delay = exponential(mu));
    transition
        failure: working -> working := false;
        repair: not working -> working := true;
end
```

(Simple) Water Supply System



A. To design your model

3. Design your AltaRica 3.0 model.

1
```
class RepairableComponent
    Boolean working (init = true);
    parameter Real lambda = 0.0001;
    parameter Real mu = 0.01;
    event failure (delay = exponential(lambda));
    event repair (delay = exponential(mu));
    transition
        failure: working -> working := false;
        repair: not working -> working := true;
end
```

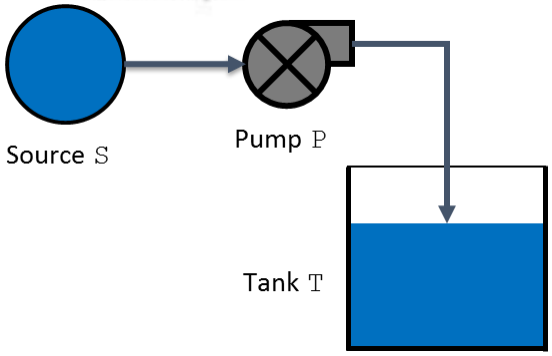
2
```
class Source
    extends RepairableComponent;
    Boolean output (reset = false);
    assertion
        output := if working then true else false;
end
```

3
```
class Pump
    extends RepairableComponent;
    Boolean input, output (reset = false);
    assertion
        output := if working then input else false;
end
```

4
```
class Tank
    Boolean input (reset = false);
end
```

5
```
block SimpleWaterSupplySystem
    Source S;
    Pump P;
    Tank T;
    assertion
        P.input := S.output;
        T.input := P.output;
end
```

(Simple) Water Supply System



A. To design your model

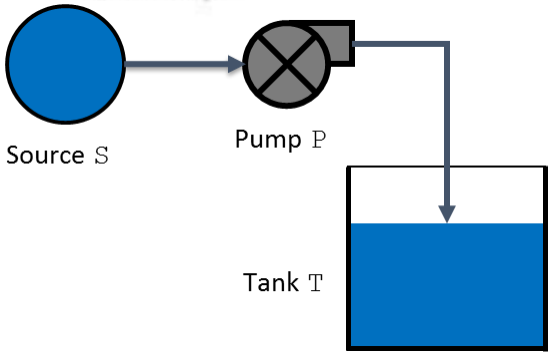
3. Design your AltaRica 3.0 model.

Add an observer providing information about the input flow of the tank. There is no input flow to the tank if:

the value of the flow variable '`input`' of the tank '`T`' is equal to '`false`'.

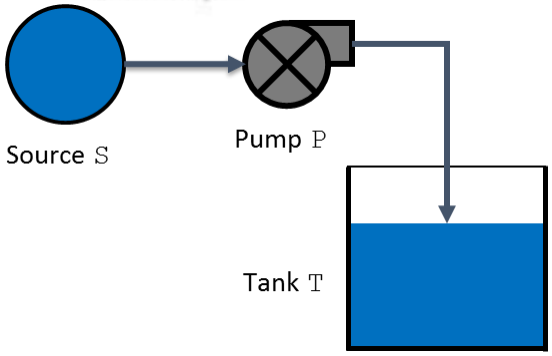
```
block SimpleWaterSupplySystem
    Source S;
    Pump P;
    Tank T;

    observer Boolean tankIsNotSupplied = T.input == false;

    assertion
        P.input := S.output;
        T.input := P.output;
end
```

(Simple) Water Supply System



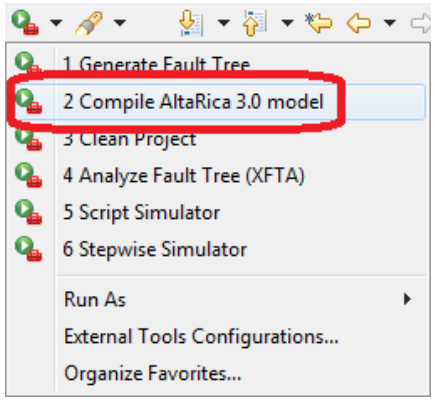
B. To simulate the model

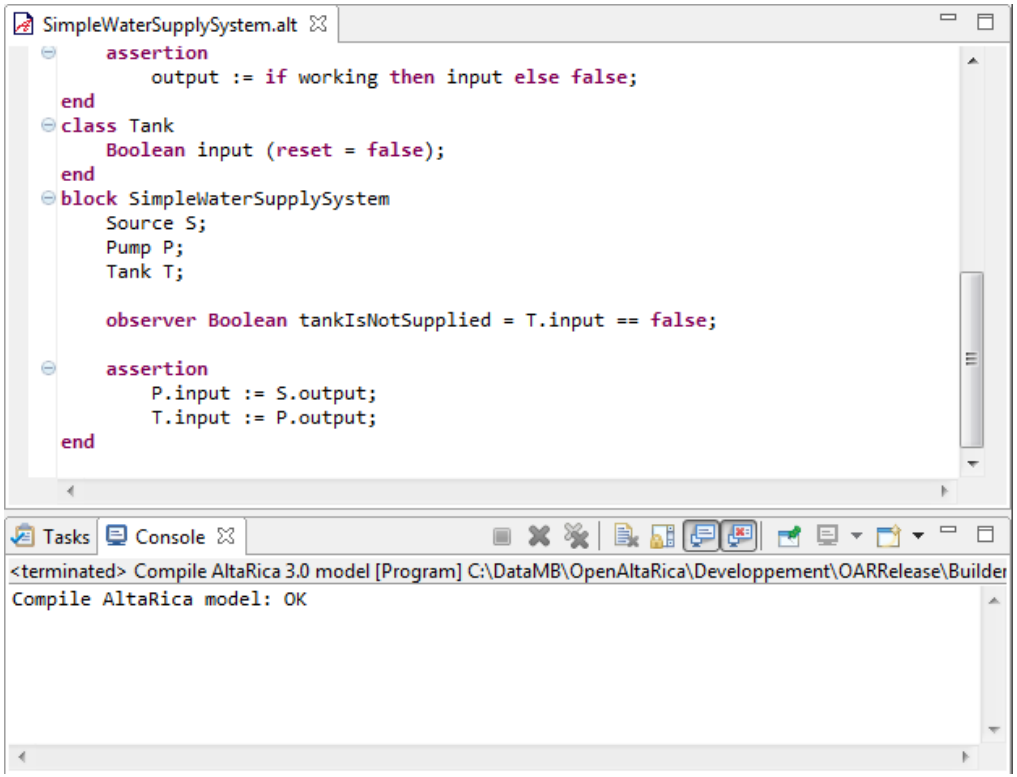
(Simple) Water Supply System



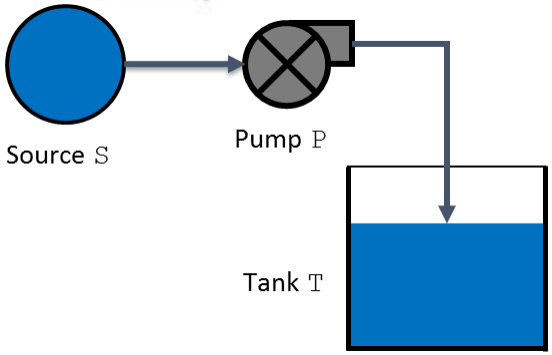
B. To simulate the model

1. Compile the AltaRica model:
   a. Select your AltaRica model;
   b. Click on the ‘Run’ button and select ‘Compile AltaRica 3.0 model’;
   c. Results of compilation are printed into the console.



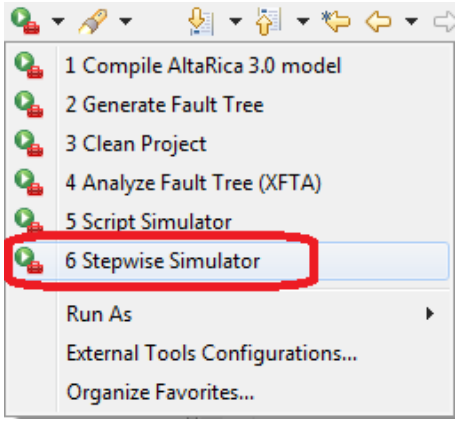


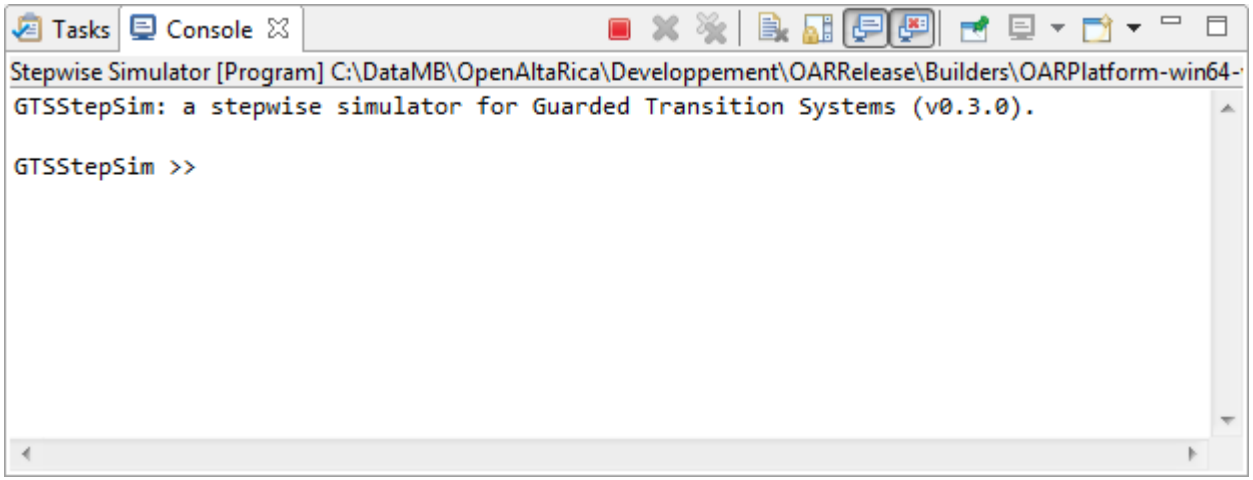
(Simple) Water Supply System



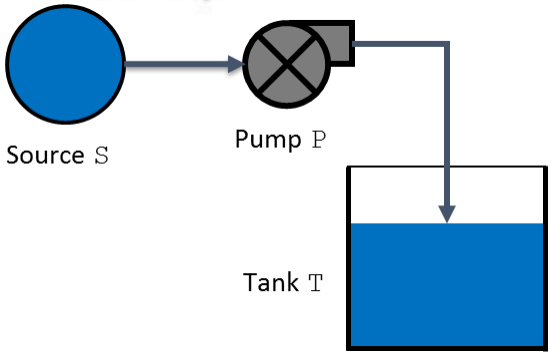
B. To simulate the model

2. Launch the stepwise simulator:
   a. Select your AltaRica model;
   b. Click on the ‘Run’ button and select ‘Stepwise Simulator’;
   c. The console starts the simulation.





(Simple) Water Supply System



B. To simulate the model

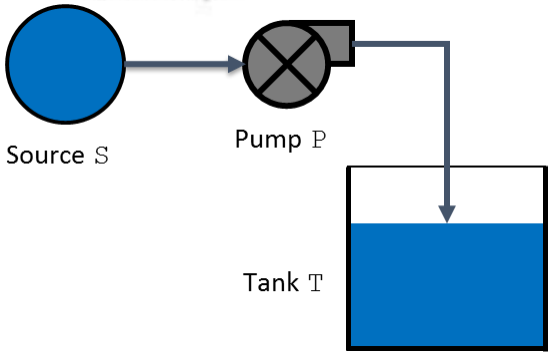
3. Run commands to perform experiments:
   - Run the command ‘help’ to find commands;
   - Run ‘q’ or ‘quit’ to stop the simulator.

Tasks | Console

Stepwise Simulator [Program] C:\DataMB\OpenAltaRica\Developpement\OARRelease\Builders\OARPlatform-win64-

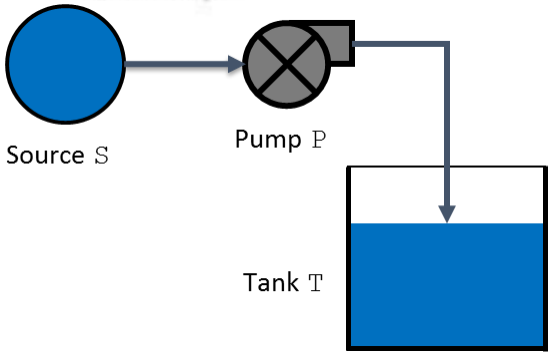
```
GTSStepSim >> help

Available commands:
-------------------------------------------------------------------------------
  h(elp)?                to print this help.
  print gts              to print the simulated GTS model.
  print v(ariables)?     to print the variables and their values.
  print o(bservers)?     to print the observers and their values.
  print tr(ansitions)?   to print the transitions fireable in the current state.
  print h(istory)?       to print the execution history.
  fire <index>           to fire the transition number <index>.
  fire <name>            to fire the transition named <name>.
  fire -i
  fire --immediate       to fire all immediate transitions.
  fire -f
  fire --first           to fire the first transition.
  back                   to cancel the last transition firing.
  restart                to restart the simulation.
  set (no)? trace        to turn on/off the trace mode.
  set (no)? display      to turn on/off the display mode.
  q(uit)?                to quit.
-------------------------------------------------------------------------------

GTSStepSim >>
```

(Simple) Water Supply System



C. To generate a fault tree

(Simple) Water Supply System

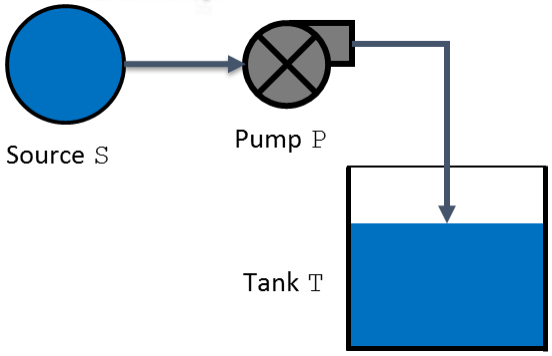


C. To generate a fault tree

1. Define the top event into the AltaRica model “There is no input flow to the tank”:
   a. Create a Boolean flow variable;
   b. Update its value into the system’s assertion (i.e. the assertion into the block ‘SimpleWaterSupplySystem’).

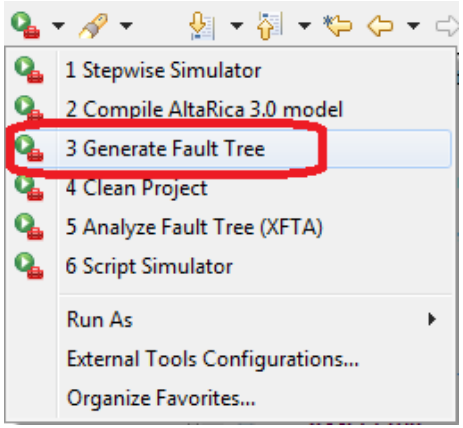
```
block SimpleWaterSupplySystem
    Source S;
    Pump P;
    Tank T;

    Boolean failed (reset = false);

    assertion
        P.input := S.output;
        T.input := P.output;
        failed := T.input == false;
end
```

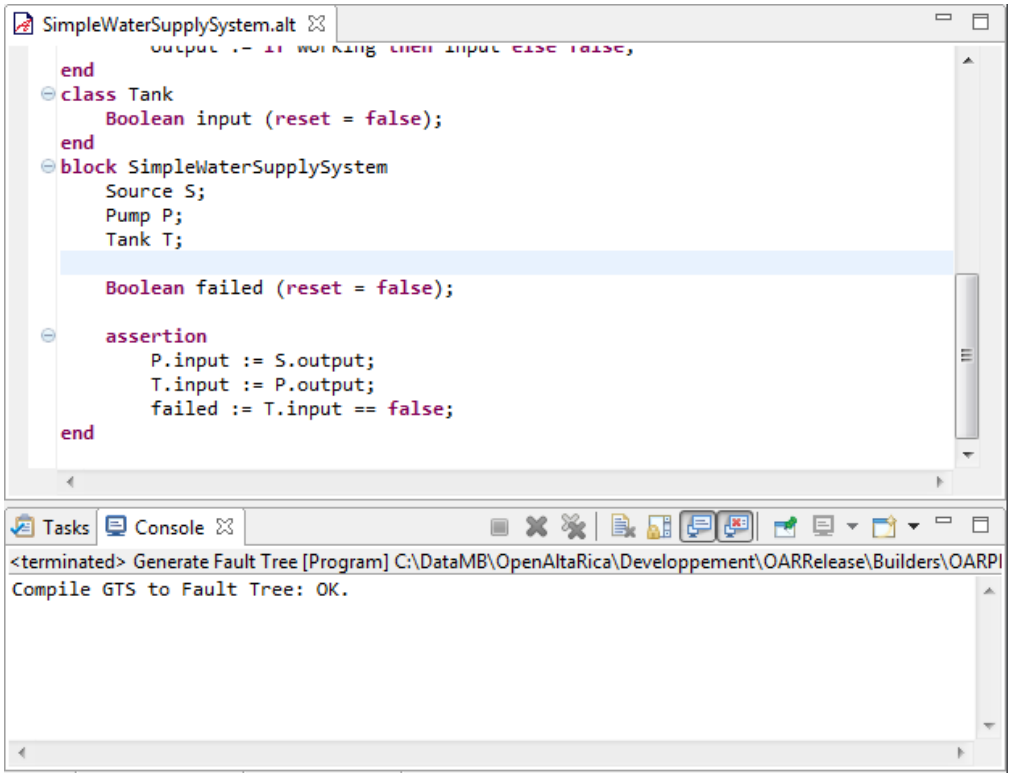
WARNING: this way of practice (define the top event by mean of a flow variable) will be deprecated for the future versions.



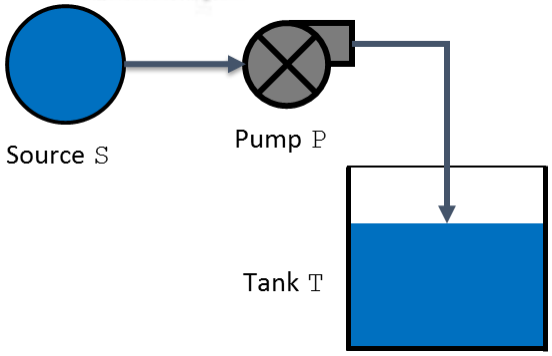
C. To generate a fault tree

2. Compile the AltaRica model:
3. Launch the compiler to fault trees:
   a. Select your AltaRica model;
   b. Click on the ‘Run’ button and select ‘Generate Fault Tree’;
   c. Results of generation are printed into the console.



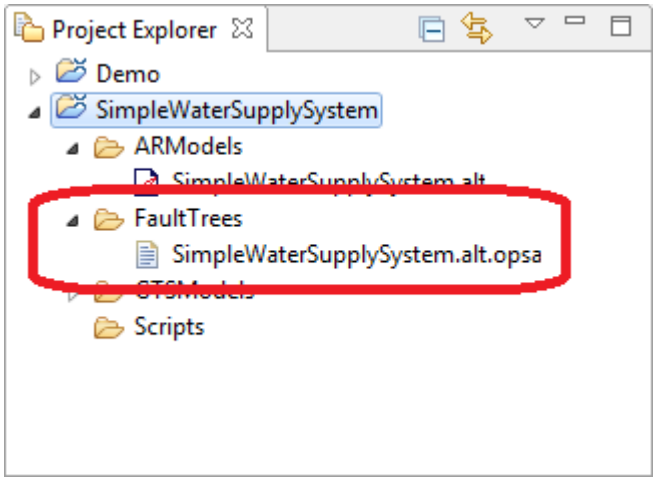


(Simple) Water Supply System

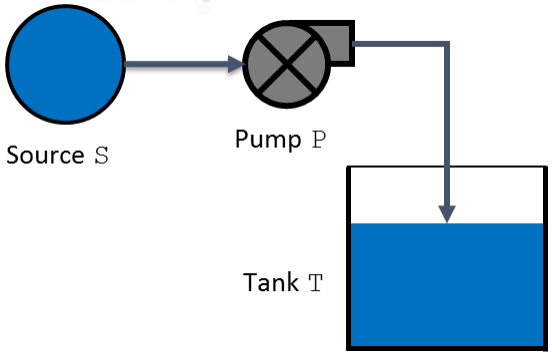


C. To generate a fault tree

Your fault tree is generated into the folder ‘Fault Trees’ on the Open-PSA format.
You can use your dedicated tool to perform analyses on this fault tree.



(Simple) Water Supply System

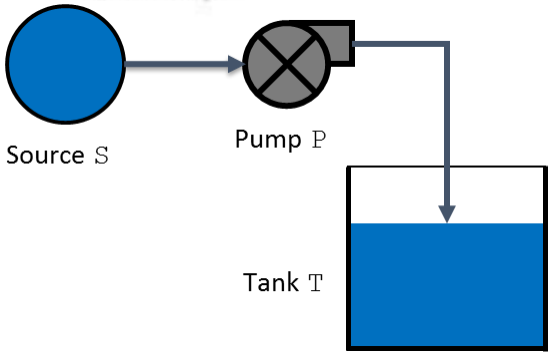


D. To analyze a fault tree

This part is optional because it depends on the considered fault tree assessment engine.

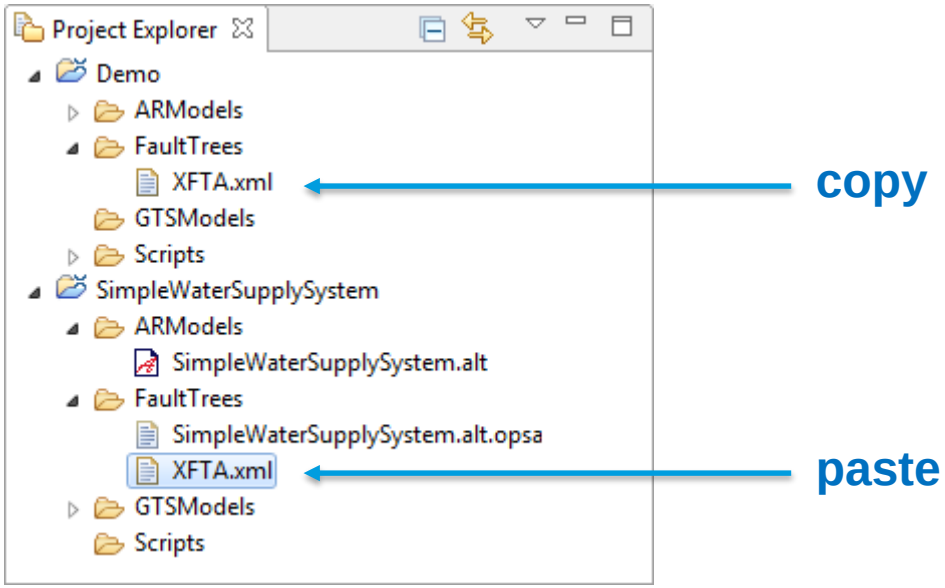
For the following, we consider XFTA.

- The OpenAltaRica platform is configured to assess fault trees with XFTA;
- To install XFTA, please refer to the ‘Getting Started’ document.

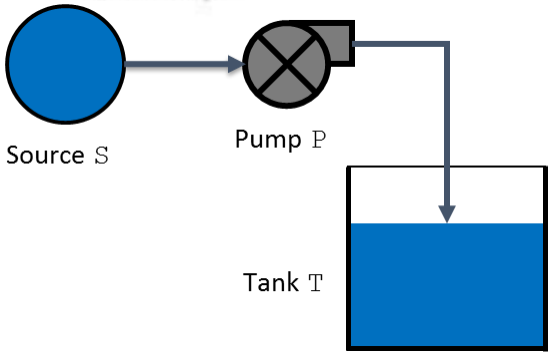


D. To analyze a fault tree

1. Create the script file for XFTA:
   a. Copy the file ‘XFTA.xml’ into the folder ‘FaultTrees’ of the project ‘Demo’;
   b. Paste it into the folder ‘FaultTrees’ of your project.



(Simple) Water Supply System



D. To analyze a fault tree

2. Edit the script file for XFTA:
   a. The input model is the generated fault tree;
   b. The output file can be created into the folder ‘Results’ of the project;
   c. Other information depend on what you want to assess. For this example, we focus on the top event ‘failed_true’ (the declared variable ‘failed’ when its value is ‘true’):
      i. The minimal cutsets for this top event ‘failed_true’;
      ii. The probabilities of this top event ‘failed_true’ with specific mission times.

SimpleWaterSupplySystem.alt | XFTA.xml

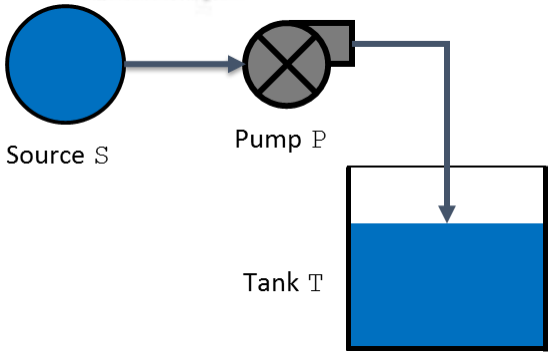
```
<?xml version="1.0"?>
<!DOCTYPE xfta>
<xfta>
  <load>
    <model input="SimpleWaterSupplySystem.alt.opsa" />
  </load>
  <build>
    <minimal-cutsets top-event="failed_true" handle="MCS"/>
  </build>
  <set>
    <option name="print-minimal-cutset-rank" value="on" />
    <option name="print-minimal-cutset-order" value="off" />
    <option name="print-minimal-cutset-probability" value="off" />
    <option name="print-minimal-cutset-contribution" value="off" />
  </set>
  <compute>
      <probability top-event="failed_true" handle="MCS" mission-time="8760" output="..\FaultTrees\Results_failed.txt">
        <option name="mode" value="append"/>
      </probability>
      <probability top-event="failed_true" handle="MCS" output="..\FaultTrees\Results_failed.txt">
        <option name="mode" value="append"/>
      </probability>
  </compute>
  <print>
    <minimal-cutsets top-event="failed_true" handle="MCS" mission-time="1000" output="..\FaultTrees\Results_failed.txt">
      <option name="mode" value="append"/>
    </minimal-cutsets>
  </print>
</xfta>
```

a

b

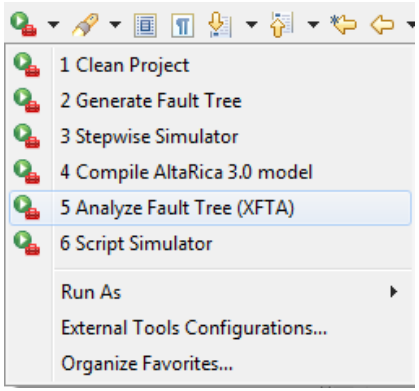
c

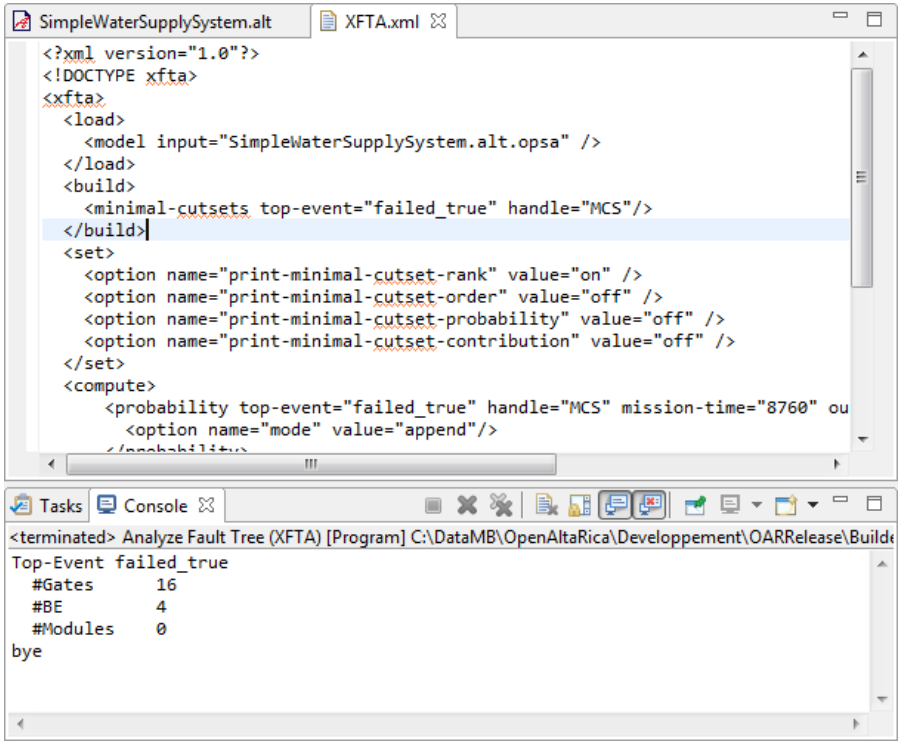
(Simple) Water Supply System



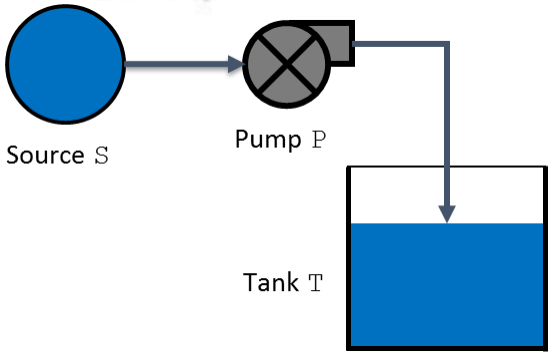
D. To analyze a fault tree

3. Launch XFTA:
   a. Select this file ‘XFTA.xml’;
   b. Click on the ‘Run’ button and select ‘Analyze Fault Tree (XFTA)’;
   c. Some results are printed into the console and the file of results is into the folder ‘Results’.

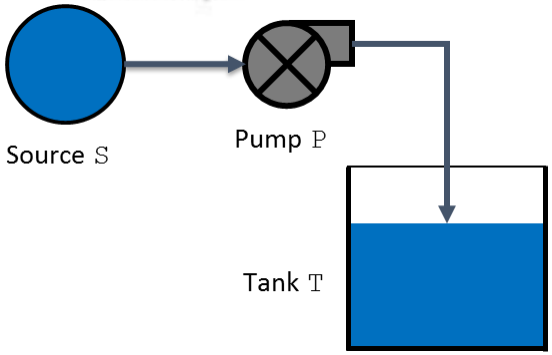




(Simple) Water Supply System

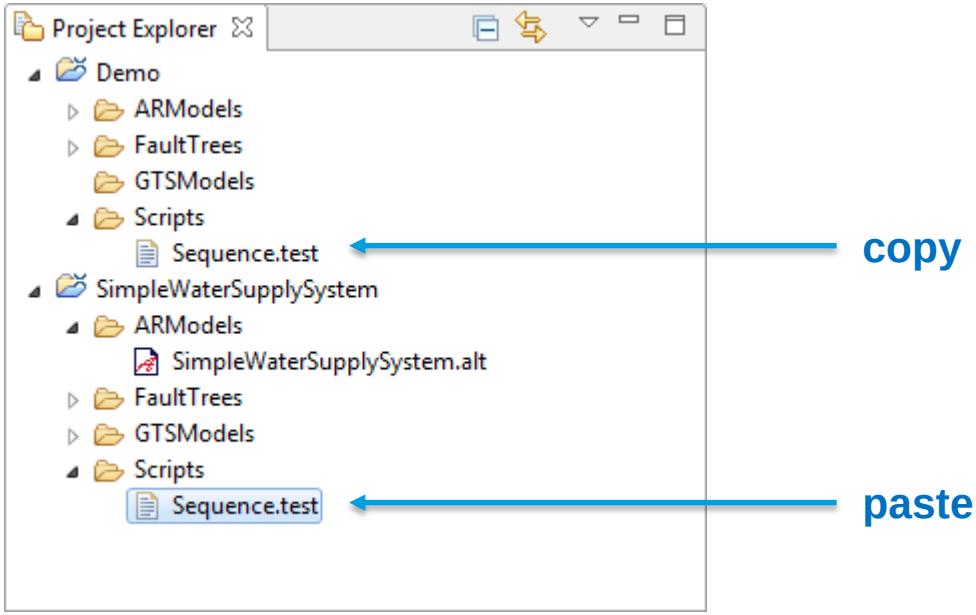


E. To script simulation

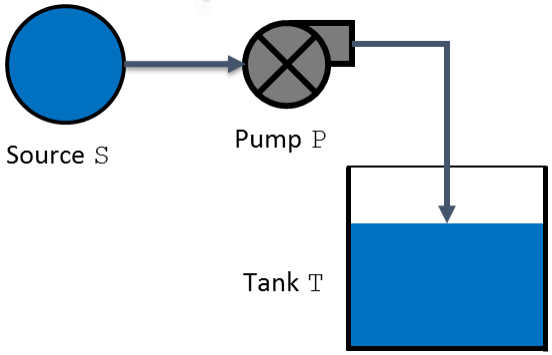


E. To script simulation

1. Create the script file for simulation:
   a. Copy the file ‘Sequence.test’ into the folder ‘Scripts’ of the project ‘Demo’;
   b. Paste it into the folder ‘Scripts’ of your project.

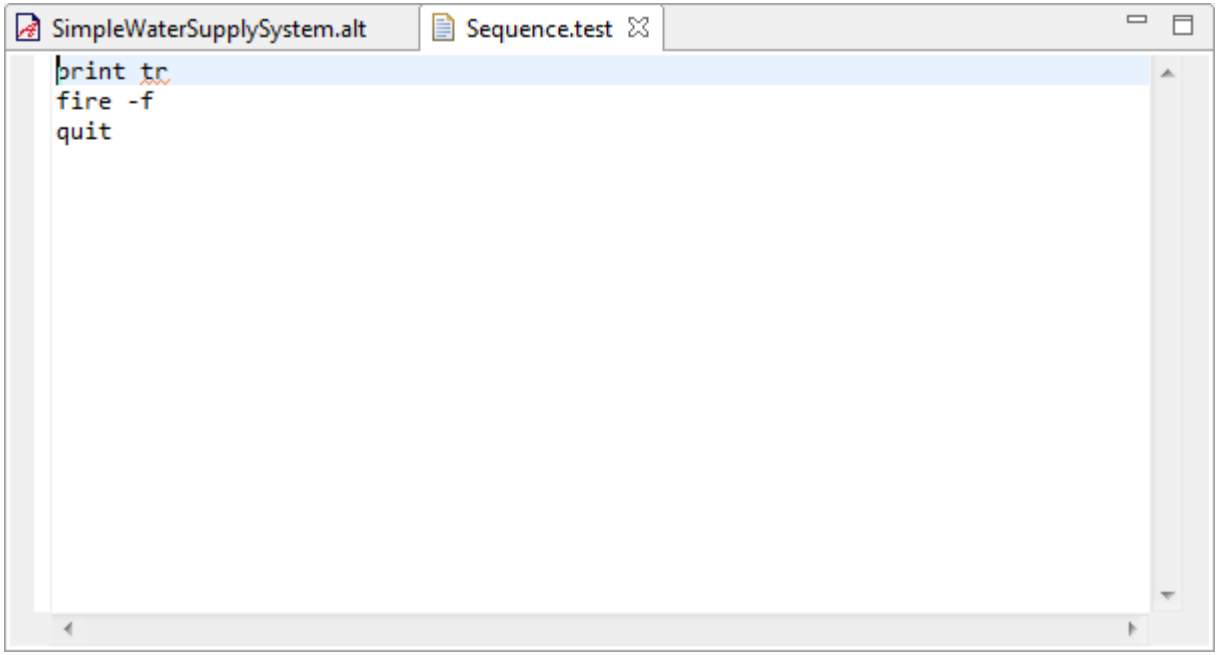


(Simple) Water Supply System

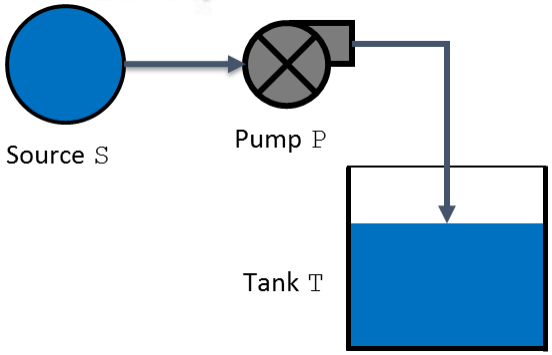


E. To script simulation

2. Edit the script file for simulation:
   a. Only one command per line;
   b. If no command ‘quit’ at the end, the simulator will continue by waiting your commands;

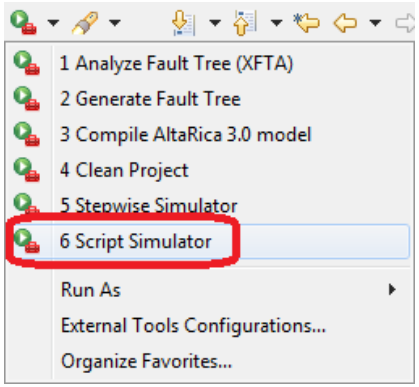


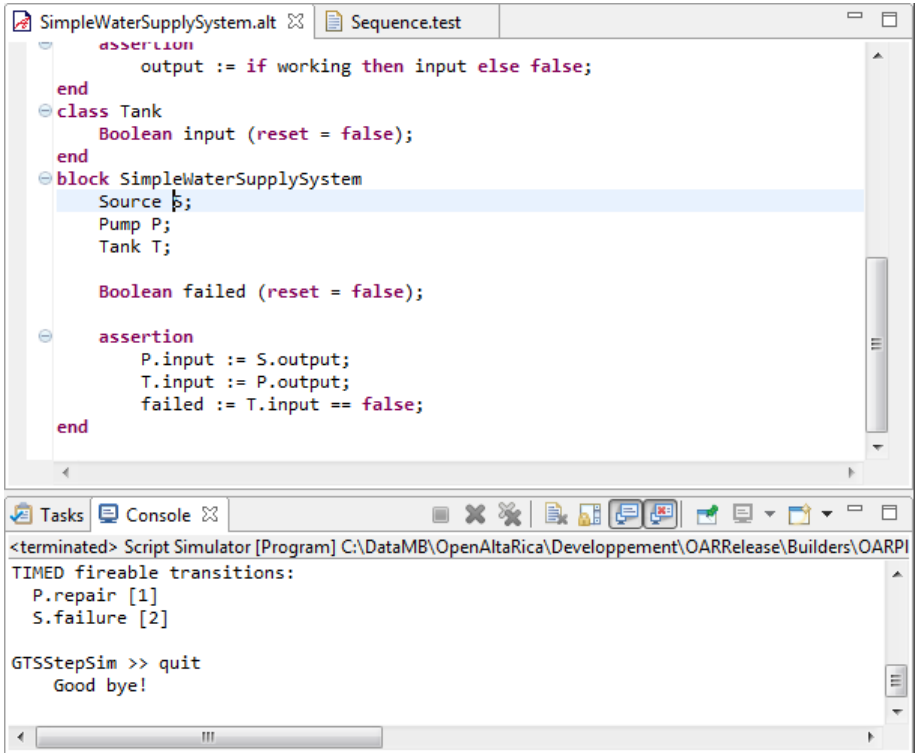
(Simple) Water Supply System



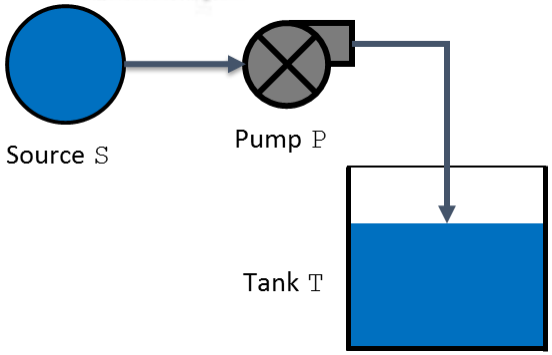
E. To script simulation

3. Launch the Script simulator:
   a. Select your AltaRica model;
   b. Click on the 'Run' button and select 'Script Simulator';
   c. Results of script simulation are printed into the console.



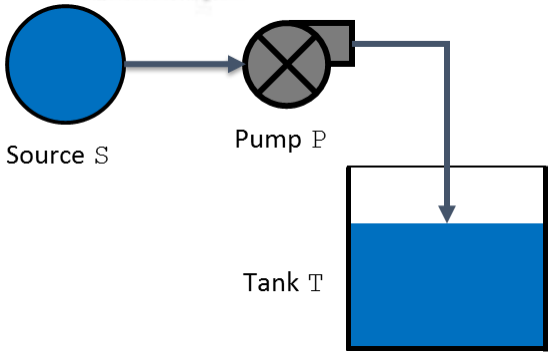


(Simple) Water Supply System



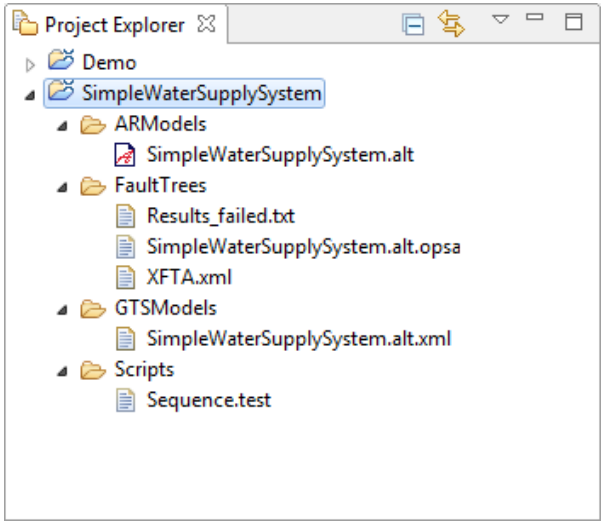
F. To clean a project

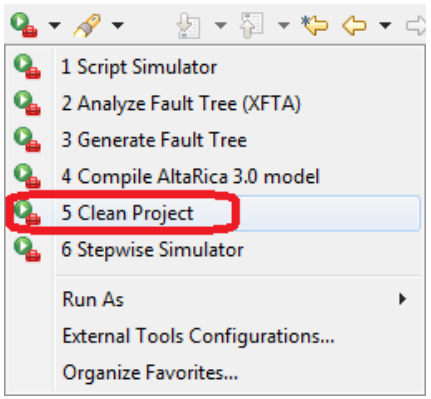
(Simple) Water Supply System

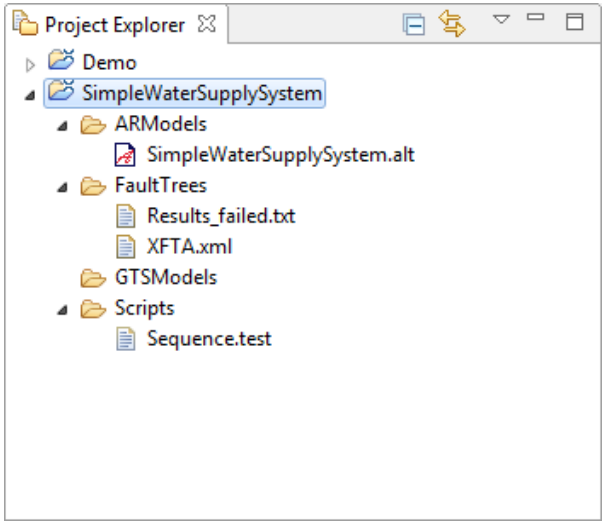


F. To clean a project

3. Launch the cleaner:
   a. Select your project;
   b. Click on the ‘Run’ button and select ‘Clean Project’;
   c. Generated elements are removed (GTS models, FT, etc.).







CONTACTS

- ***OpenAltaRica project***
  www.openaltarica.fr
  contact.oar@irt-systemx.fr