TTool Training

III. Using TTool

Ludovic Apvrille

ludovic.apvrille@telecom-paris.fr

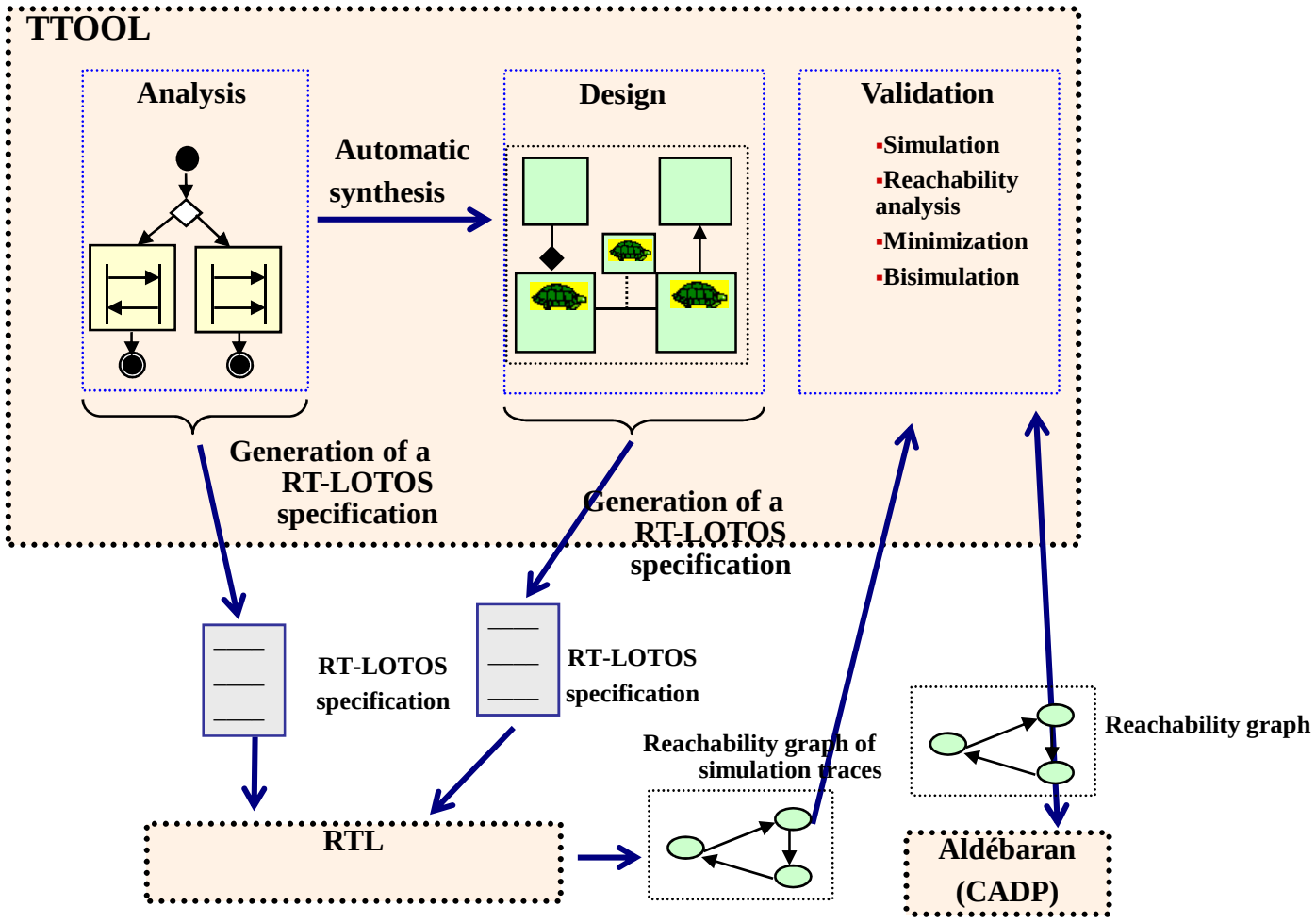
Eurecom, Office 223

I. Introduction

TELECOM PARIS
école nationale supérieure des télécommunications

- TTool: main features
- Installing TTool
- Diagramming with TTool
- Formal validation
- Java code generation

TTool: General Overview



I. Introduction

- TTool: main features
- Installing TTool
- Diagramming with TTool
- Formal validation
- Java code generation

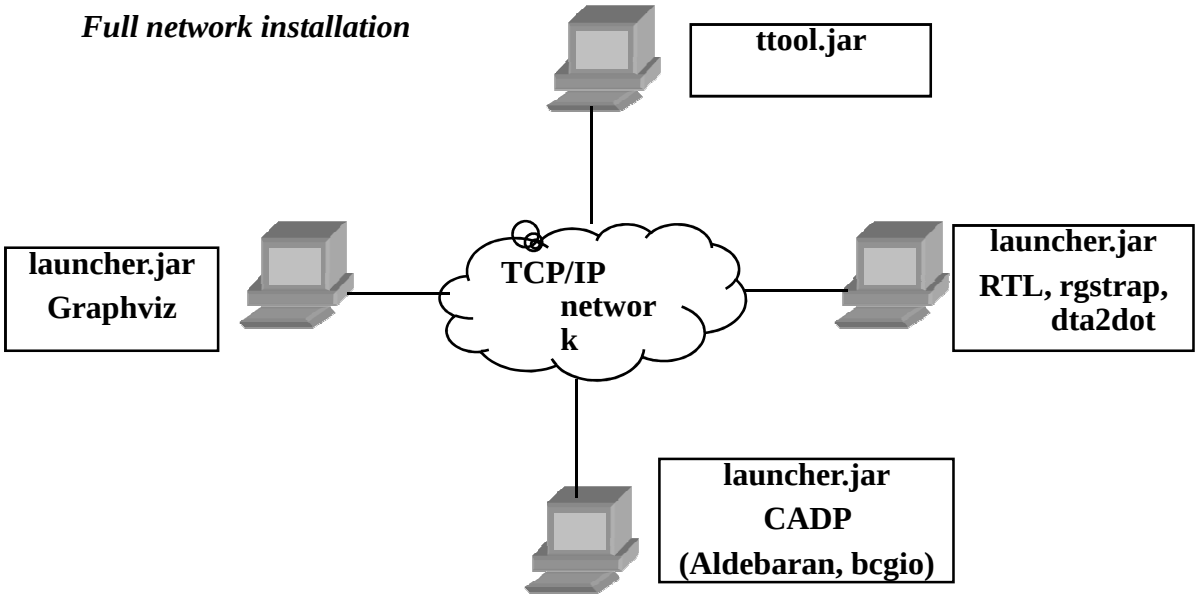
Inside a TTool package

- ttool.jar
- launcher.jar
- config.xml
- ttoolmanual.doc
- *.xml: examples (files into which ttool modeling are saved)
- *.lib: librairies (diagrams that can be included into other diagrams)

Starting TTool

- **JVM 1.4.2 or later must be installed on every machine running TTool or an external program**
- **Configuration of the xml file**
  - ***See next slide***
- **Starting of the main program from a terminal**
  - ***java –Xmx256m –jar ttool.jar MyConfig.xml***
- **The launcher must be started on each machine on which an external application is used (RTL, CADP/Aldébaran, Graphviz)**
  - ***java –jar launcher.jar***

Example of Configuration



Configuring config.xml

TELECOM PARIS
école nationale supérieure des télécommunications

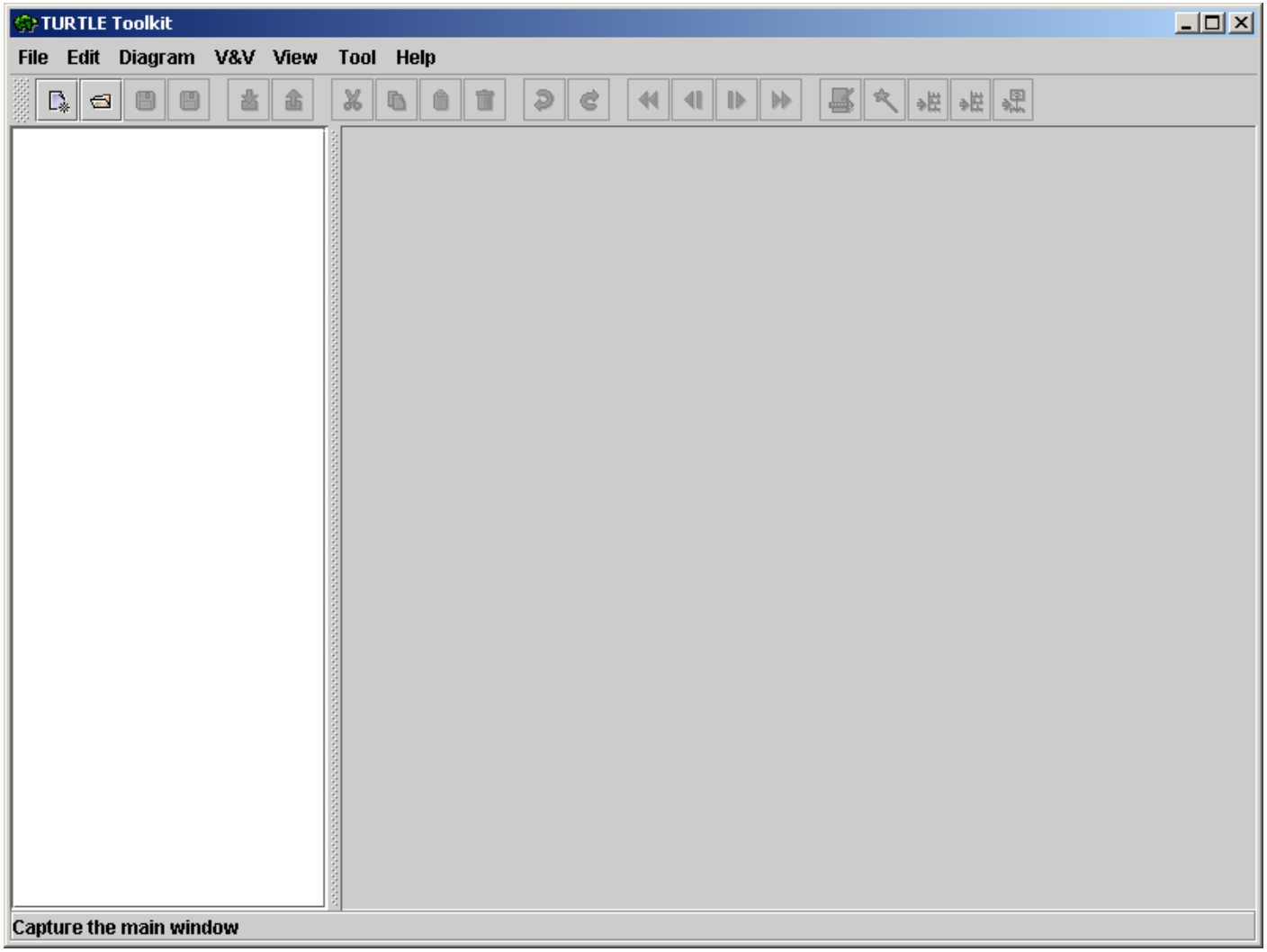
```
<RTLHost data="cow13.eurecom.fr" />
<RTLPath data="/homes/apvrille/RT-LOTOS/Linux/rtl/bin-linux/rtl" />
<DTA2DOTPath data="/homes/apvrille/RT-LOTOS/Linux/rtl/bin-linux/dta2dot" />
<RG2TLSAPath data="/homes/apvrille/RT-LOTOS/Linux/rtl/bin-linux/rg2tlsa" />
<RGSTRAPPath data="/homes/apvrille/RT-LOTOS/Linux/rtl/bin-linux/rgstrap" />
<DOTTYHost data="chaland.eurecom.fr" />
<DOTTYPath data="/homes/apvrille/softs/gv1.7c/bin/dotty" />
<AldebaranHost data="cow13.eurecom.fr" />
<AldebaranPath data="/homes/apvrille/cadp/bin.iX86/aldebaran" />
<BcgioPath data="/homes/apvrille/cadp/bin.iX86/bcg_io" />
<FILEPath data="U:\RT-LOTOS\TURTLEModeling" />
<LIBPath data="U:\RT-LOTOS\TURTLELibrary" />
<IMGPath data="U:\Figure" />
<LOTOSPath data="U:\RT-LOTOS\TURTLEModeling" />
<GGraphPath data="U:\RT-LOTOS\TURTLEGraphs" />
<TGraphPath data="U:\RT-LOTOS\TURTLEGraphs" />
<TToolUpdateURL data="http://www.eurecom.fr/~apvrille/TURTLE/ttoolversion.html" />
<TToolUpdateProxy data="false" />
<TToolUpdateProxyPort data="8080" />
<TToolUpdateProxyHost data="To Be Completed" />
```

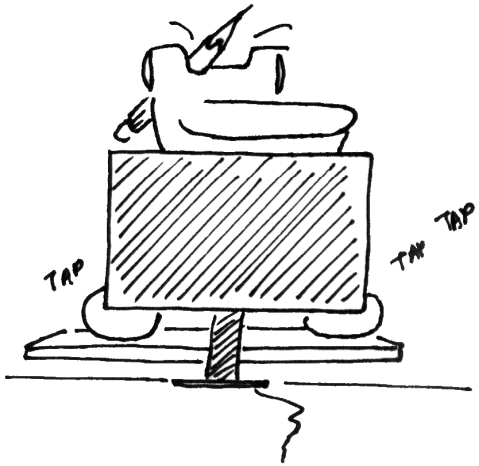
I. Introduction

- TTool: main features
- Installing TTool
- Diagramming with TTool
- Formal validation
- Java code generation

Starting TTool

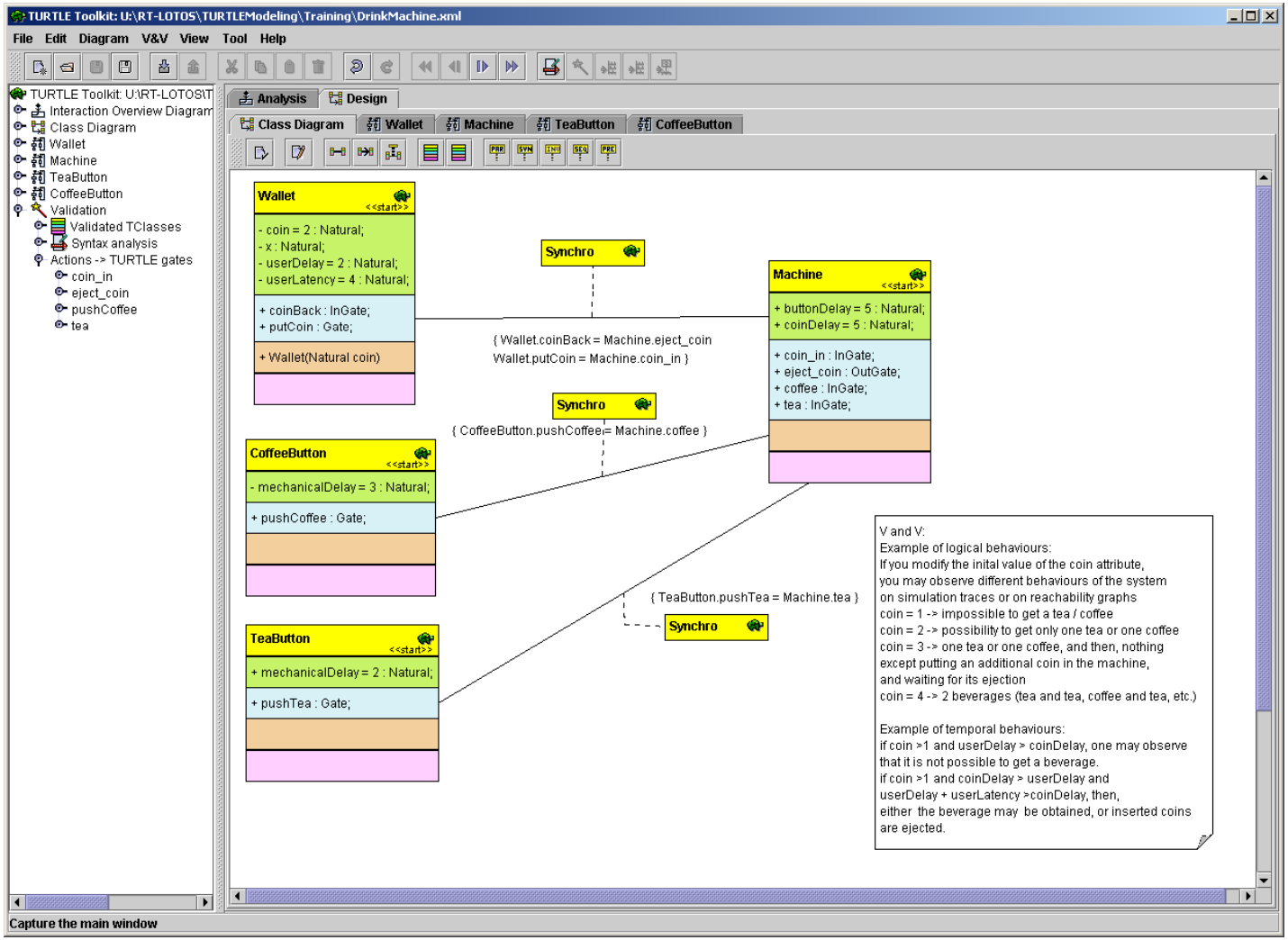
TELECOM PARIS
école nationale supérieure des télécommunications



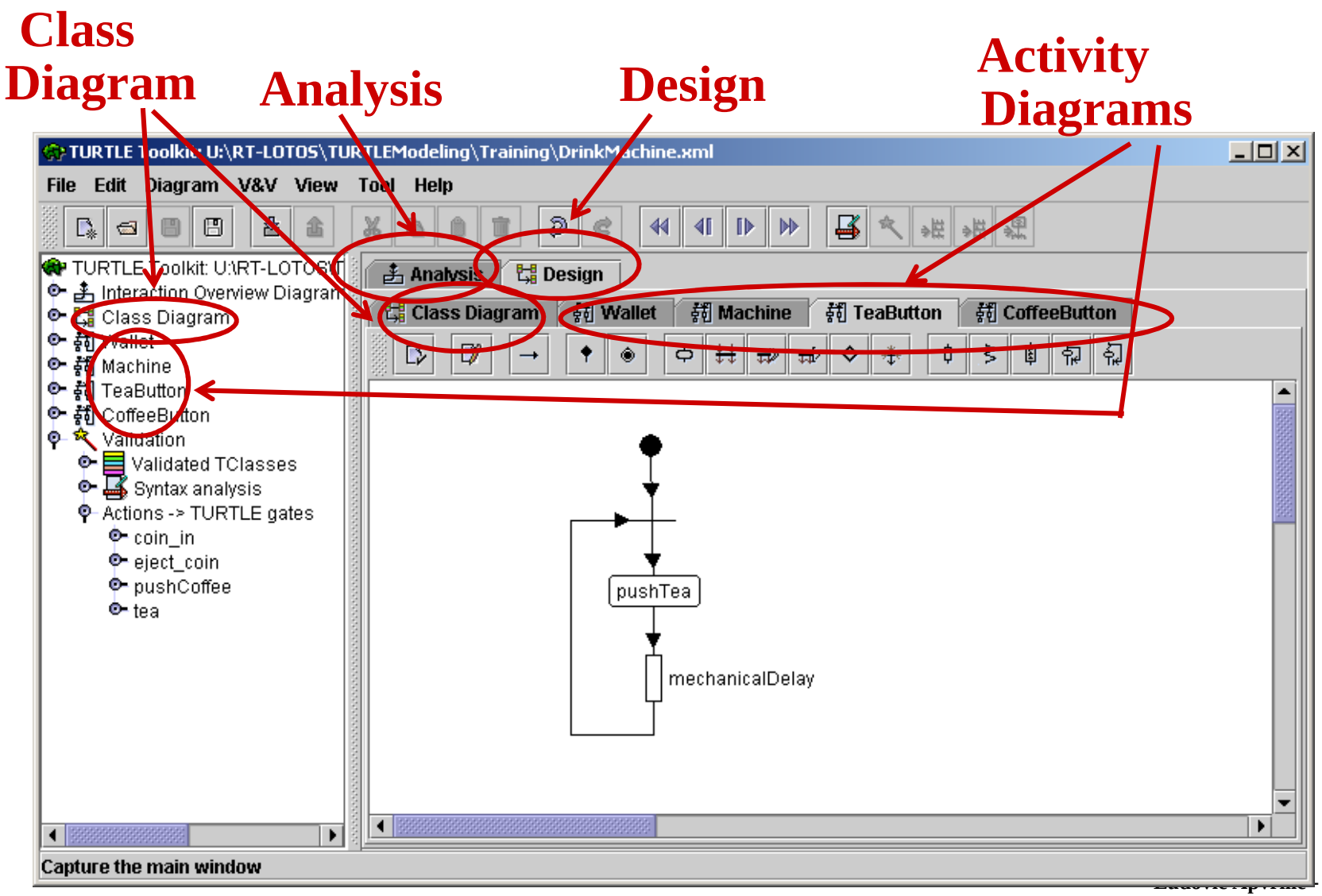


Opening a TURTLE Modeling

- Menu File, Open, and then select the desired modeling



Navigating into Diagrams

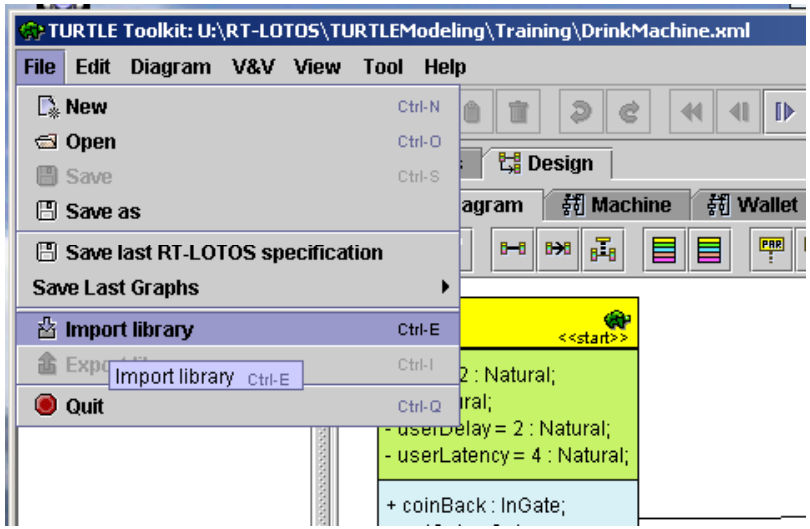


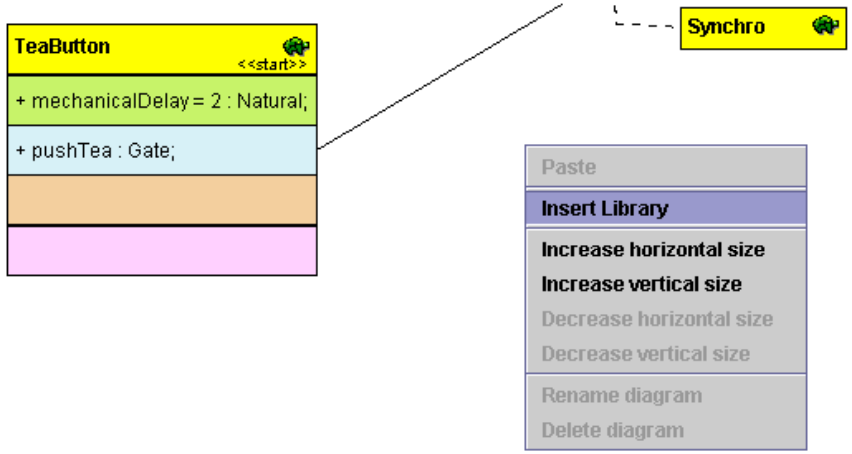
Dealing with Libraries

TELECOM PARIS
école nationale supérieure des télécommunications

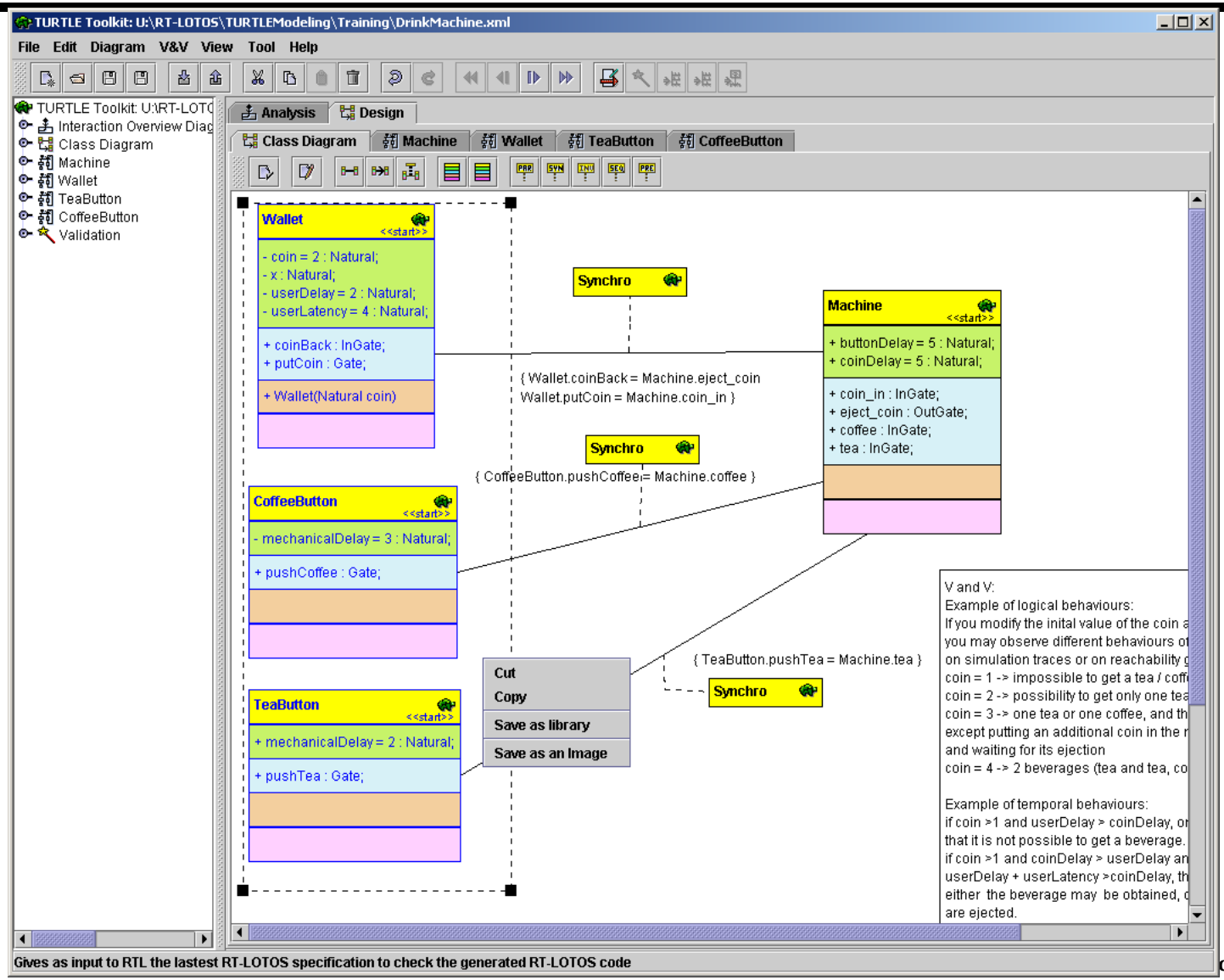
- **Libraries = file in which parts of diagrams are stored**
- **Example: classes, activity diagrams, etc.**
- **From TTool:**
  - ***Creation of libraries***
    - **Select components**
    - **Menu File, "Export library"**
  - ***Inclusion of libraries***
    - **Put the mouse pointer in the diagram in which you wish to import the library and Make a right click, select "Insert library"**
    - **Or, menu File, "Import library"**
- **Only way to copy/paste diagrams from a modeling to another**

Dealing with Libraries: Example





Creating a Library



I. Introduction

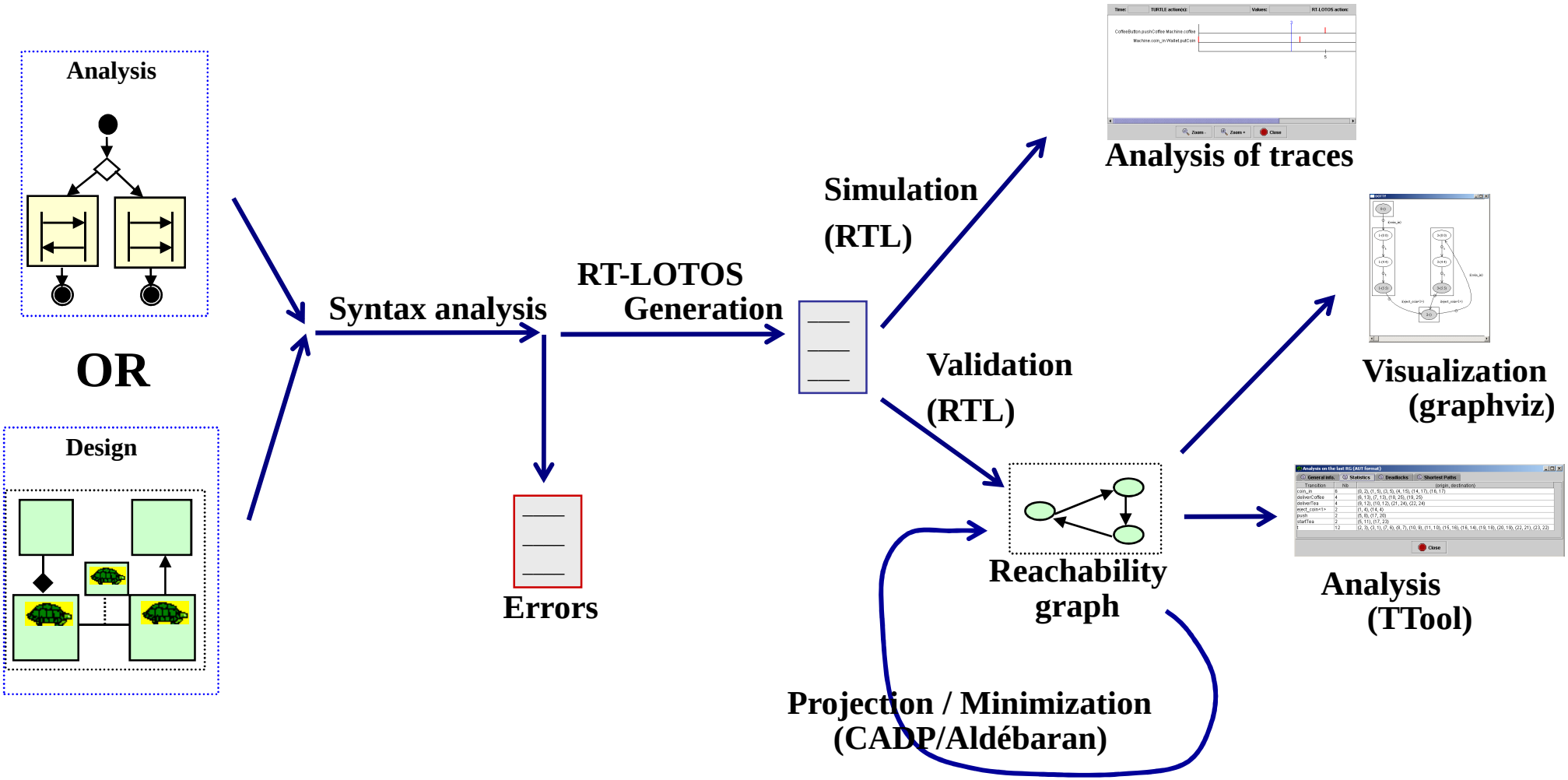
- TTool: main features
- Installing TTool
- Diagramming with TTool
- Formal validation
- Java code generation

What is Validation?

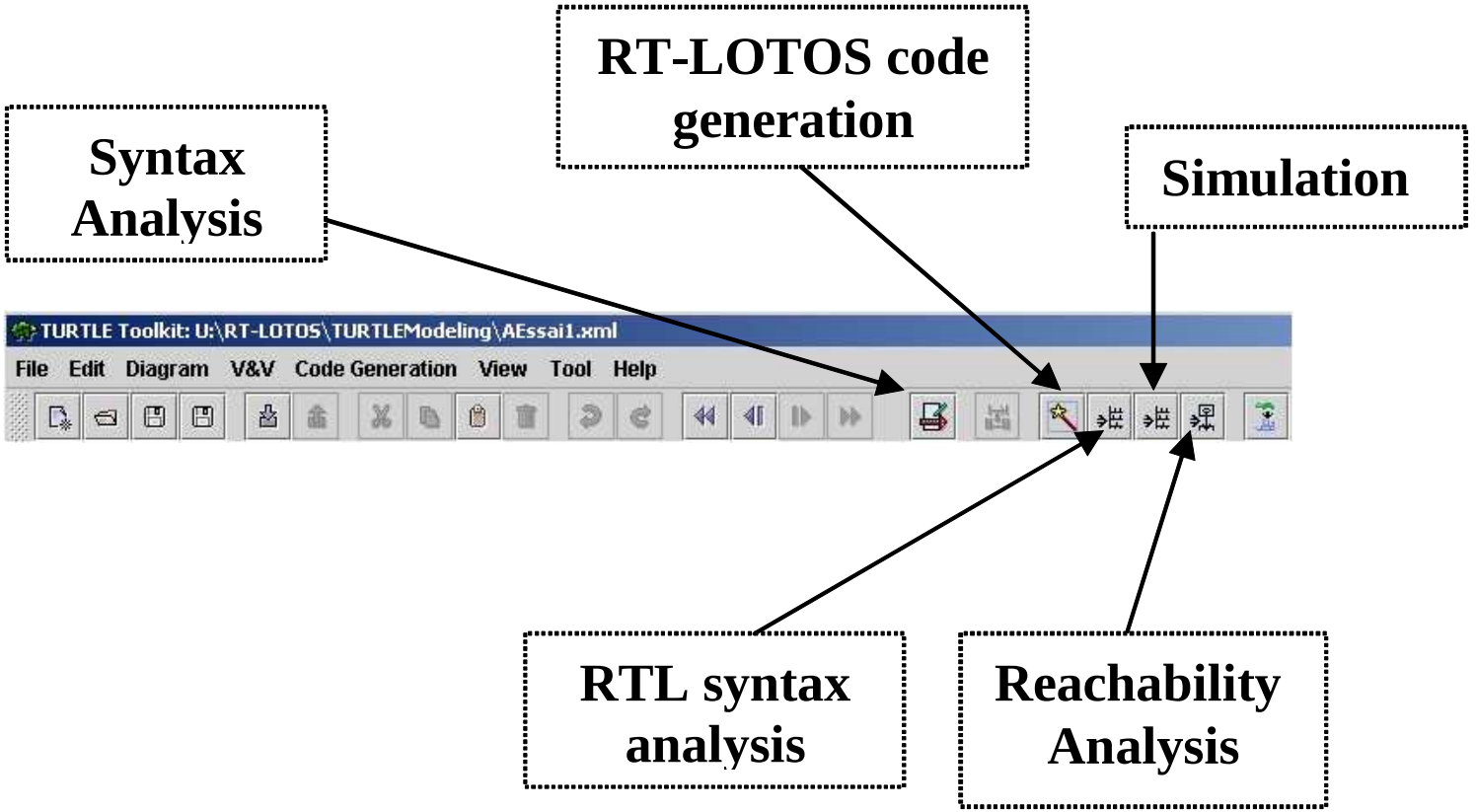
TELECOM PARIS
école nationale supérieure des télécommunications

- **Simulation**
  - *Partial exploration of the system*
  - *Used for debugging purposes*
  - *Increases confidence in a design when the system' state space cannot be explored exhaustively*
- **Validation / Verification**
  - *For bounded systems of reasonable size, exhaustive analysis becomes possible*
  - *Verification relies on the whole exploration of the system's state space in order to demonstrate deadlock freeness and other properties that should be satisfied by any system*
  - *Validation also relies on exhaustive analysis to demonstrate that a model meets specific requirements*

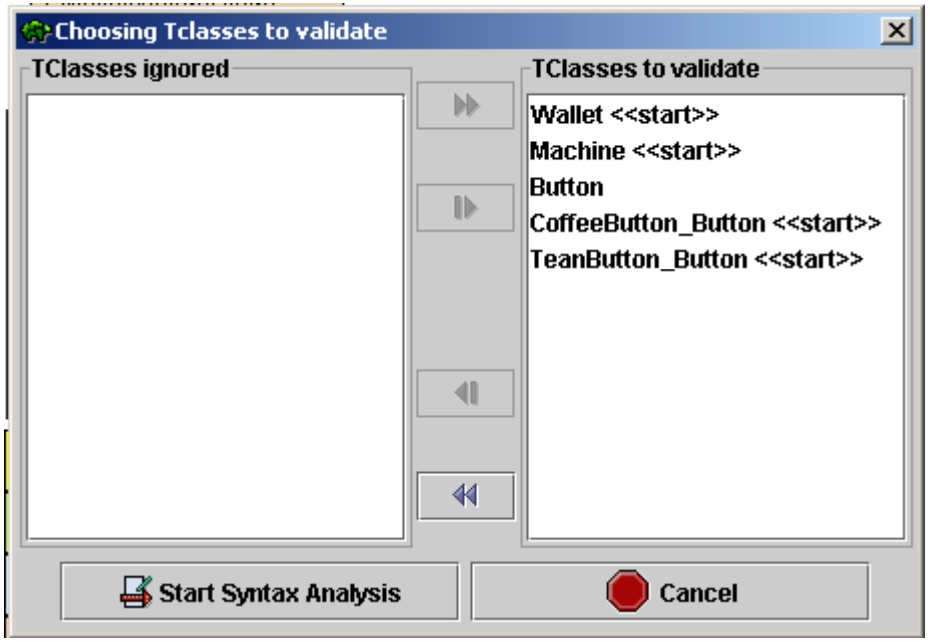
Simulation / Validation Process

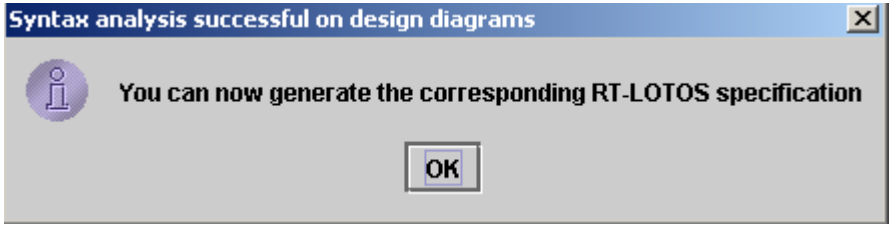


Simulation / Validation Process (Cont.)

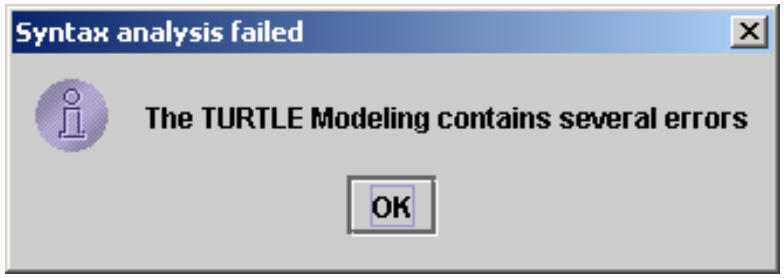


Syntax Analysis (From a Design)

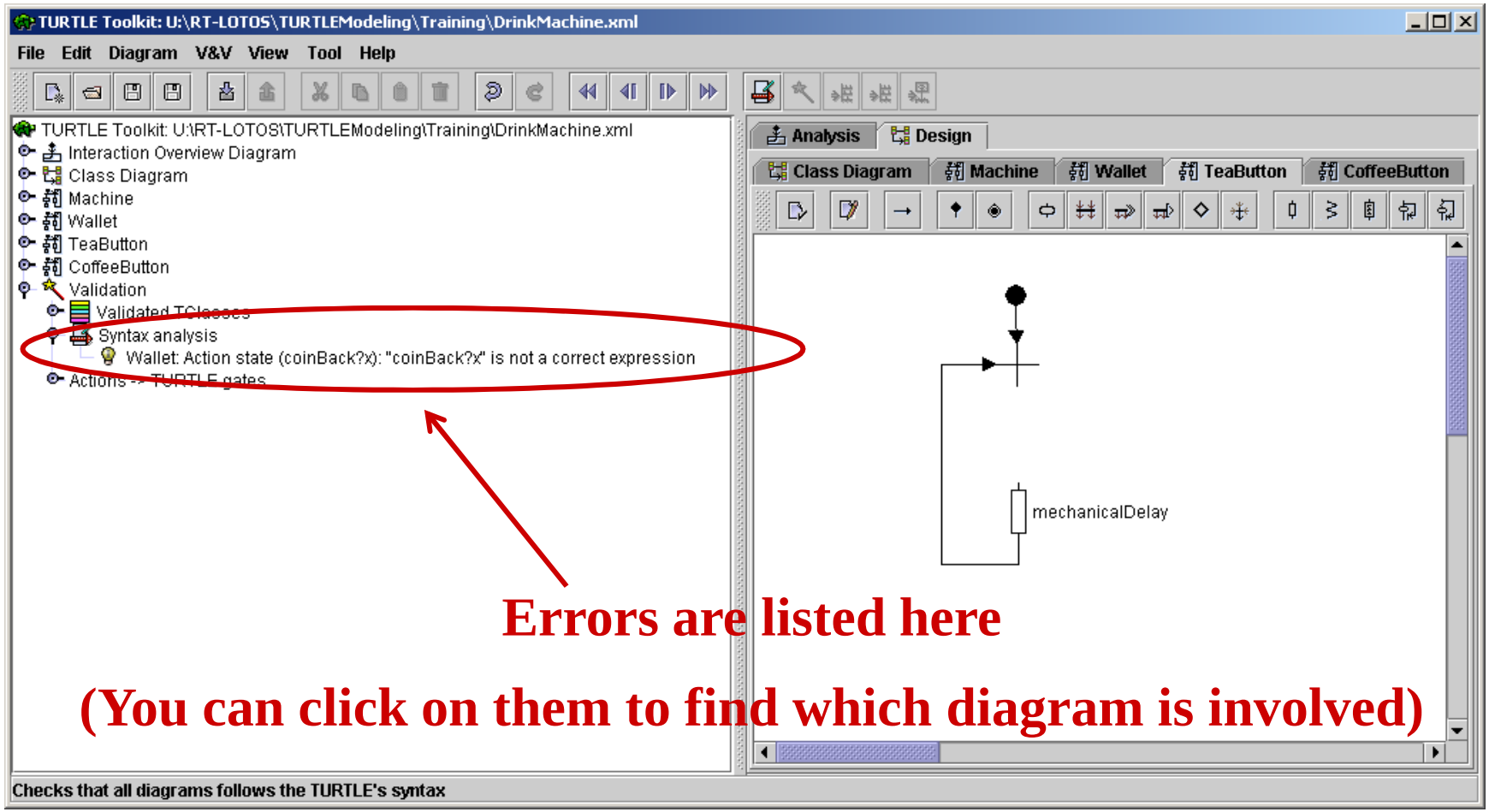




OR



Syntax Analysis (From a Design): Errors



Errors are listed here

(You can click on them to find which diagram is involved)

Generation of an RT-LOTOS Specification

- **Click on the magic stick**

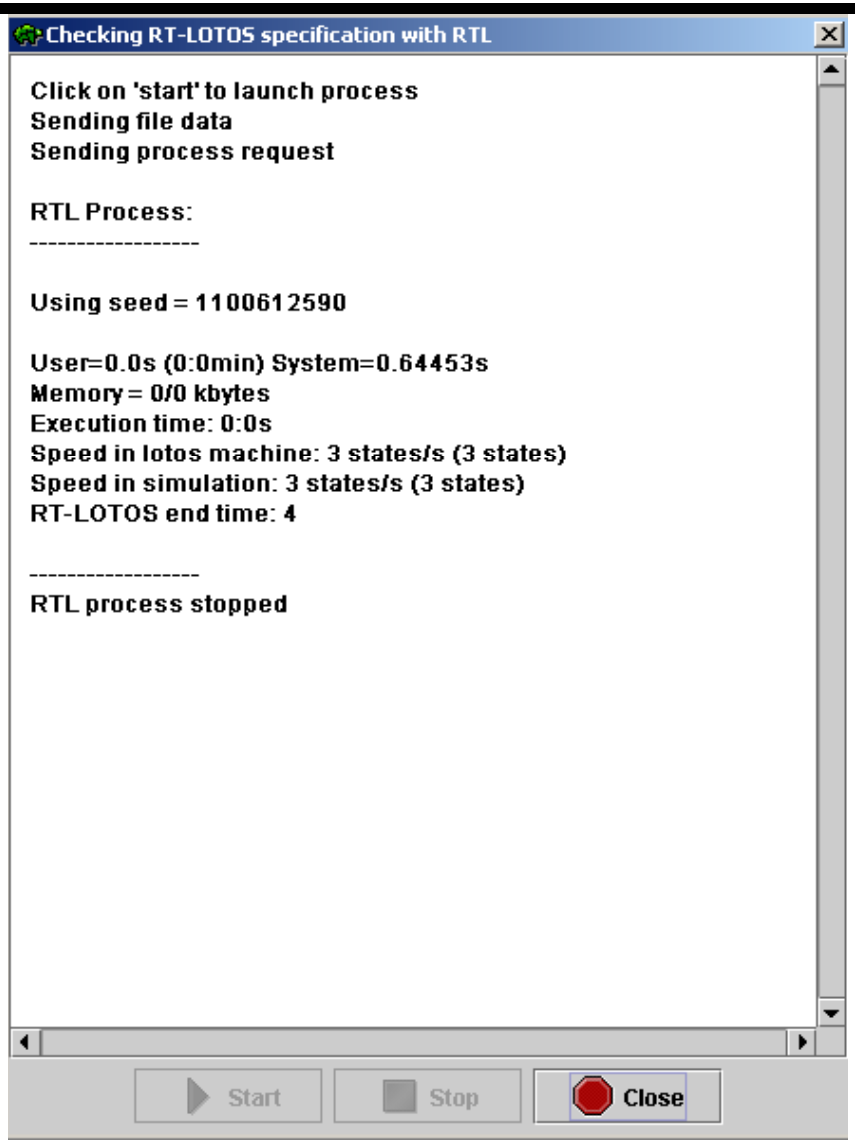
- **You can visualize the RT-LOTOS specification if needed, and save it from**
  - ***Menu "View"***
  - ***"Show last RT-LOTOS specification"***
  - ***Advice: for you, this visualizing this specification is useless***

RT-LOTOS Code Checking

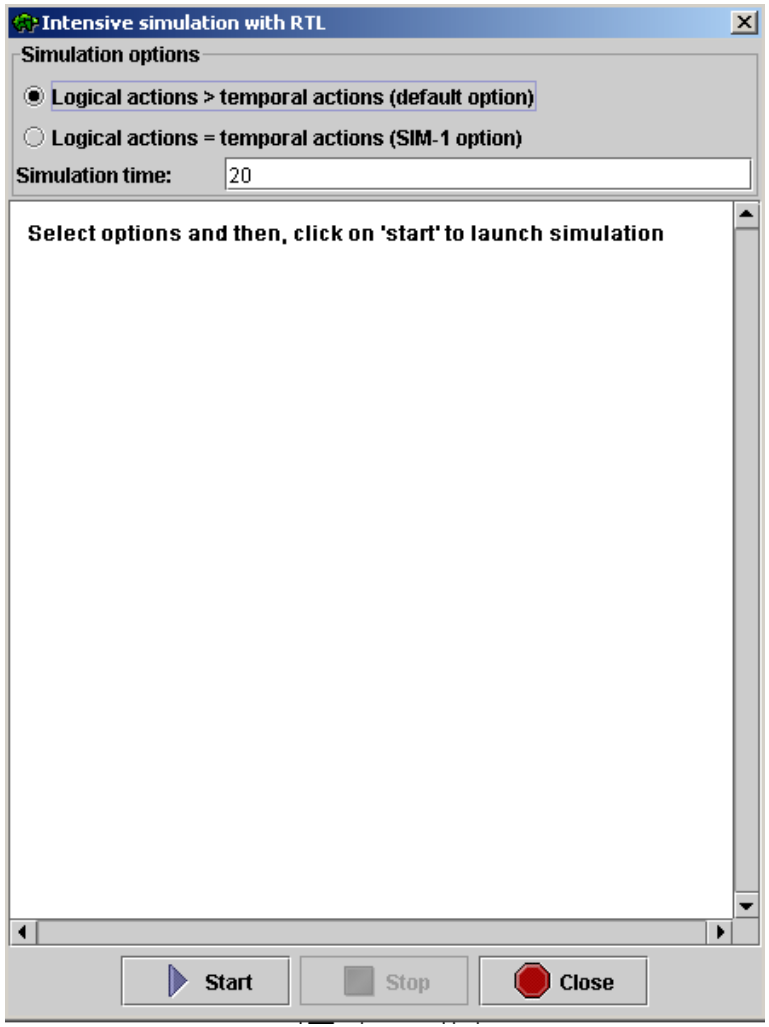
TELECOM PARIS
école nationale supérieure des télécommunications

- The syntax analysis of TURTLE models that you applied before generating RT-LOTOS code avoids the most common errors.
- Nevertheless, the current version of TTool does not implement a full syntax analysis.
- Before starting simulation or a reachability analysis, it is advisable to use the Check RT-LOTOS Code icon. The code is analyzed by RTL.
- This takes the form of a one second simulation with RTL

RT-LOTOS Code Checking (Cont.)



Simulation



Simulation options

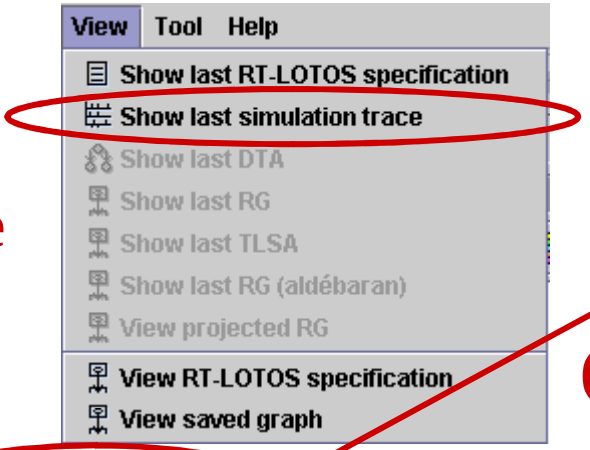
Maximum number of time units used for simulation

Information about the simulation with RTL, and results

Analyzing Simulation Traces

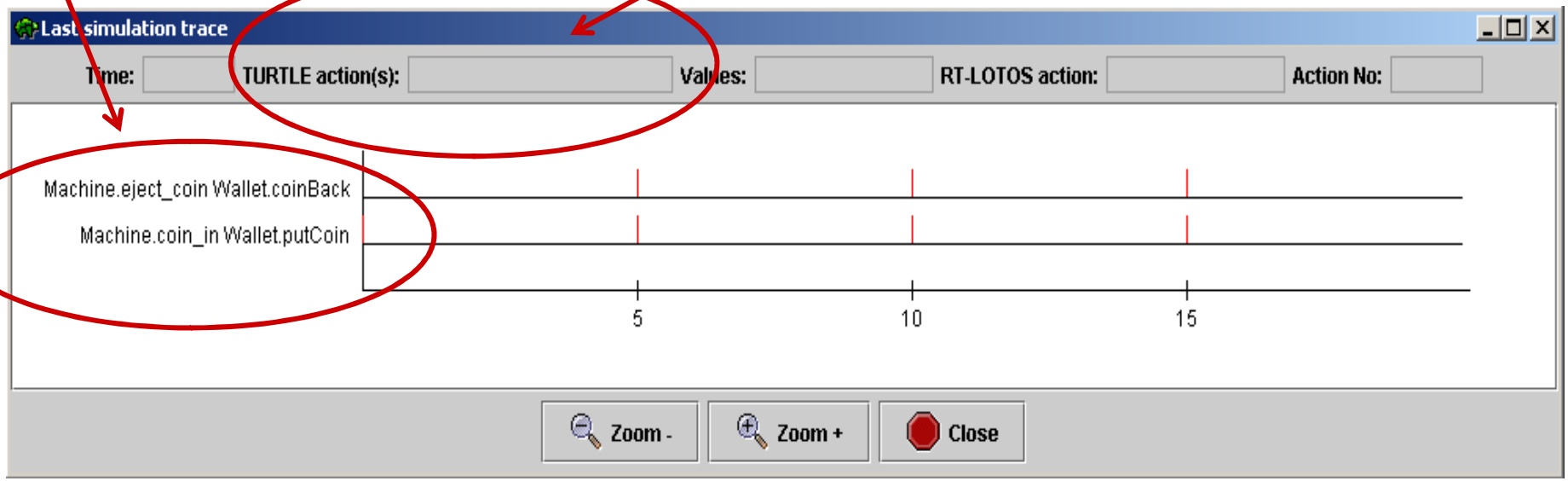
TELECOM PARIS
école nationale supérieure des télécommunications

On the left, you can see a list of paired gates

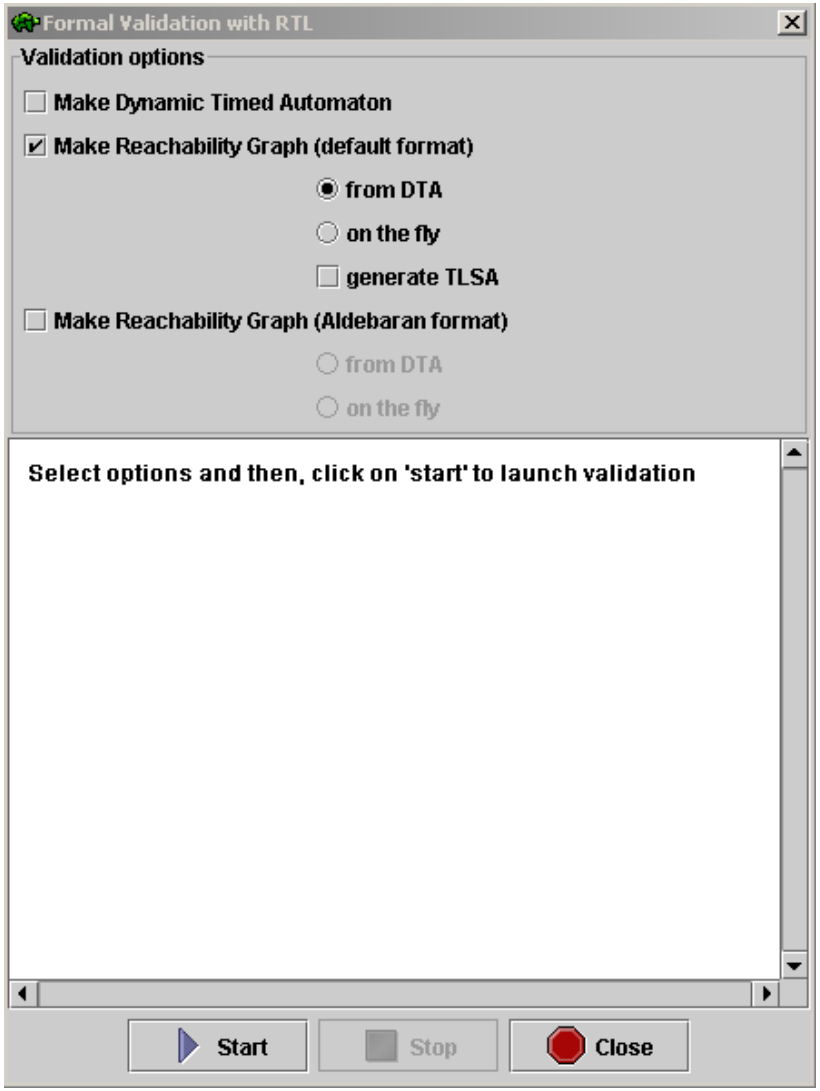


Corresponding TURTLE actions

(when a click is made on a dirac)



Generating a Reachability Graph (RG)

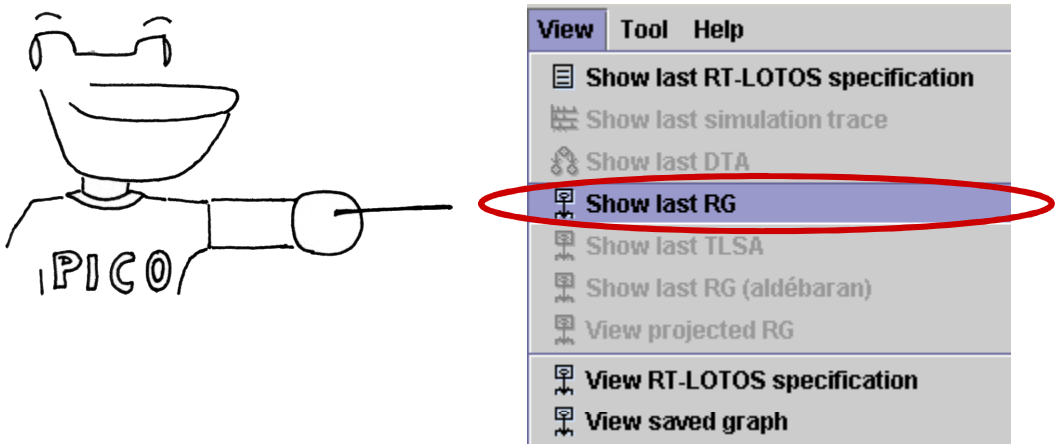


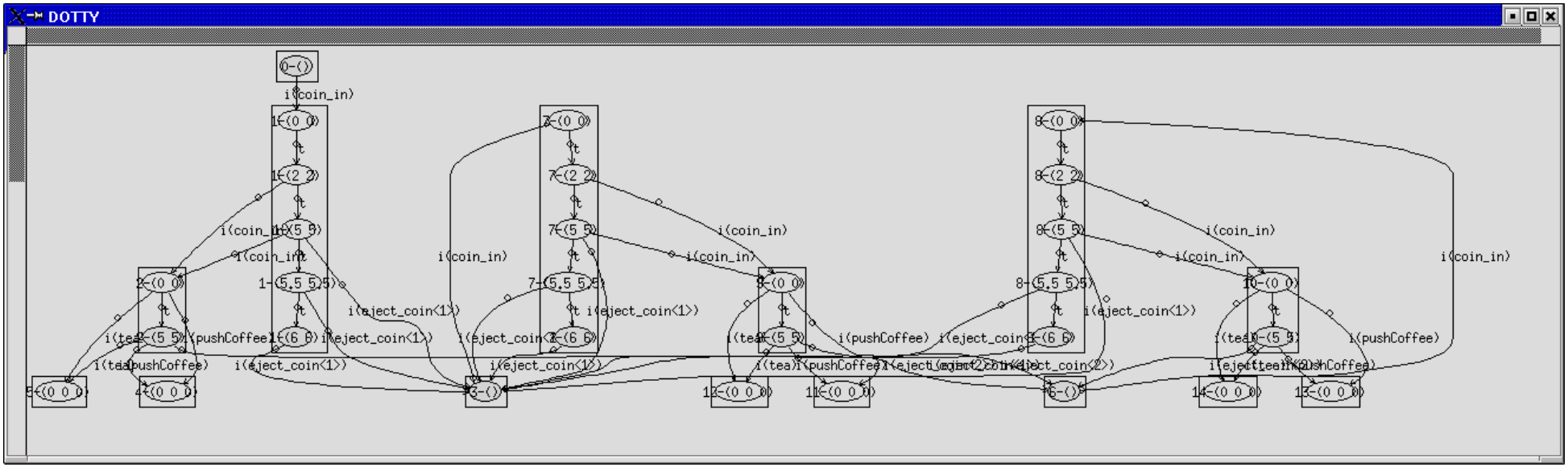
Graph with temporal information

Graph without temporal information. Only graph that can be used as input for projection / minimization

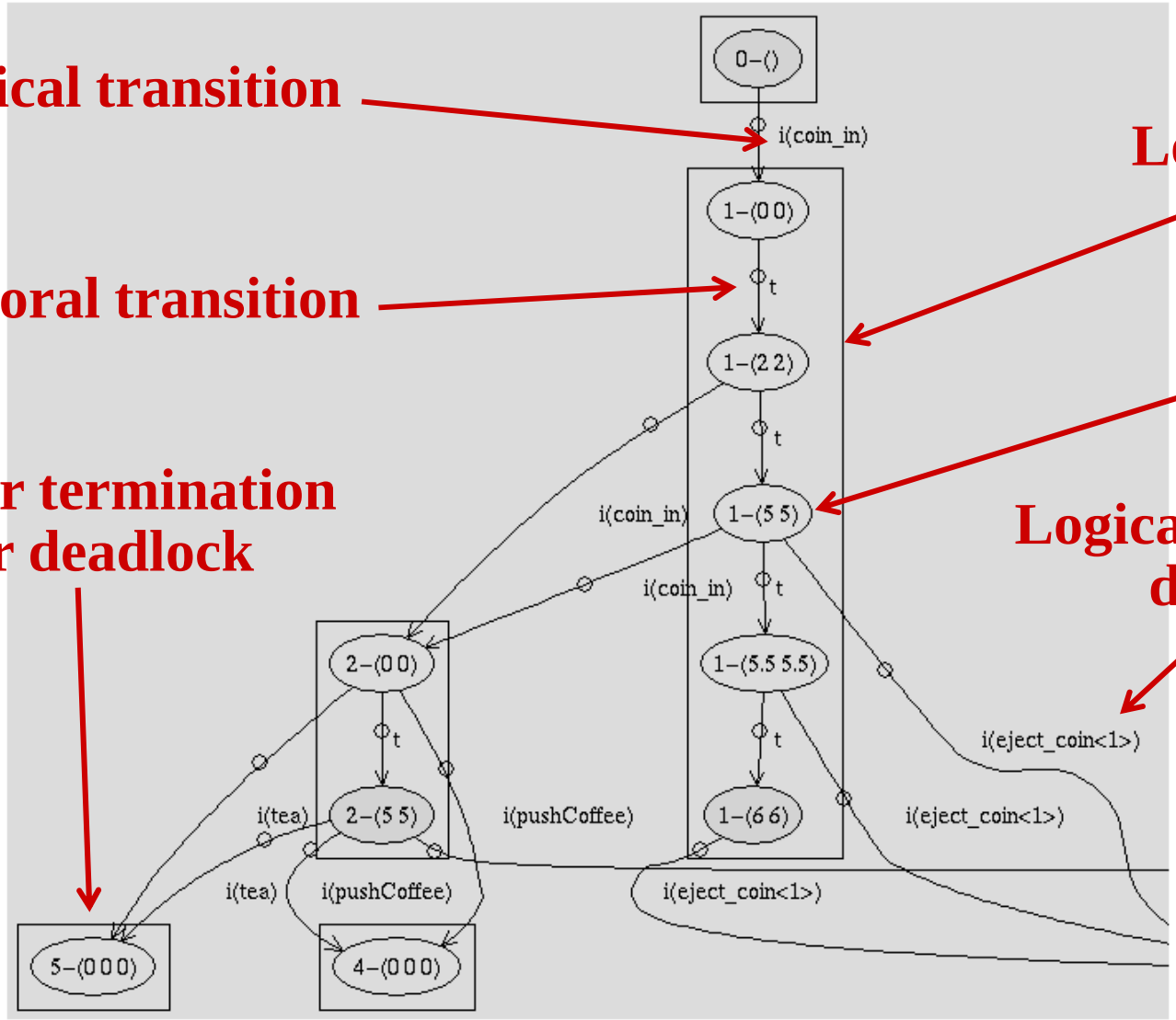
Information about the generation of RG with RTL, and results

Visualization of Graphs (Default Format)

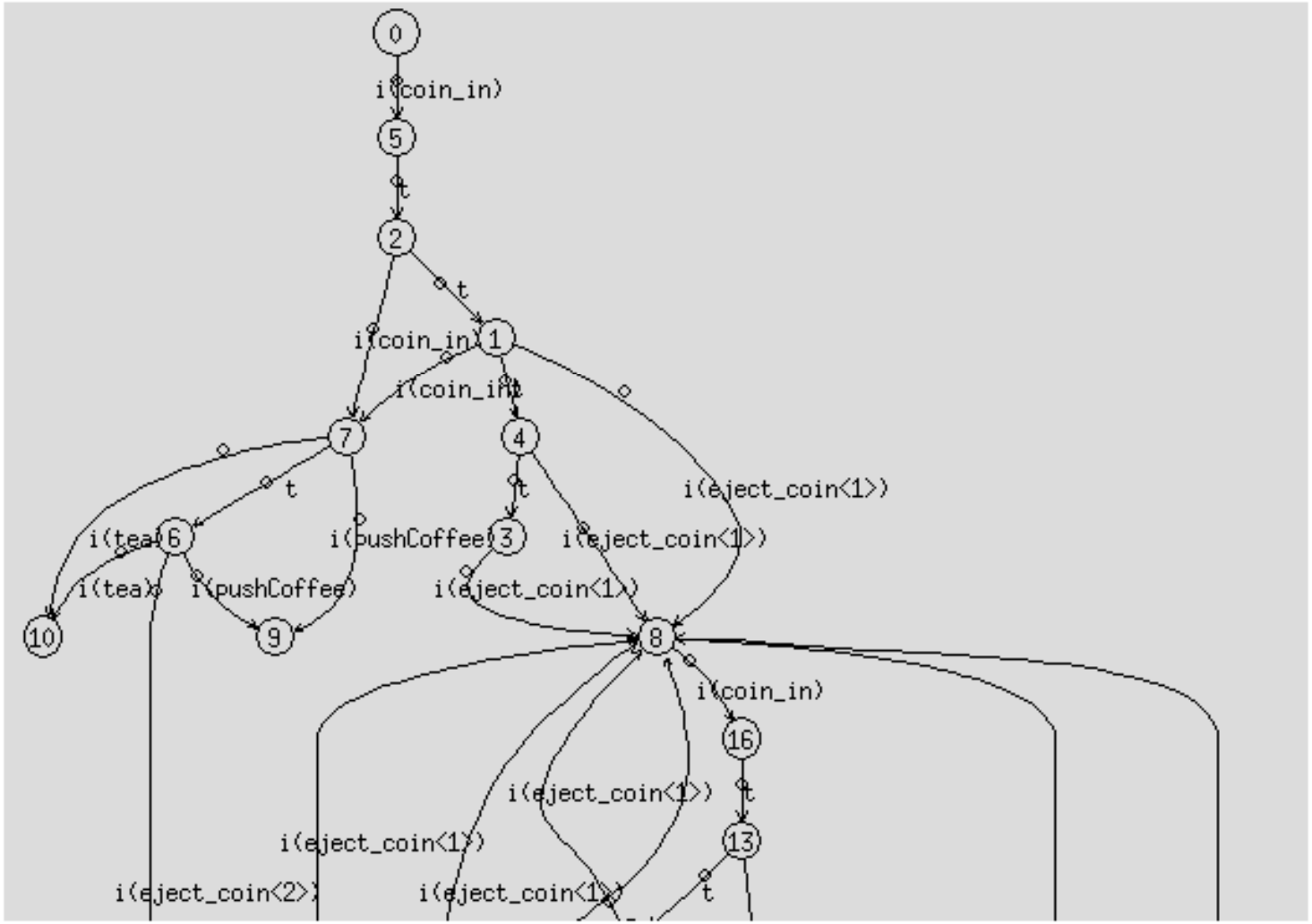




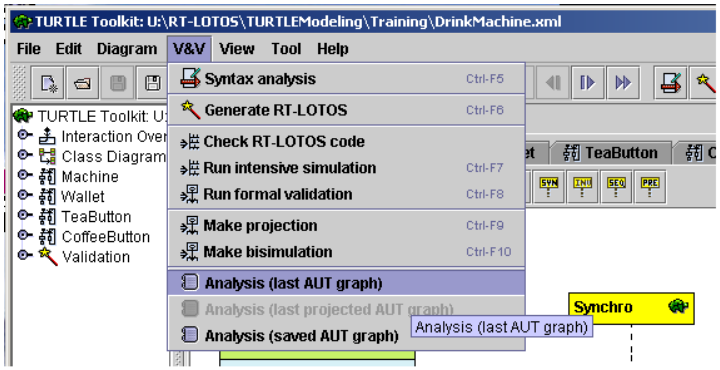
Visualization of Graphs (Cont.)

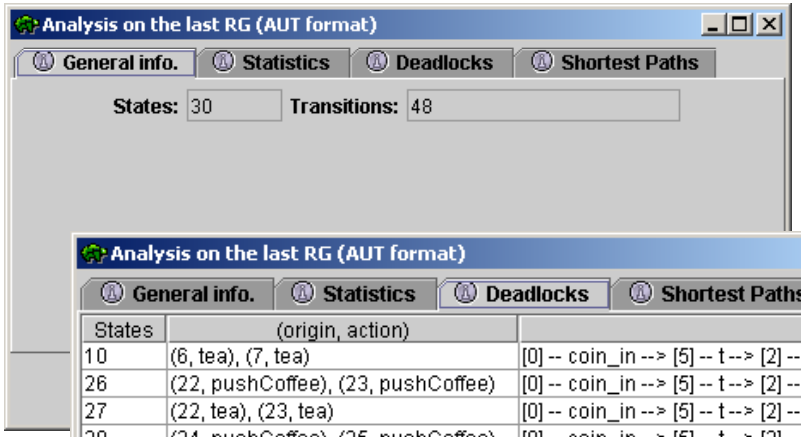


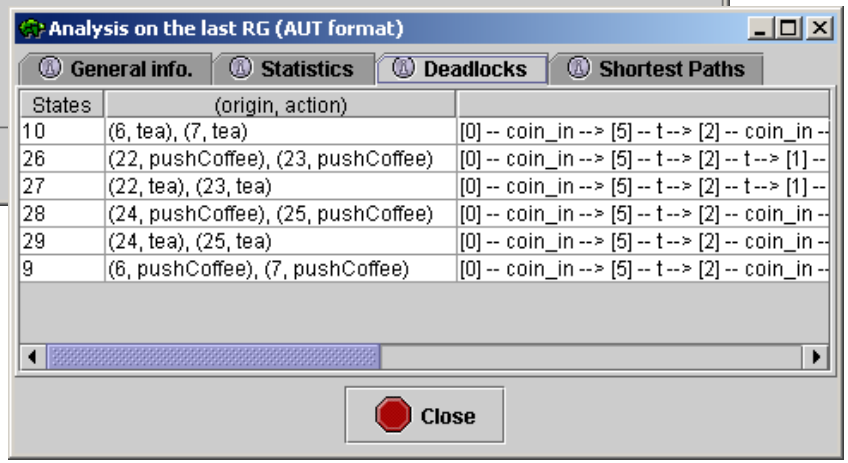
Visualization of Graphs: AUT Format

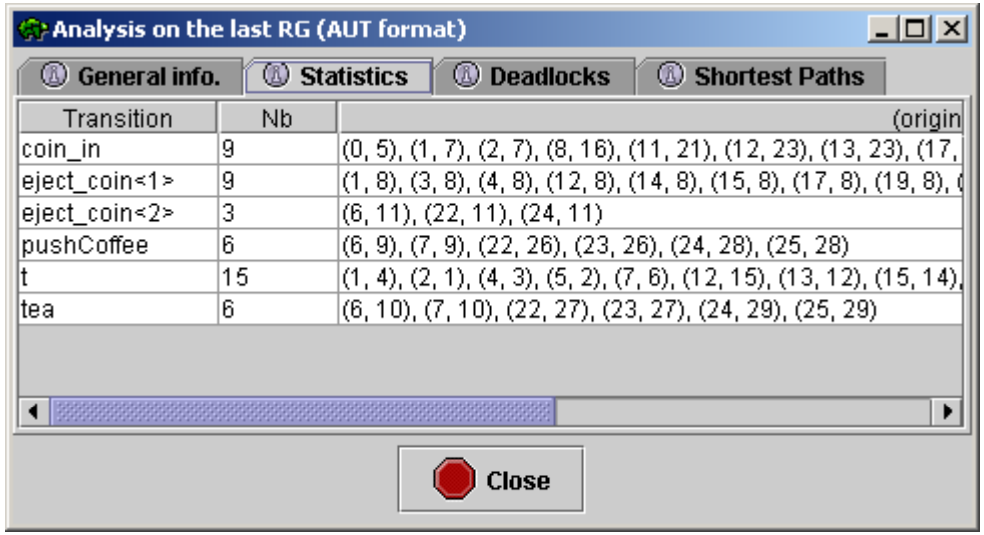


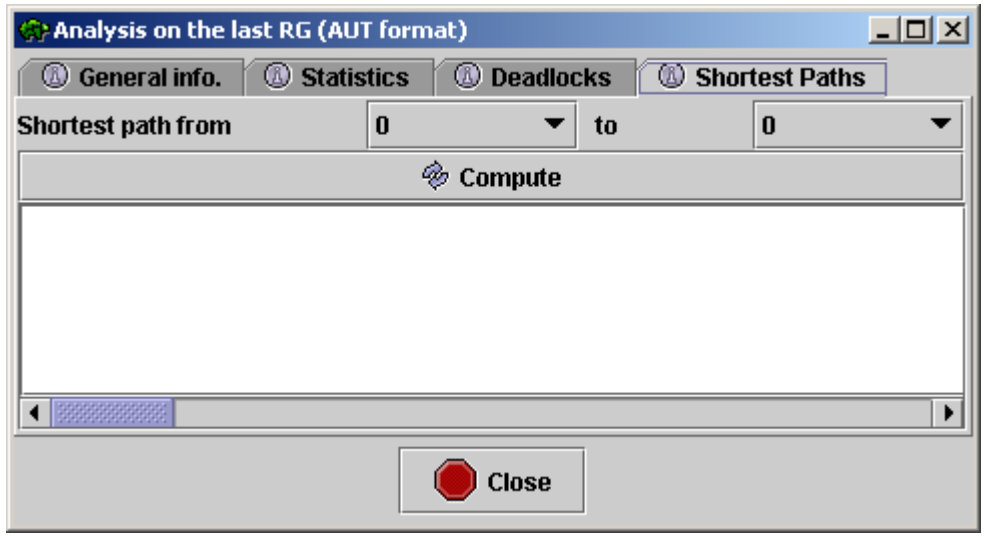
Analysis of Graphs: AUT Format





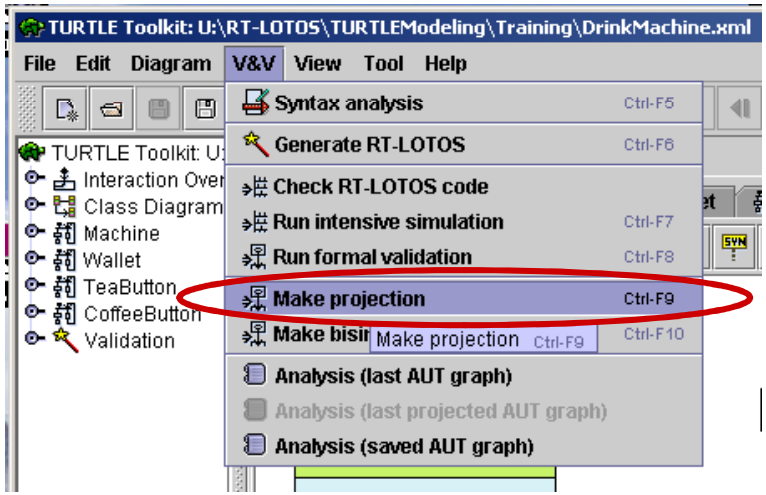




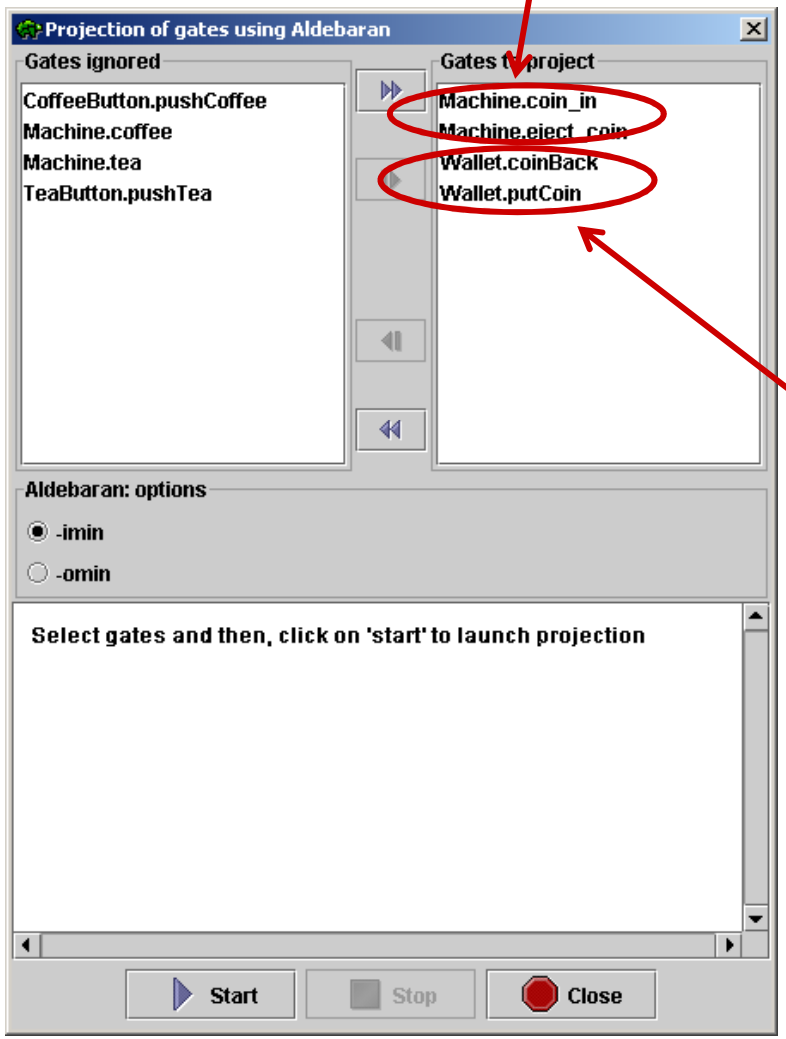


Projections (Aldebaran)

- Reduce the graph to a set of actions
- Minimize the graph



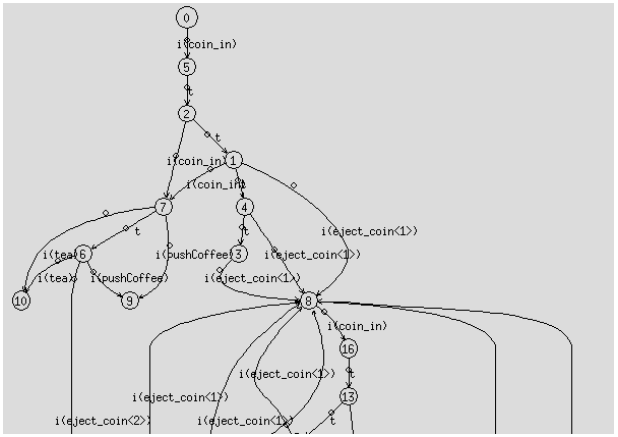
Gates synchronized with the selected gates

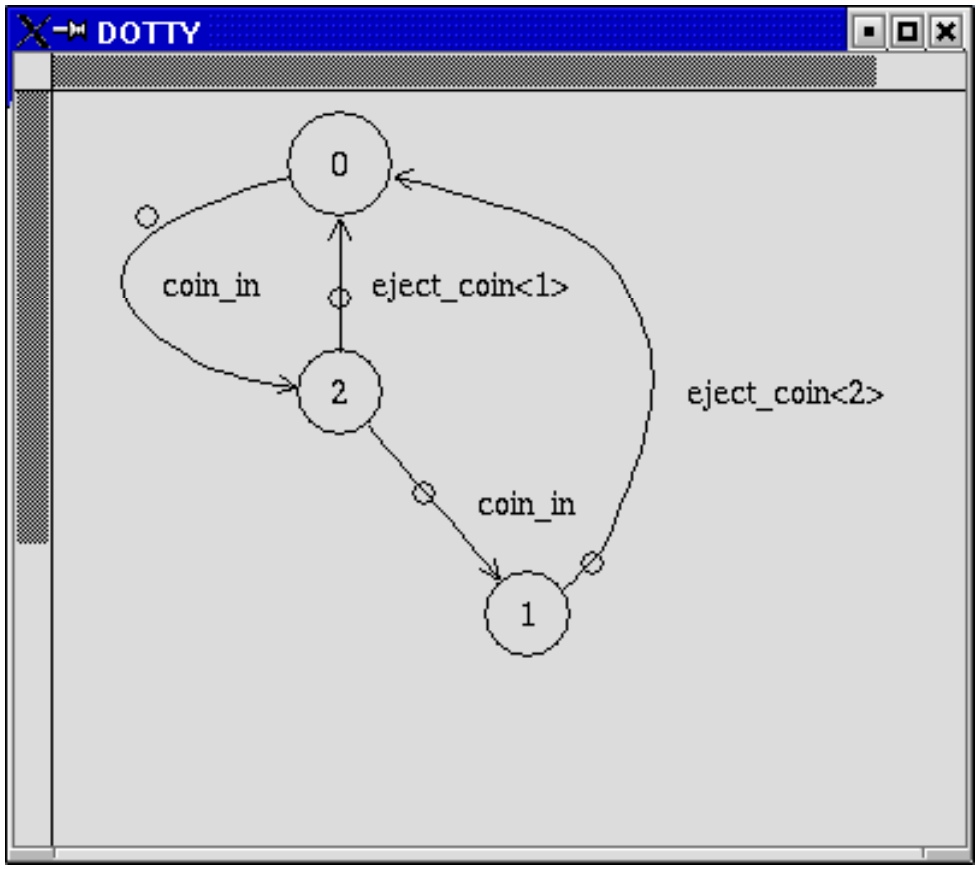


Selected gates

Projected Graph

Projection and Minimization



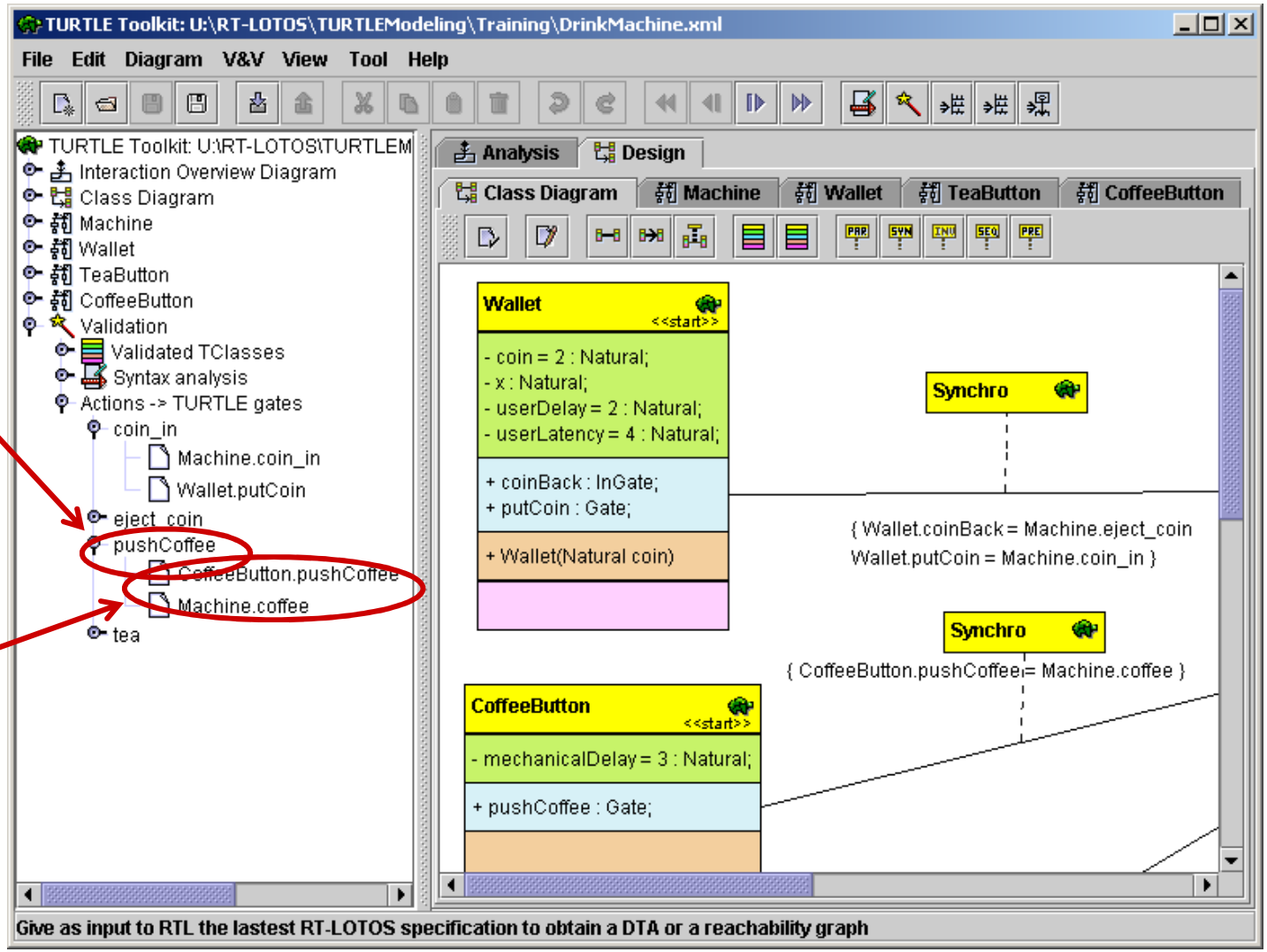


From Graphs to TURTLE Actions

TELECOM PARIS
école nationale supérieure des télécommunications

Action on the graph

Corresponding TURTLE actions

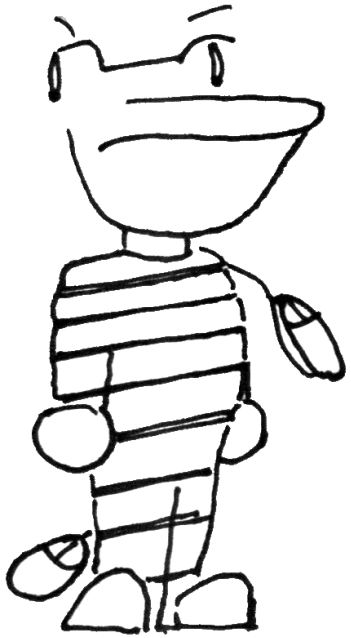


Common Modeling Errors

- Infinite loops
- Number of actions generated at t=0 is infinite
- -> Simulation cannot be performed

- Use of many attributes / variables
- Use of asynchronous messages (analysis)
- Use of many non-deterministic delays and choices
- -> Generation of reachability graph is difficult / long (combinatory explosion)

- Non-implementable scenarios (analysis)
  - *-> Abnormal deadlock situations or nothing particular except that traces on the graph / simulation traces do not match specified traces*

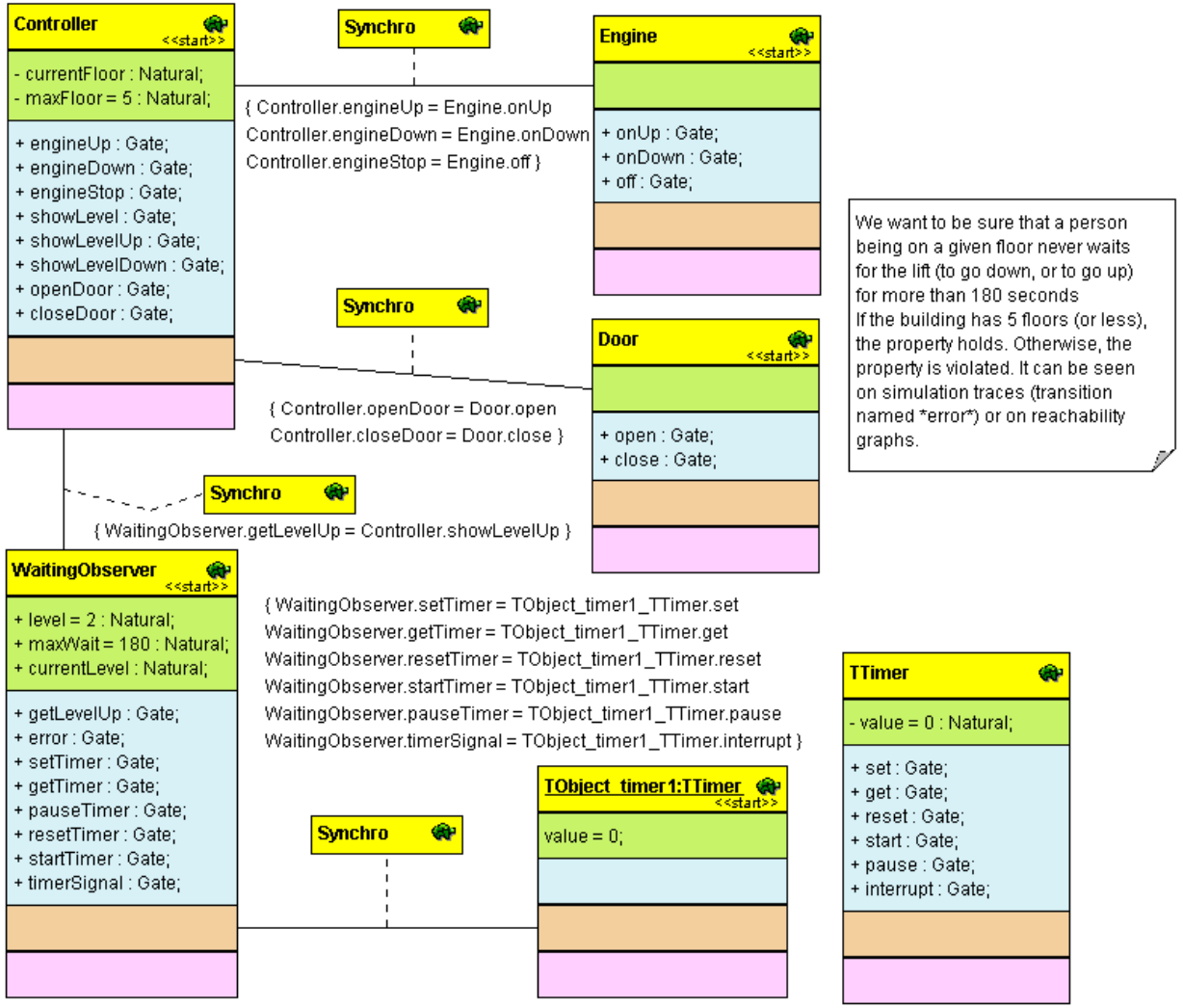


Observing Properties

TELECOM PARIS
école nationale supérieure des télécommunications

- **Model checking**
  - ***kronos***
- **Observers**
  - ***Avoid generating the full graph -> cut branches before they are generated***
  - ***Modeled as Tclasses at class diagram level***
  - ***Should be non intrusive***
  - ***Should not induce an important overhead when generating the graph***
    - **Keep them simple!**
  - ***Should stop the application under verification when a property violation is detected***
    - **“error” transition**
    - **Observer should synchronize with other classes to stop them**

Example of Property Observation



I. Introduction

- TTool: main features
- Installing TTool
- Diagramming with TTool
- Formal validation
- Java code generation

Fundamentals of Java Code Generation

TELECOM PARIS
école nationale supérieure des télécommunications

- **Java code may be generated from**
  - ***TURTLE designs***
    - **Monolithic application -> 1 executable application**
  - ***TURTLE deployments***
    - **Distributed application -> 1 executable application / execution node**
- **Steps**
  - ***Configuration of TTool***
  - ***Adding of directives***
  - ***Generation of code***
  - ***Compilation of code***
  - ***Execution of code***

Configuration of TTool

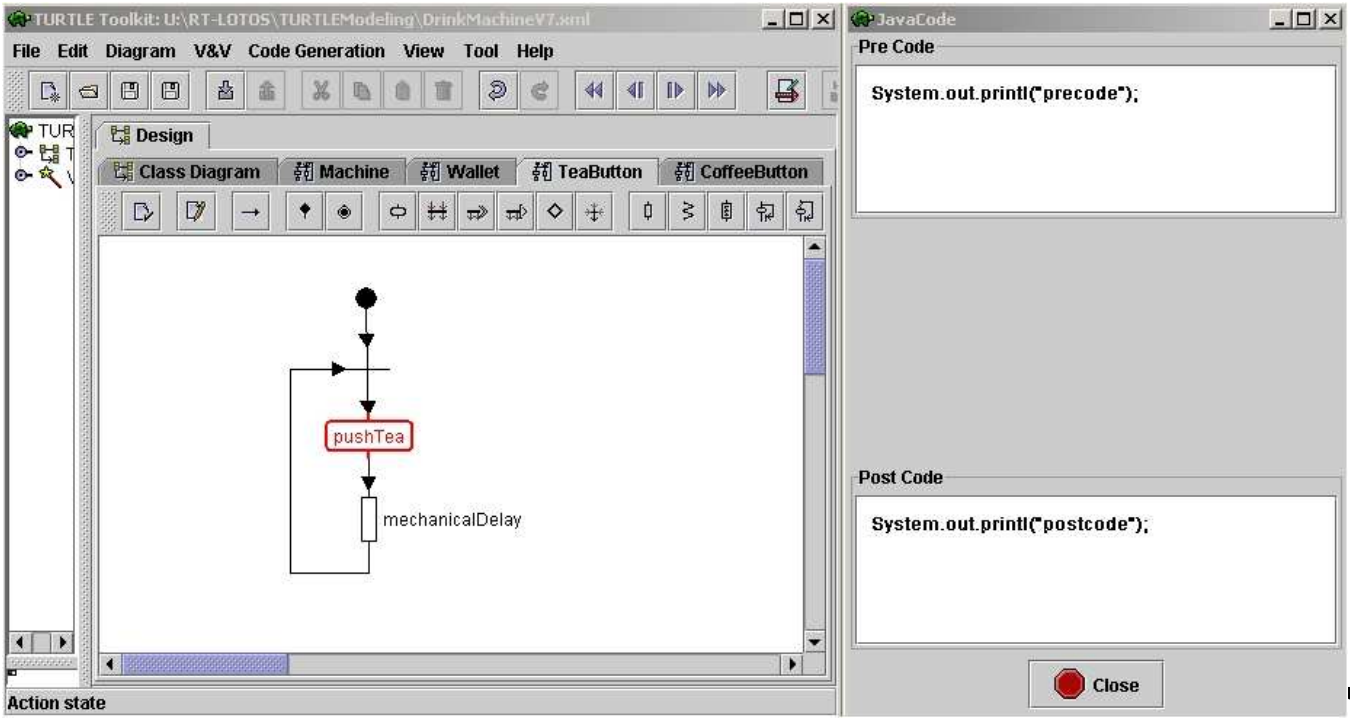
TELECOM PARIS
école nationale supérieure des télécommunications

- **config.xml**
  - *<JavaCodeDirectory data="U:\RT-LOTOS\JavaCode\" />*
    - **Directory in which Java code is generated**
  - *<JavaCompilerPath data="[C:\Program Files\j2sdk_nb\j2sdk1.4.2\bin\javac.exe" />*
    - **Compiler used to compile generated Java code**
  - *<TToolClassPath data="U:\RT-LOTOS\JavaCode" />*
    - **Classpath used at compilation / execution**
  - *<JavaExecutePath data="[C:\Program Files\j2sdk_nb\j2sdk1.4.2\bin\java.exe" />*
    - **Link to the Java virtual machine used for executing generated Java code**
- **Libraries**
  - ***Used by the generated Java code***
  - ***Should be located in the TToolClassPath***

Adding directives

TURTLE design

- *Code may be added on the following operators of activity diagrams*
  - **Actions**
  - **Variable settings**
- *Java code executed before the action*
- *Java code executed after the action*

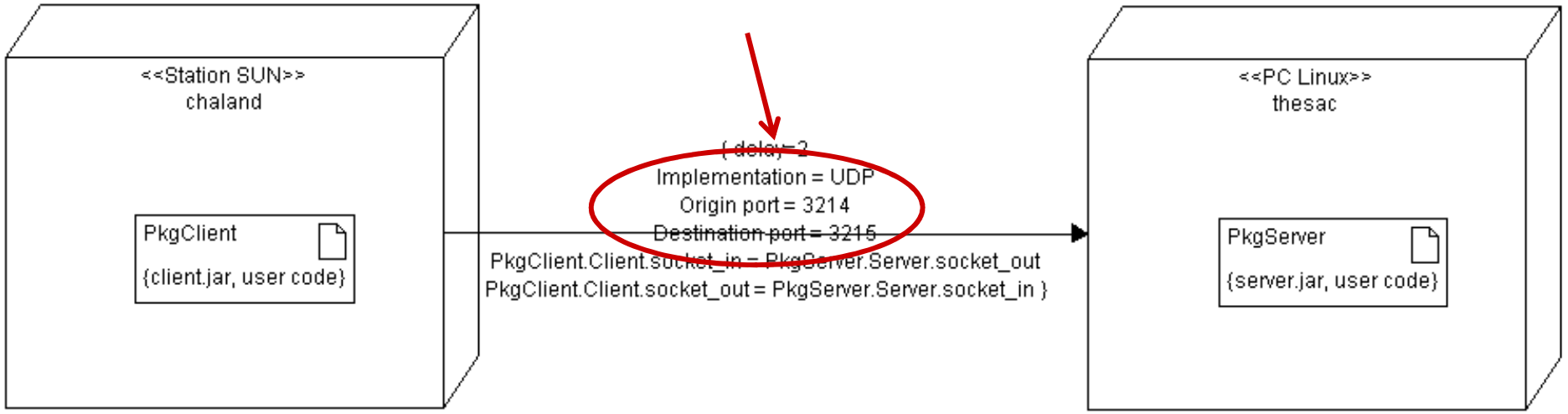


Adding directives (Cont.)

TURTLE deployment

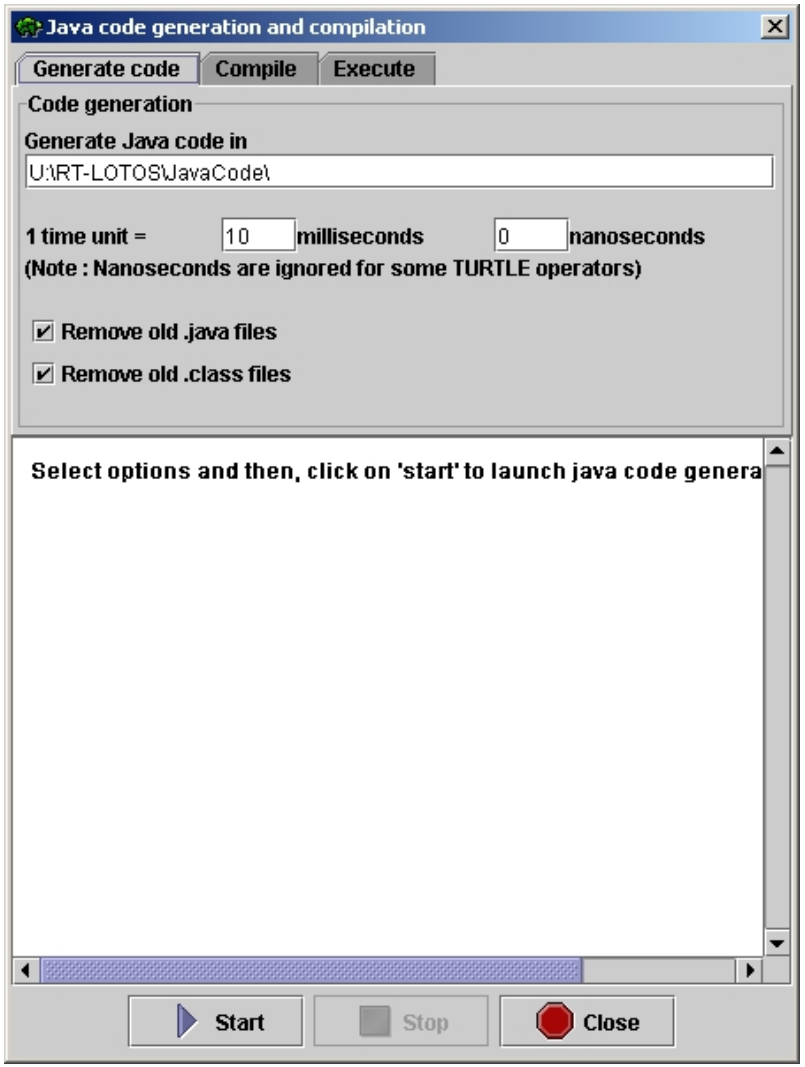
- *Code may be added to TURTLE components (considered as TURTLE designs)*
- *Communication protocols between TURTLE components may be specified on links between TURTLE nodes*
  - **UDP, TCP, RMI**

Info. on protocols

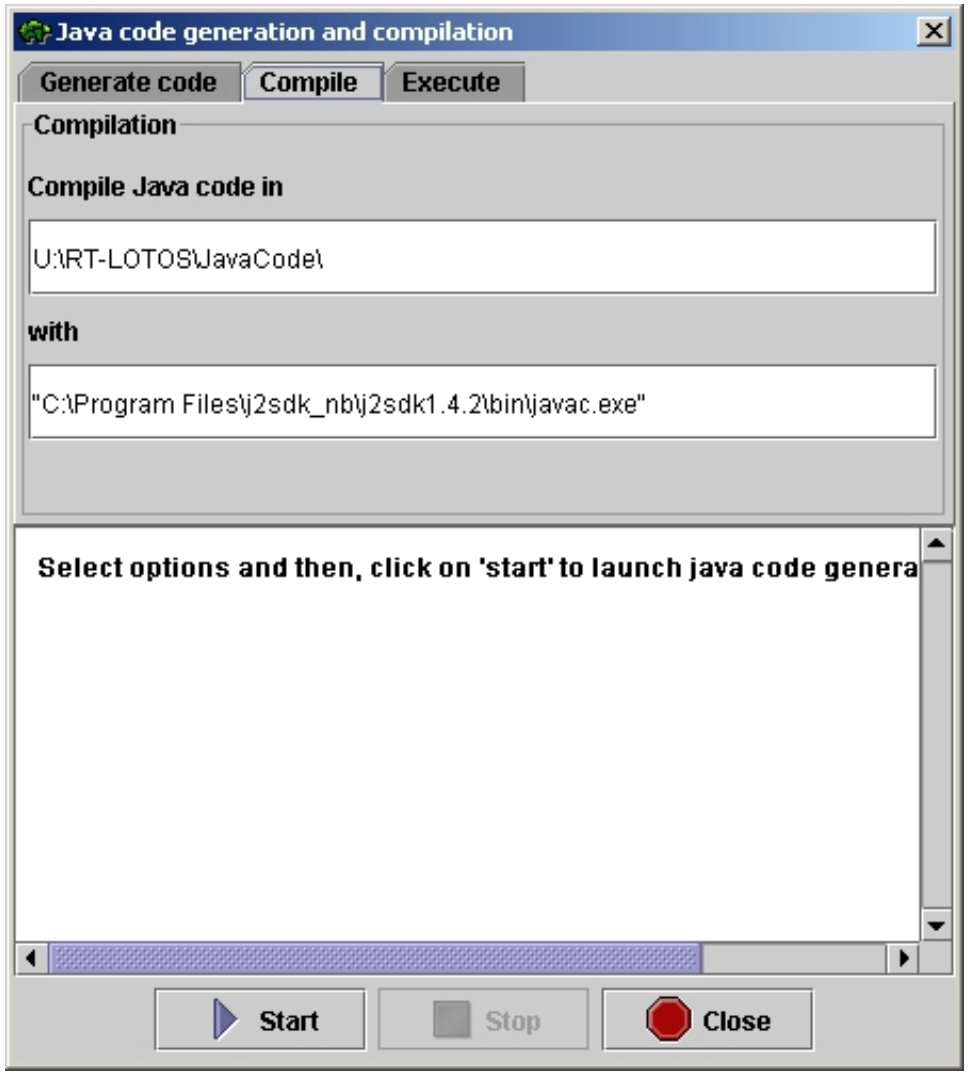


Generating Java Code

- Check the syntax of your modeling
- Then, select the *Generate JAVA* icon
- Set various options, including
  - *TURTLE time unit vs. Java time unit*
  - *Debug information*

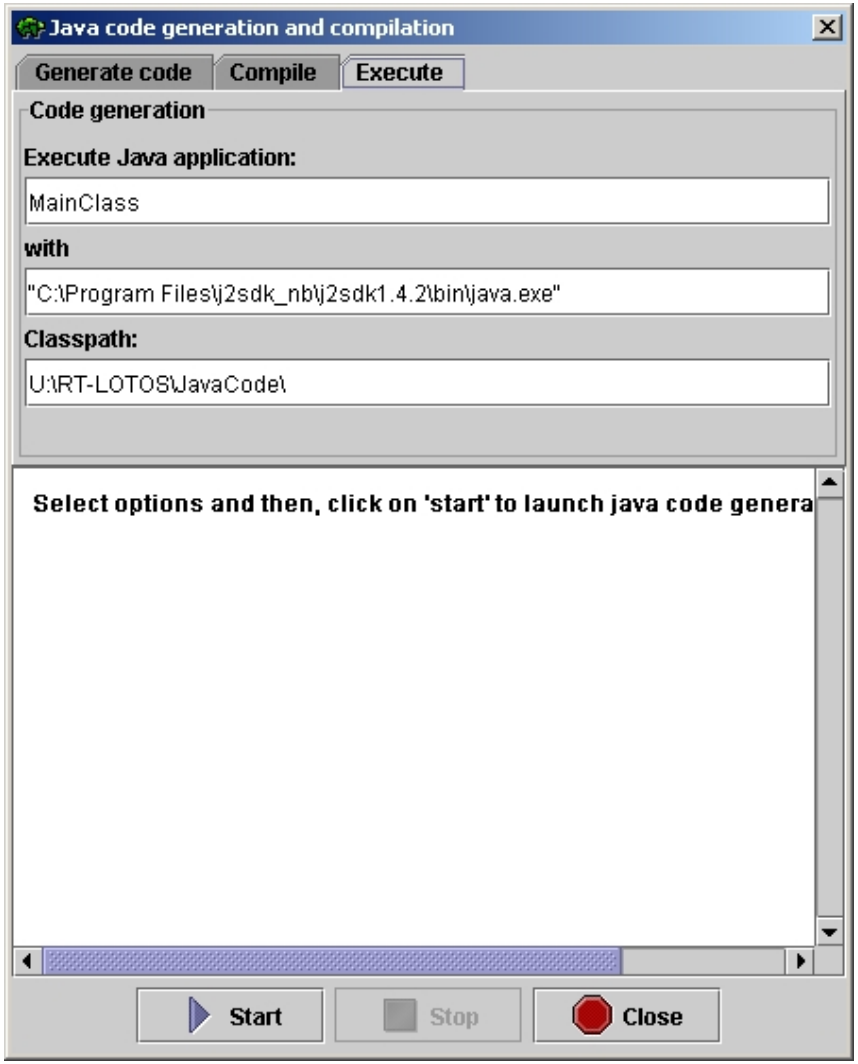


Compiling Java code



Executing Generated Java Applications

- **Monolithic applications**
  - *Generated from TURTLE designs*
  - *Use of the provided graphical interface*
- **Distributed systems**
  - *Command line*
  - *Various applications should be started in the right order*
    - Server side before client side
  - *RMI*
    - Start *rmiregistry* first
    - Set the security policy
    - Run various instances
    - See the online help for more information



How to Improve Generated Code?

TELECOM PARIS
école nationale supérieure des télécommunications

- **Do not use non deterministic choices**
  - ***They are not deterministic***
  - ***The Java code generator may not interpret them as you expected***
- **Creation of parallel subactivities, sequence operators, preemption operators are translated using Java threads for each subactivies**
  - ***Performance drawback***

Examples / Exercises

TELECOM PARIS
école nationale supérieure des télécommunications

- **http://www.eurecom.fr/~apvrille/TURTLE/HELP/index.html**
  - ***"Example" section***

- **Technical support, maintenance, bug report**
  - ***ludovic.apvrille@enst.fr***