

Accélérateur de la transformation numérique



# **OpenAltaRica** - Example

A (Simple) Water Supply System

www.irt-systemx.fr













© Copyright IRT SystemX 2015-2018 (v3.1)



#### A (Simple) Water Supply System



#### WHAT Feel the tank

HOW Move the water from the source  ${\mathbb S}$  to the tank  ${\mathbb T}$  by means of a pump  ${\mathbb P}$ 



A (Simple) Water Supply System



### A. To design your model



#### A (Simple) Water Supply System





## A. To design your model

- 1. Create a new project into the AltaRicaWizard:
  - a. 'File'->'New File or Project'->'New Project';
  - b. 'SimpleWaterSupplySystem' -> 'OK'.



#### A (Simple) Water Supply System



## A. To design your model

- 2. Create a new AltaRica 3.0 model:
  - a. Right-click on the project 'SimpleWaterSupplySystem';
  - b. 'Add new file to "SimpleWaterSupplySystem";
  - c. You can create a new folder (e.g. 'ARModels');
  - d. 'SimpleWaterSupplySystem.alt' -> 'Save'.

😵 AltaRicaWizard Editor		
<u>File Edit</u> Project Tool Window Help		
🗋 🖉 🔚 🗊 ≽ 🕅		
Projects 🗗 🗙	SimpleWaterSupplySystem.alt 🗵	
<ul> <li>SimpleWaterSupplySystem</li> <li>ARModels</li> <li>SimpleWaterSupplySystem.alt</li> </ul>	1	
Application Output		₽×
		 đ



#### A (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
```



#### A (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 'Tank' is equal to 'false'.

```
block SimpleWaterSupplySystem
Source S;
Pump P;
block Tank
Boolean input (reset = false);
end
assertion
P.input := S.output;
Tank.input := P.output;
observer Boolean TE_tankIsNotSupplied = Tank.input == false;
end
```





### B. To check and compile the model





## B. To check and compile the model

Launch the AltaRica 3.0 compiler:

- a. Click on 'Tool' -> 'Flattening';
- b. You can create a new output folder (e.g. 'GTSModels');
- c. Results are printed into the 'Application Output' part. A new file is created into the output folder (e.g. 'GTSModels').

AltaRicaWizard Editor		
File Edit Project Tool Window Help		
0 🖉 🗖 🗇 ≽ 🕅		
Projects 🗗 🗙	SimpleWaterSupplySystem.alt 🔀	
<ul> <li>FimpleWaterSupplySystem</li> <li>ARModels</li> <li>SimpleWaterSupplySystem.alt</li> <li>GTSModels</li> <li>SimpleWaterSupplySystem.gts</li> </ul>	11       end         12       class Source         14       extends Repare         15       Boolean output         16       assertion         17       output ::         18       end         19       olass Pump         20       class Pump         21       extends Repare         22       Boolean input         23       assertion         24       output ::         25       end         26       Source S;         29       Pump P;         30       block SimpleWate:         32       end         33       assertion         34       F.input         35       Tank.input         36       end         37       observer Bool         38       end         40	<pre>rableComponent; t (reset = false); if working then true else false; rableComponent; , output (reset = false); if working then input else false; SupplySystem nput (reset = false); = S.output; t := F.output; ean tankIsNotSupplied = Tank.input == false;</pre>
Application Output		5×
Flatten AltaRica Model: OK. Generate GTS Model: OK. Compile AltaRica model: OK gtatck version 1.0.2: GTS type-c	hecker	



A (Simple) Water Supply System



### C. To simulate the model



#### A (Simple) Water Supply System



## C. To simulate the model

- 1. Launch the stepwise simulator:
  - a. Click on 'Tool' -> 'Stepwise Simulation';
  - b. Click on 'Next' for the flattening part (it compiles the model if it has been modified) -> Click on 'Next' for the 'Stepwise Simulation' part;
  - c. A new window opens.

😵 AltaRicaWizard Stepper						
File Window Help						
Selection 🗗 🗙	Tree View Stepp	er Output			Sequence	₽×
<ul> <li>✓ Show variables</li> <li>Show events</li> <li>Show parameters</li> <li>✓ Show transitions</li> <li>✓ Show observers</li> <li>✓ Apply</li> </ul>	Element B SimpleWa B P F inj C ou S wc T fai T rep B S S wc T fai T rep A B S F ou S wc T fai T rep A B S S ou S wc T fai T rep A B S S wc S wc T fai T rep A B S S wc S	terSupplySyste but tput lure [0] bair [1] tput tput prking lure [2] bair [3] bair [3] but NotSupplied	Value m true true true fireable true fireable true fireable			
Control						₽×
Command:						
				Restart	Backtrack Ap	aly Exit



#### A (Simple) Water Supply System



## C. To simulate the model

#### 2. Launch commands to perform experiments:

- Select elements you want to see (variables, events, etc.);
- Click on fireable transitions to perform a simulation;
- You can backtrack or restart to a new simulation;
- You can run other commands and see results onto the 'Stepper Output' part (type 'help' and click on 'Apply' to see available commands).

- 0 × - 0 × 📌 AltaRicaWizard Stepper 🚀 AltaRicaWizard Stepper File Window Help File Window Help đΧ Tree View Stepper Output đΧ Β× Tree View Stepper Output Β× Selection Sequence Selection Sequence P.failure [0] P.failure [0] AR3Models/SimpleWaterSupplySystem/GTSModels/ Element Value Show variables Show variables S.failure [2] SimpleWaterSupplySystem.gts S.failure [2] SimpleWaterSupplySystem cwd C:/DataMB/OpenAltaRica/Development/AR3Models/ S.repair [3] S.repair [3] Show events Show events SimpleWaterSupplySystem ⊿ **B** P simulation started Show parameters Show parameters F input true GTSStepSim >> Show transitions Show transitions F output false Available commands: Show observers Show observers 5 working false h(elp)? print this debug help. failure [0] print the simulated GTS model. print gts Apply Apply print v(ariables)? print the variables and their values. T repair [1] fireable print o(bservers)? print the observers and their values. ⊿ **B** S print tr(ansitions)? print the transitions fireable in the current state. F output true print h(istory)? print the execution history. 5 working true fire <index> fire the transition number <index>. fire <name> fire the transition named <name>. failure [2] fireable cancel the last transition firing. back T repair [3] restart restart the simulation. echo <aString> print <aString> (it can contain spaces). ▲ B Tank set (no)? trace turn on/off the trace mode. F input set (no)? display turn on/off the display mode. false set mode <normal|advanced> turn the mode to normal or tankIsNotSupplied true advanced a(uit)? auit. GTSStepSim >> đΧ Control đΧ Control Command: Command: Restart Backtrack Apply Exit Restart Backtrack Apply Exit



#### A (Simple) Water Supply System



### D. To generate and study a fault tree





## D. To generate and study a fault tree

1. We consider the top event defined by the observer named 'TE\_tankIsNotSupplied' and meaning "There is no input flow to the tank"

```
block SimpleWaterSupplySystem
   Source S;
   Pump P;
   block Tank
      Boolean input (reset = false);
end
   assertion
      P.input := S.output;
      Tank.input := P.output;

   observer Boolean TE_tankIsNotSupplied = Tank.input == false;
end
```





## D. To generate a fault tree

- 2. Launch the compiler to fault trees and analyze the generated fault tree:
  - a. Click on 'Tool' -> 'Compilation into Fault Tree';
  - b. Click on 'Next' for the flattening part (it compiles the model if it has been modified) -> Click on 'Next' for the 'Compilation into Fault Tree' part;
  - c. Select the name of the generated fault tree (you can create a new output folder: e.g. 'FaultTrees') -> click on the case 'Launch Fault Tree Assessment' and select the top event you want to observer ('TE\_tankIsNotSupplied' with the value 'true') -> 'Next'

A new window opens

- Define the XFTA command file (into the created output folder e.g. 'FaultTrees') and select features you want to compute (only orders of minimal cutsets with results into the file 'FaultTrees/Results.txt') -> click on 'Create';
- e. Into the 'Calculations' part, click on 'Launch'.



#### A (Simple) Water Supply System



## D. To generate a fault tree

- 2. Open results:
  - a. Click on 'File' -> 'Open File or Project' -> 'Open File';
  - b. Select the file of results 'FaultTrees/Results.txt';

😵 AltaRicaWizard Editor			x
<u>File Edit</u> Project Tool Window Help			
0 🖉 🖥 Ӣ 🏃 🕅			
Projects 🗗 🗙	SimpleWaterSupplySystem.alt 🗵	Results.txt 🗵	
<ul> <li>SimpleWaterSupplySystem</li> <li>ARModels         <ul> <li>SimpleWaterSupplySystem.alt</li> <li>FaultTrees                  <ul> <li>SimpleWaterSupplySystem.o</li> <li>GTSModels</li> <li>SimpleWaterSupplySystem.o</li> <li>SimpleWaterSupplySystem.o</li> <li>SimpleWaterSupplySystem.o</li> <li>SimpleWaterSupplySystem.o</li> <li>SimpleWaterSupplySystem.o</li> <li>SimpleWaterSupplySystem.o</li></ul></li></ul></li></ul>	1 1 0.583555 0.3 2 2 0.583555 0.3 3	).5 S.failure ).5 P.failure	
Application Output			ēΧ
#Gates 1 #BE 2 #Modules 2 bye			• •
			at



A (Simple) Water Supply System



### E. To script simulation





## E. To script simulation

- 1. Create the script file for simulation:
  - a. Click on 'File' -> 'New File or Project' -> 'New File' (an untitled file opens);
  - b. Save this untitled file : click on 'File' -> 'Save File as...' (you can create a dedicated folder: e.g. 'Scripts') and save it (e.g. 'Sequences.seq');

REM: the dedicated folder and the script file are not listed to the project.







## E. To script simulation

- 2. Edit the script file for simulation:
  - a. Enter directly commands into this script file;
  - b. Launch the stepwise simulator and select the 'Stepper Output' part;
  - c. Type the command 'run' with the path to the script file ('run ./Scripts/Sequences.seq')

% AltaRicaWizard Ste	epper							x
File Window Help								
Selection	P ×	Tree View	Stepper Output		Sequ	ence	I	8
<ul> <li>✓ Show variables</li> <li>Show events</li> <li>Show parameters</li> <li>✓ Show transitons</li> <li>✓ Show observers</li> <li>✓ Apply</li> </ul>		C:/DataMB, AR3Toole/g Simple/Wate simulation s GTSStepSin GTSStepSin GTSStepSin GTSStepSin GTSStepSin GTSStepSin GTSStepSin GTSStepSin GTSStepSin GTSStepSin	(OpenAltaRica/OARP tsstp.exe C:/DataMi stupplySystem(GTSI stupplySystem(GTSI avM8/OpenAltaRica/C stupplySystem tarted > >> fre P.rapair > >> fre P.rapair > >> fre P.rapair > >> re (delay = exponer >>>	latforms/OARPlatform-win64-Custom/Tools/ 3/OpenIATRICa/Development/AR3Models/ Models/SimplevtareSupplySystem.gts evelopment/AR3Models/ ntal(0.0001)) ntal(0.0001))				
Control							1	8
Command:								
				Rest	art 🖣	Backtrack	Apply Exit	



# **CONTACTS**

#### The OpenAltaRica project

www.openaltarica.fr contact.oar@irt-systemx.fr

www.irt-systemx.fr









