Model-Based Testing

Test Selection and

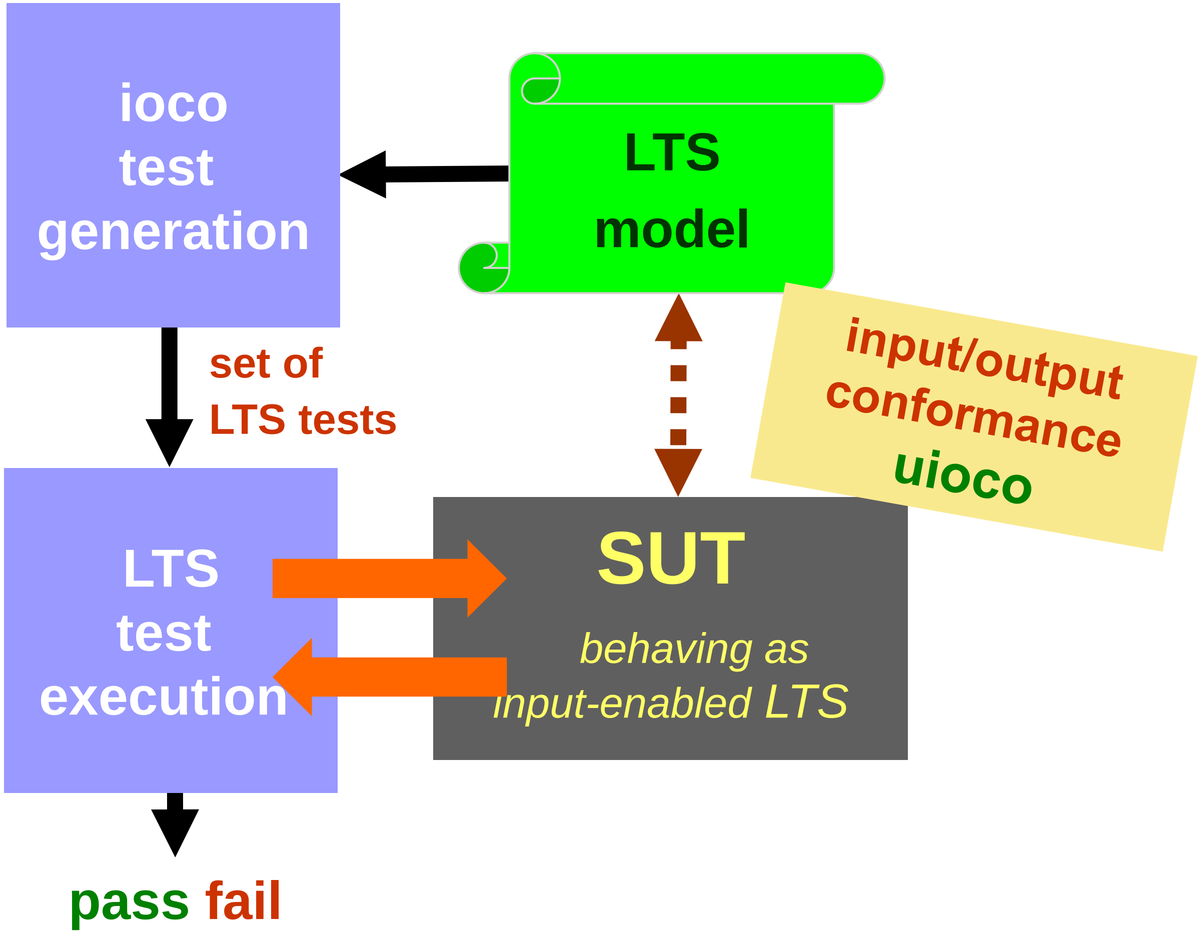
Test Purposes

MBT : LTS & ioco

SUT ioco model

sound ⇓ ⇑ *exhaustive*

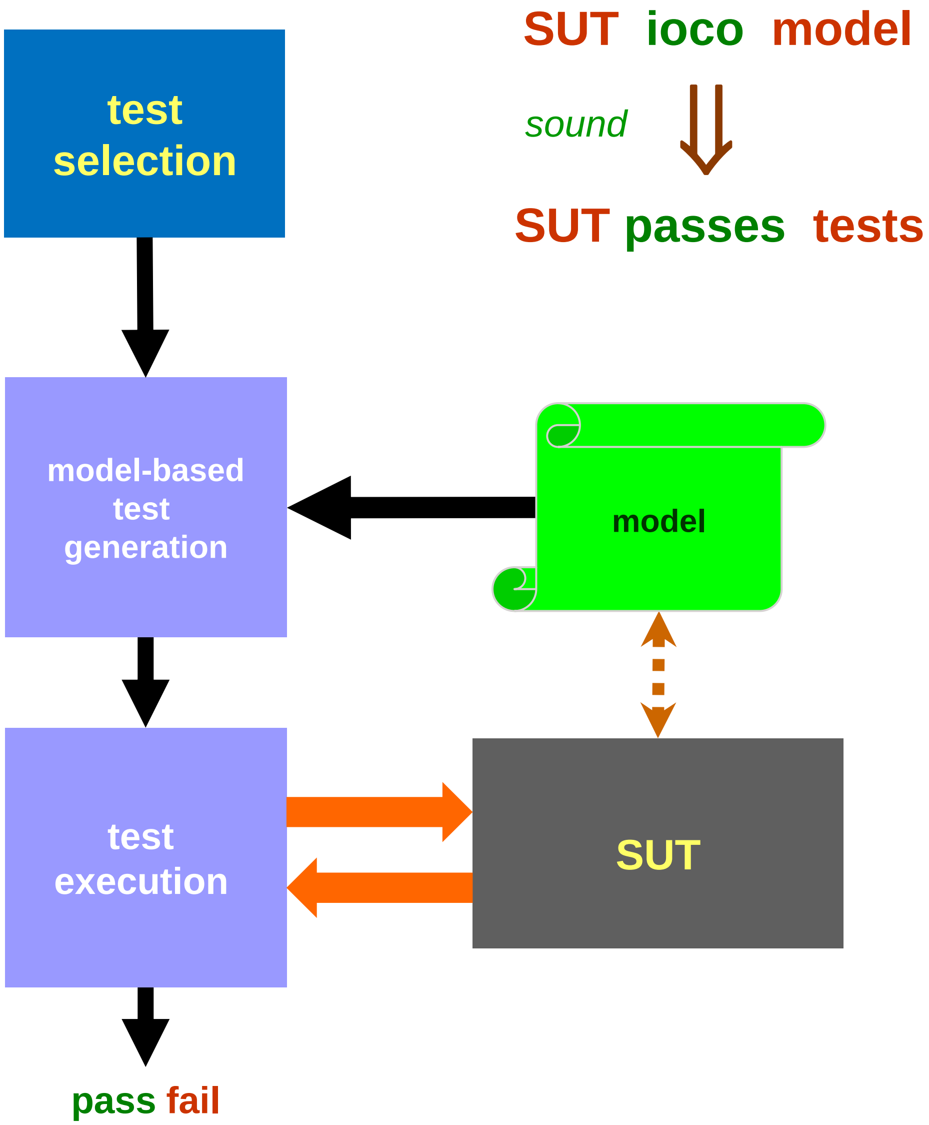
SUT passes tests



Test Selection Problem

Many sound test cases can be generated:

- No time to execute them all
- Which ones are the best?
- How many?



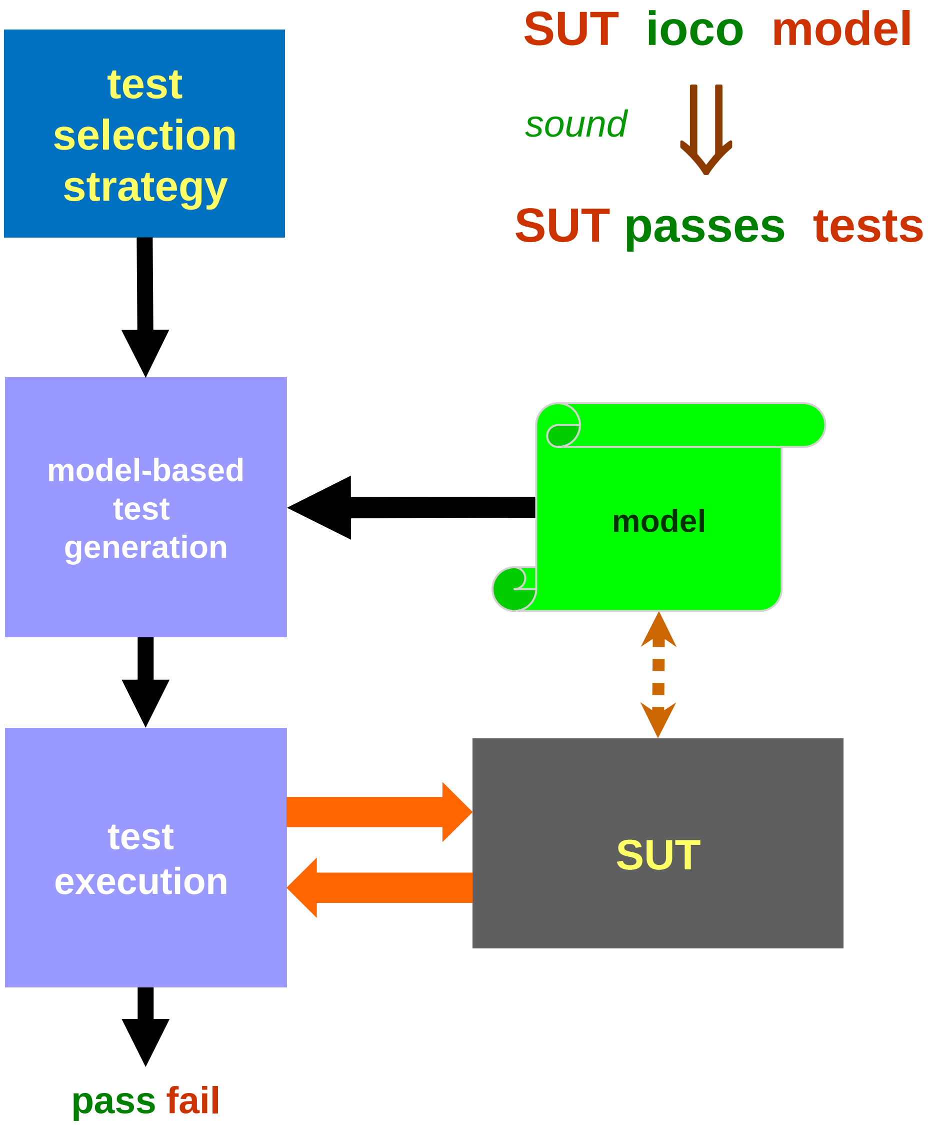
Test Selection Strategy

Strategy to select test cases:

- bottom line: **random**
- by tester: **test purposes**
- structure of model: **coverage**

Extra (domain) information required:

- which test cases have high value?
- which errors are likely?
- which errors have high impact?
- what is the user / customer doing?

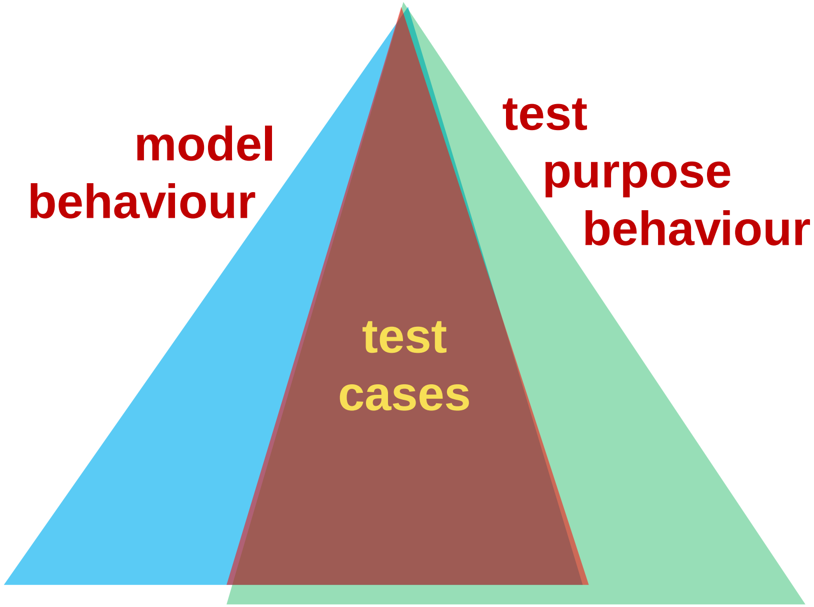


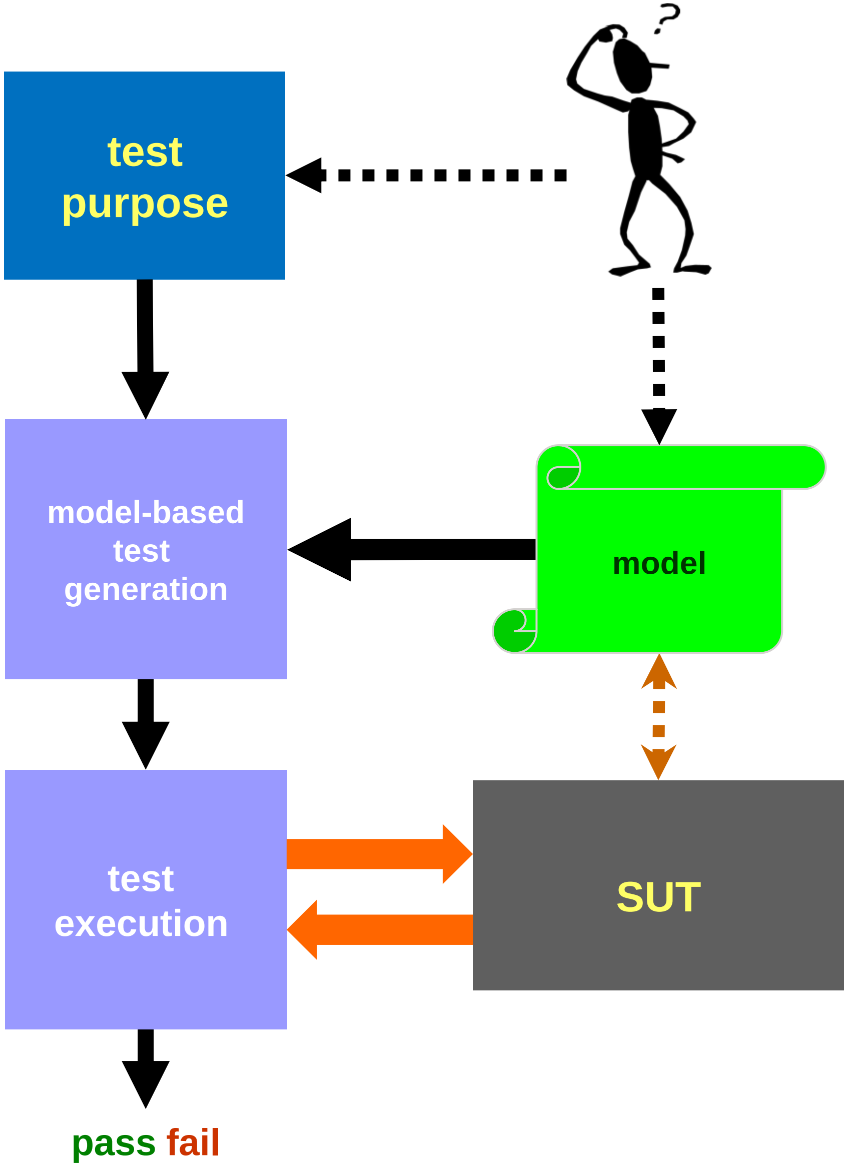
Test Purpose

Tester defines explicitly

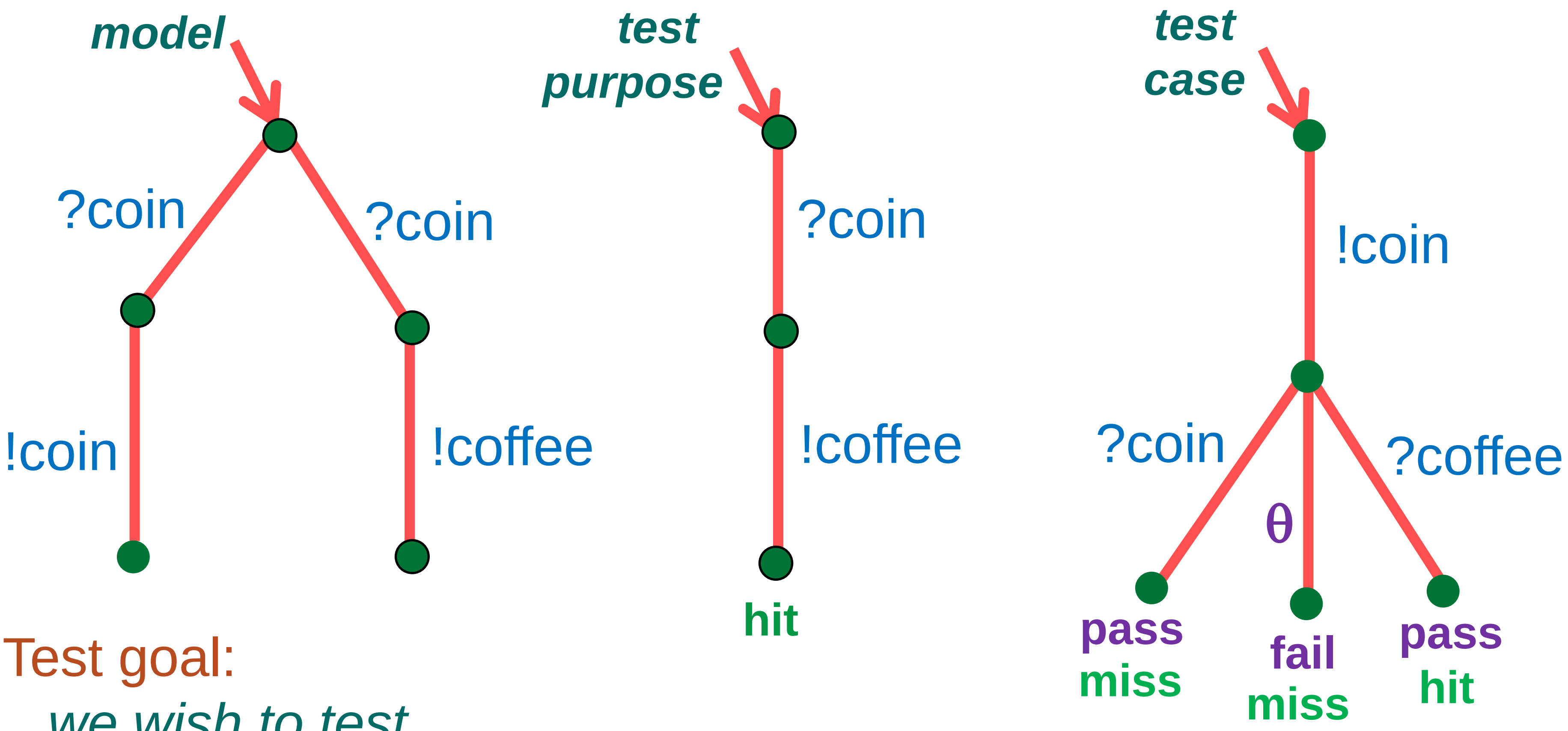
the purpose of testing

- Model: allowed behaviour
- Test Purpose: targeted behaviour





Test Purpose : Example



Test goal:

we wish to test whether the machine can deliver !coffee *after* ?coin

Desired observation:

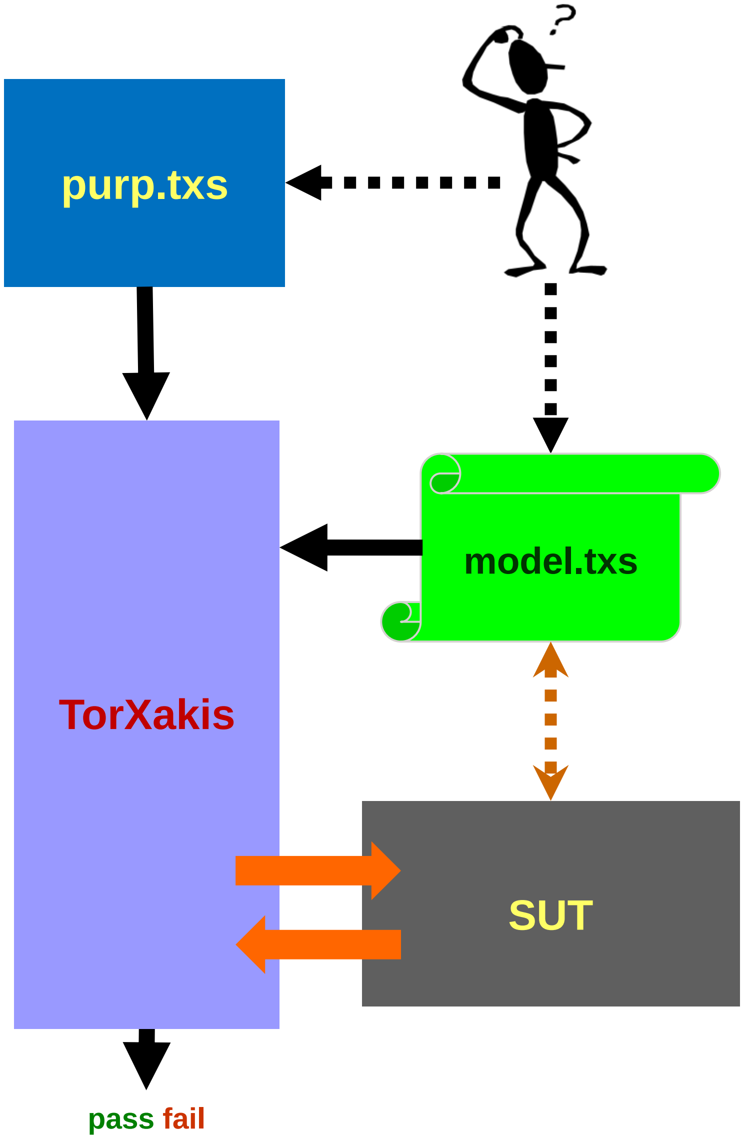
?coin . !coffee

TorXakis

Test Purposes

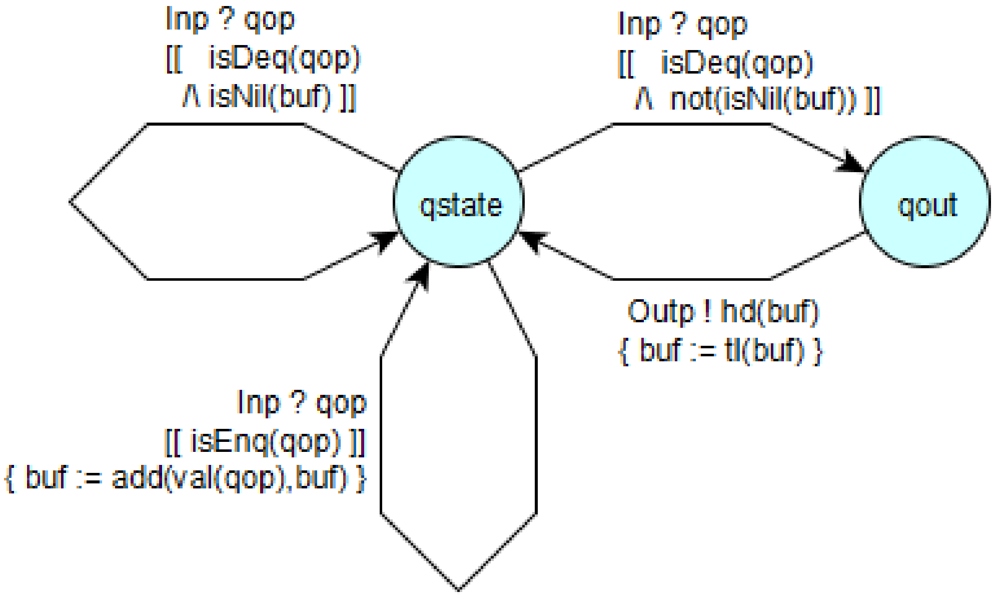
TorXakis
Test Purpose

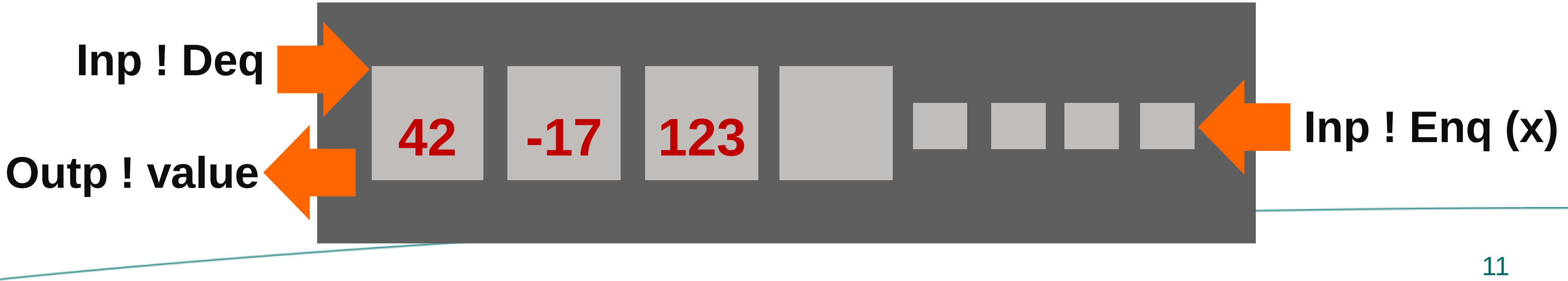
- Steering test generation via PURPDEF
- TorXakis selects random actions consistent with MODEL *and* with TEST PURPOSE
- Stops if *either* MODEL *or* TEST PURPOSE reaches end
- TEST PURPOSE = *LTS*
- TEST PURPOSE can be previous trace → *replay*



TorXakis: Queue

```
STAUTDEF  queueStaut  [ Inp :: QueueOp;  Outp :: Int ]  ( )
 ::=
   STATE  qstate, qout
   VAR  buf :: IntList
   INIT  qstate  { buf := Nil }
```



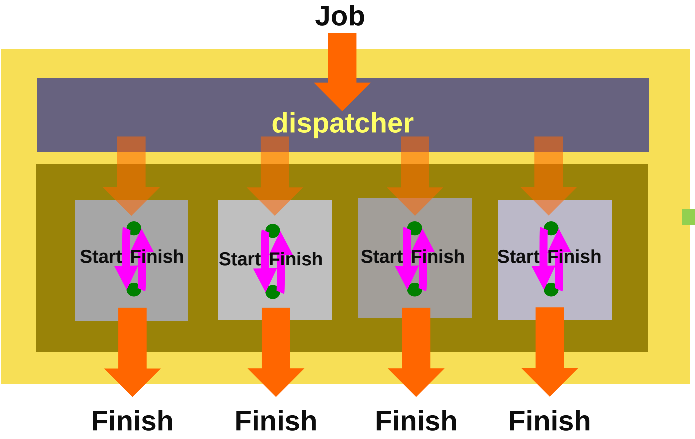


TorXakis

Test Purposes

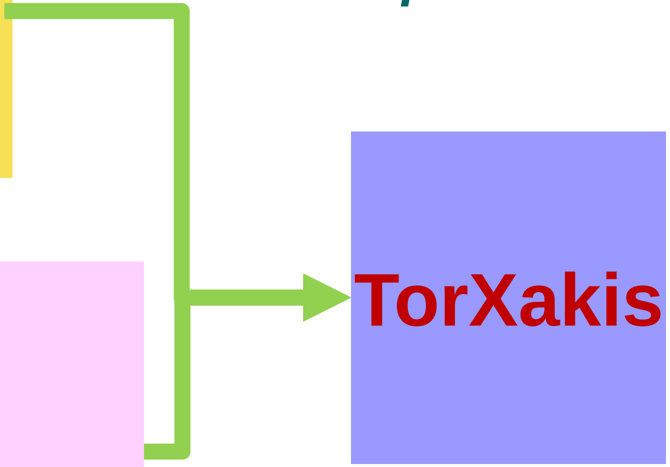
Example: *Dispatcher System*

Example: Dispatcher-Processing System



Test Purpose:

test with inputs such that (g)cd = 1 … 10



```
PURPDEF Purp
 ::=
    CHAN IN      Job
    CHAN OUT  Finish

    GOAL  gcdgoal  ::=  gcd10 [Job, Finish] (1)
ENDDEF
```

```
PROCDEF  gcd10 [Job :: JobData; Finish :: JobOut ] (d :: Int) HIT
 ::=
       [[ d <= 10 ]] =>>      Job ?job [[ (x(job)%d) == 0,  (y(job)%d) == 0 ]]
                              >-> Finish ?r
                              >-> gcd10 [Job, Finish] (d+1)
     ##
       [[ d > 10 ]]   =>>  HIT
ENDDEF
```