Model-Based Testing

Example :

Testing *Dropbox* with *TorXakis*

Model-Based Testing of Dropbox

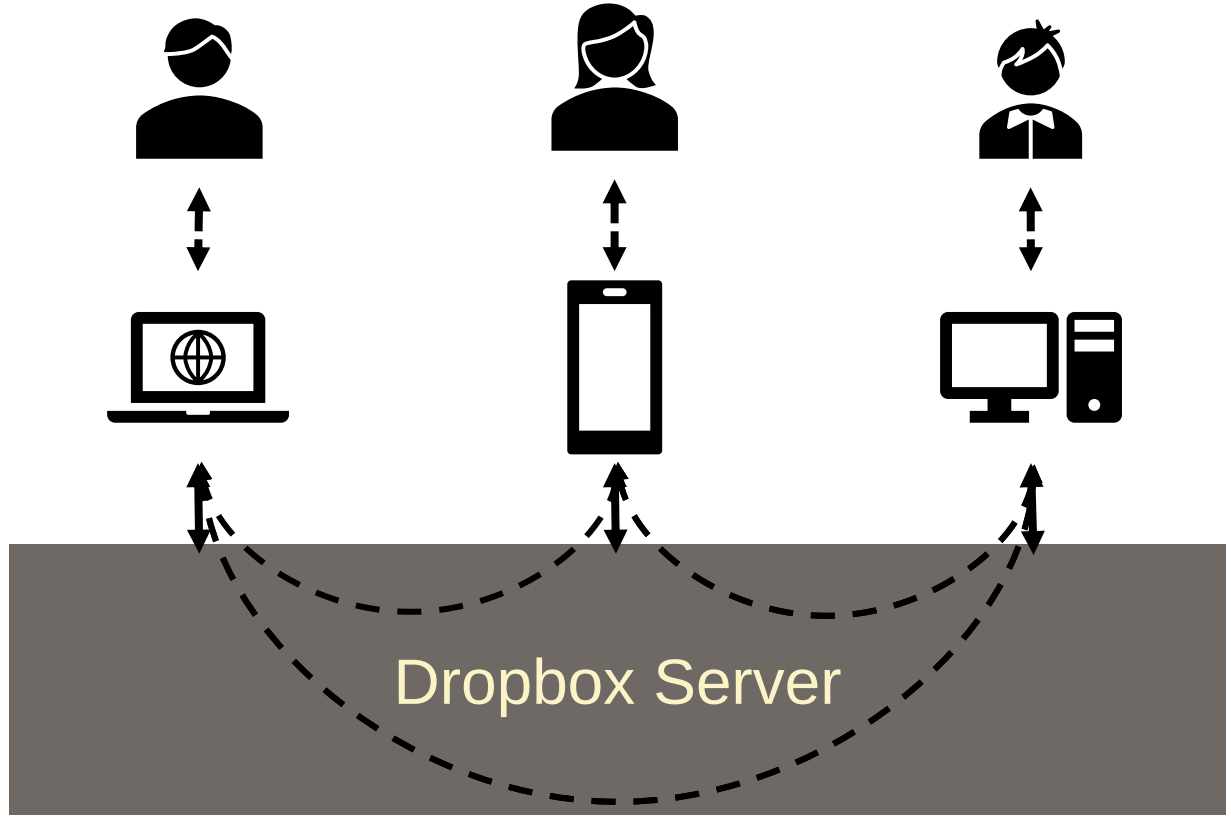
- Inspired by Dropbox testing with *Quviq Quickcheck*:

  J. Hughes, B. Pierce, T. Arts, U. Norell,
  Mysteries of DropBox: Property-Based Testing of a Distributed Synchronization Service.
  In IEEE ICST, pp. 135–145, 2016.

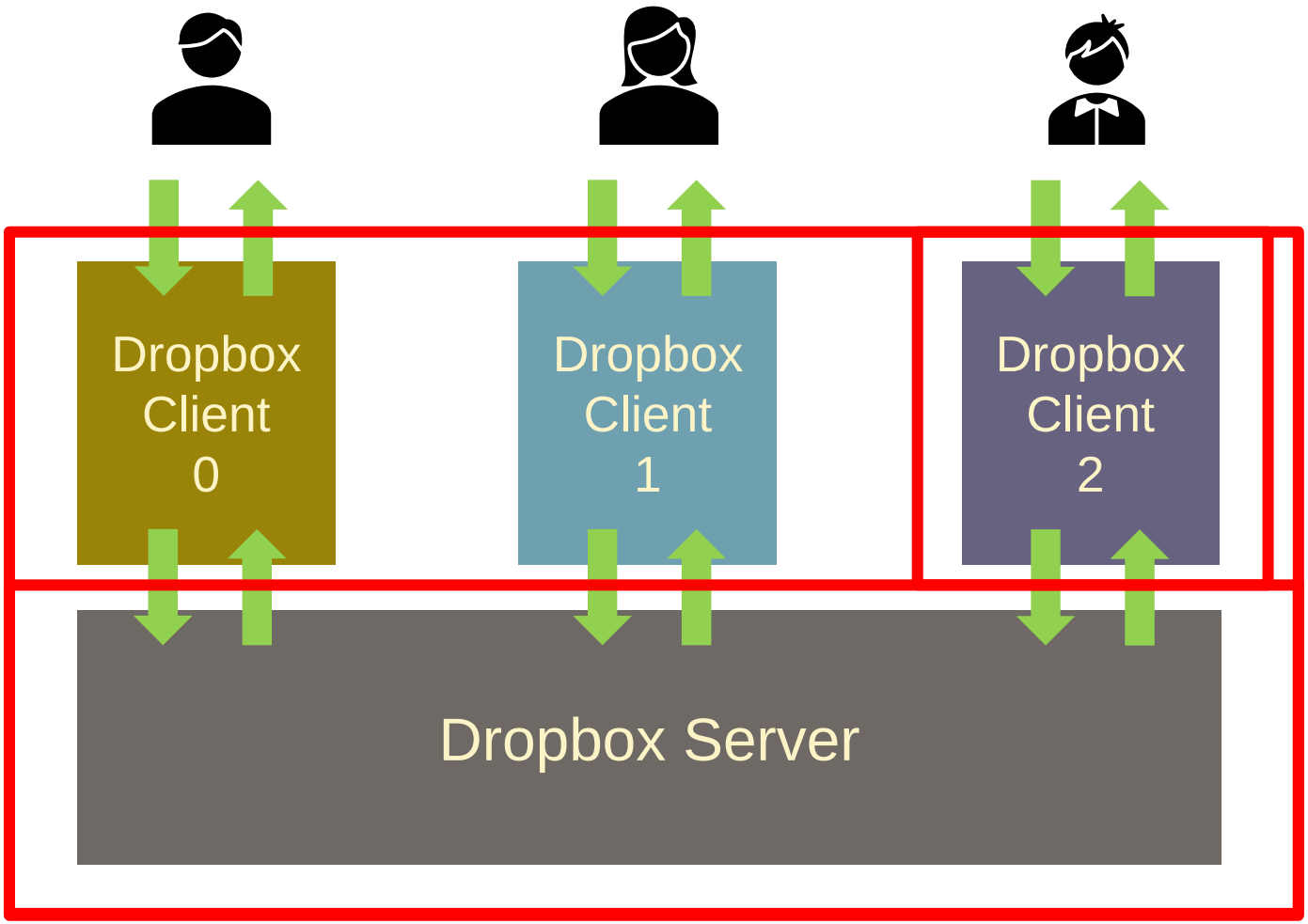
1. Dropbox
2. Testing approach
3. Model
4. Test runs

Jan Tretmans, Piërre van de Laar,
Model-Based Testing with TorXakis – The Mysteries of Dropbox Revisited.
In CECIIS, pp. 247–258, 2019.

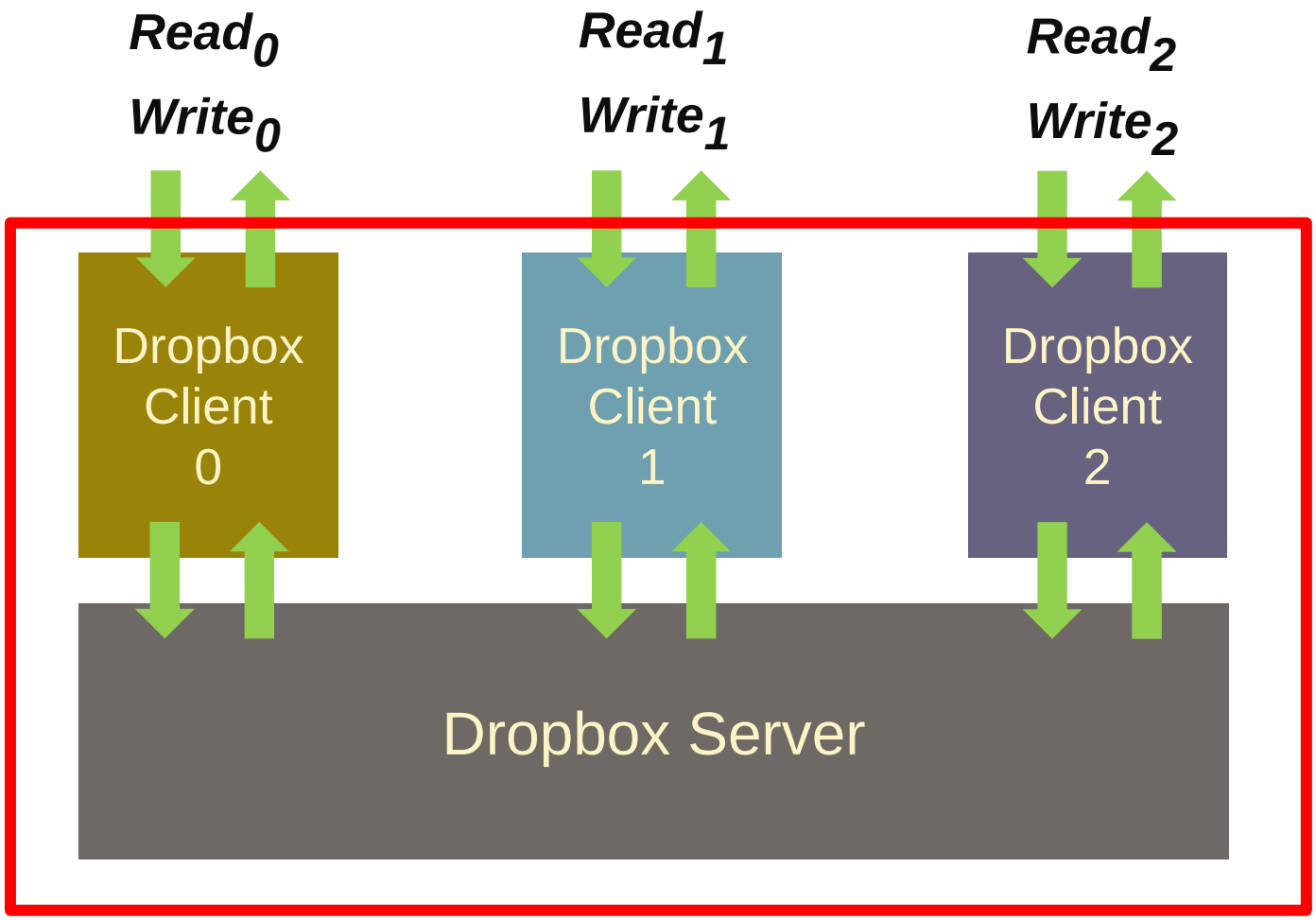
Dropbox : A File Synchronization Service



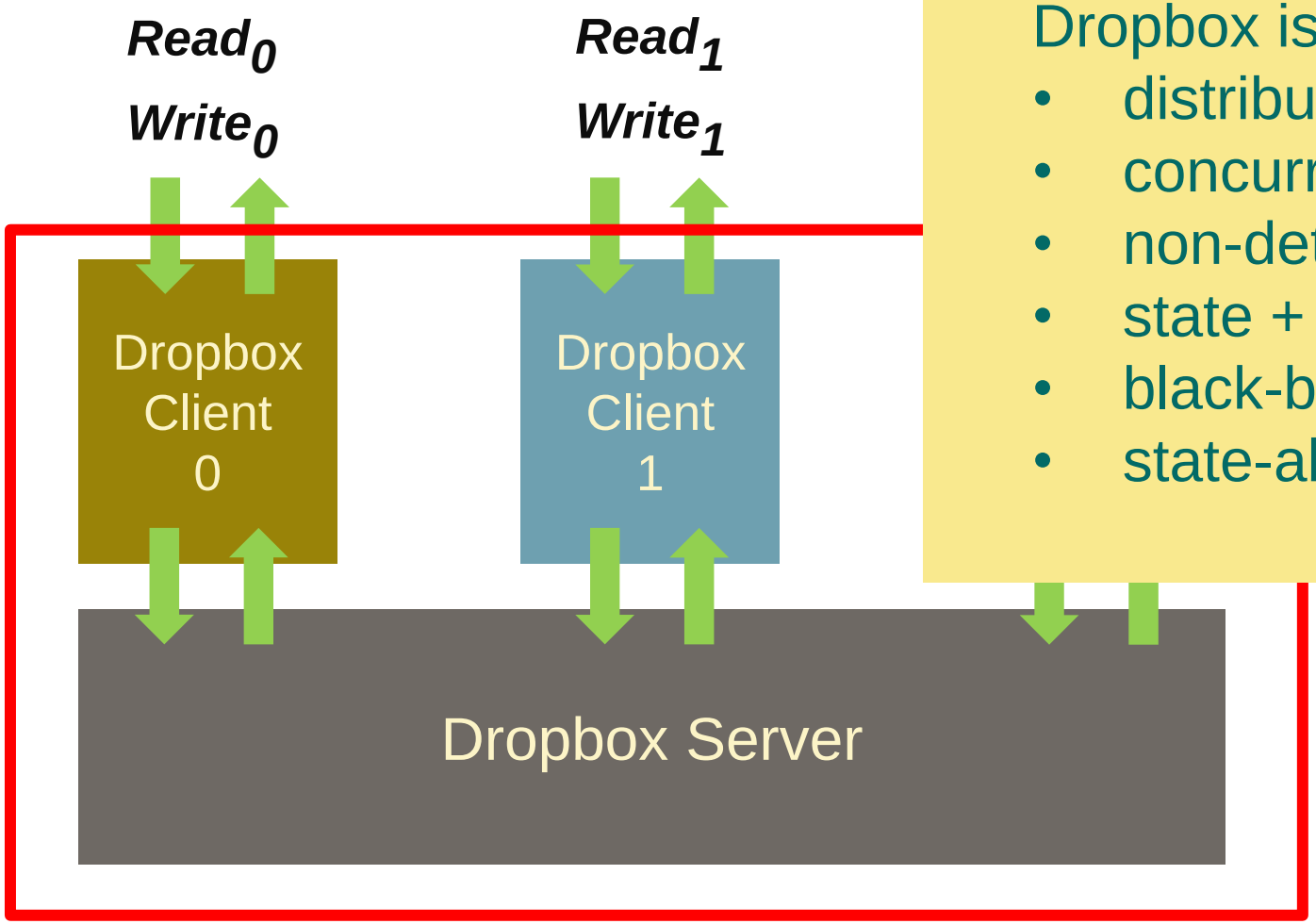
Testing Dropbox



Testing Dropbox



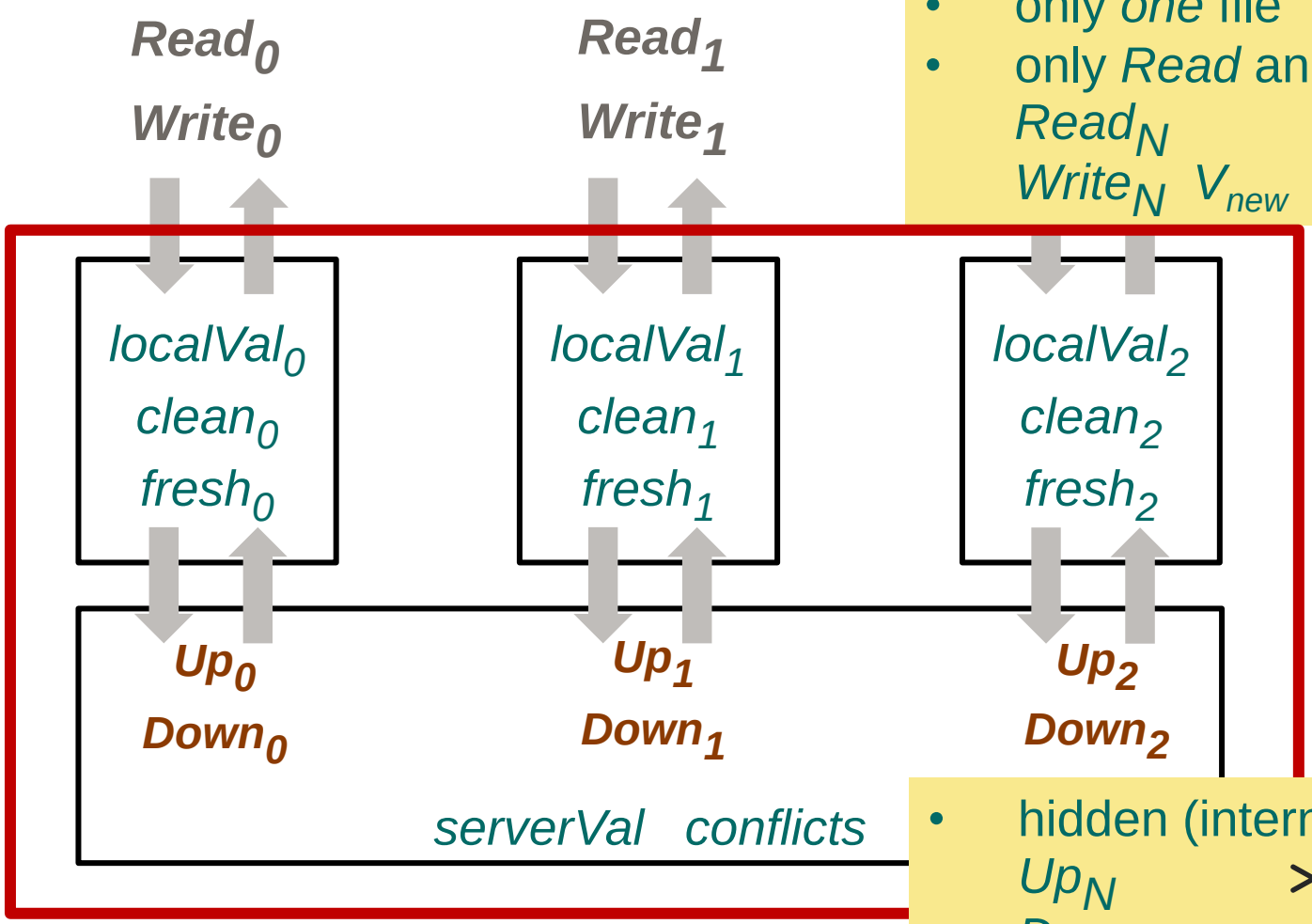
Testing Dropbox



Dropbox is

- distributed
- concurrent
- non-deterministic
- state + data
- black-box
- state-abstracted

A Dropbox Model (*Hughes et al.*)



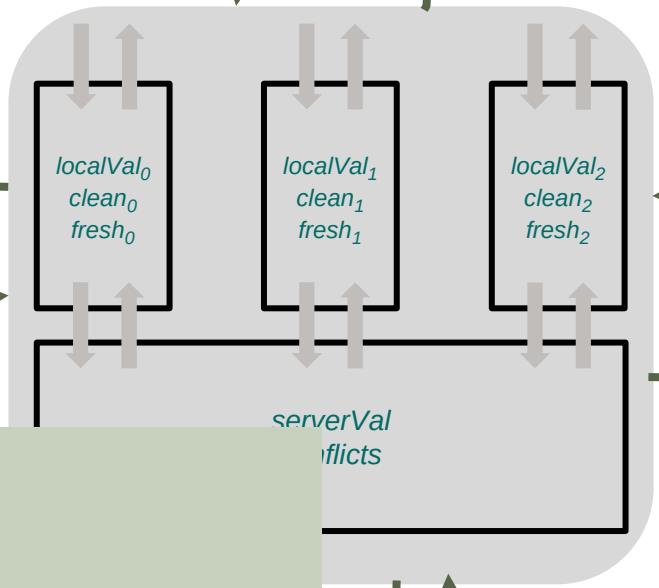
- only *one* file
- only *Read* and *Write* :
  $Read_N$ >-> *file value*
  $Write_N\ V_{new}$ >-> *old file value*

- hidden (internal) actions
  Up_N >-> *upload file*
  $Down_N$ >-> *download file*

A Dropbox Model

$In\,?\,Read_N$
$Out\,!\,localVal_N$

$In\,?\,Write_N\ V_{new}$
$Out\,!\,localVal_N$
$localVal_N ::= V_{new}$
$clean_N ::= False$



$\neg\, clean_N \rightarrow$
Up_N
$clean ::= True$
$if\ fresh_N\ then$
$if\ localVal_N \neq serverVal\ then$
$fresh_{N'} ::= False\ for\ all\ N' \neq N$
$serverVal ::= localVal_N$
$else\ if\ localVal_N \notin \{serverVal, \perp\}\ then$
$conflicts ::= conflicts \cup \{localVal_N\}$

$\neg\, fresh_N \wedge clean_N \rightarrow$
$Down_N$
$localVal_N ::= serverVal$
$fresh_N ::= True$

A Dropbox Model

$In\,?\,Read_N$
$Out\,!\,localVal_N$

$In\,?\,Write_N\ V_{new}$
$Out\,!\,localVal_N$
$localVal_N ::= V_{new}$
$clean_N ::= False$

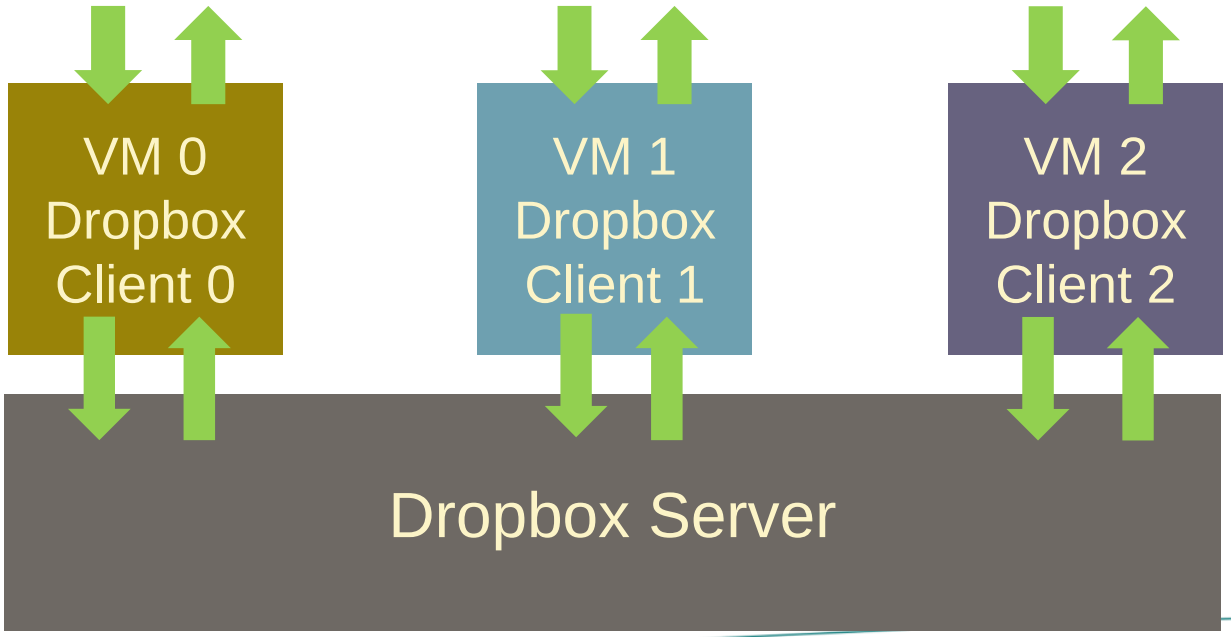
```
[[ not(lookup(fresh,Node(N))) /\ lookup(clean,Node(N)) ]]
=>>  Down
>->  dropBox [ In,Out,Down,Up ]
             ( serverVal
             , conflicts
             , update(localVal,Node(N),serverVal)
             , update(fresh,Node(N),True)
             , clean
             )
```

$\neg\, clean_N$
Up_N
$clean$
$if\ fresh_N\ then$
$if\ localVal_N \neq ser$
$fresh_N, ::= Fal$
$serverVal ::=$
$else\ if\ localVal_N \notin \{$
$conflicts ::=$

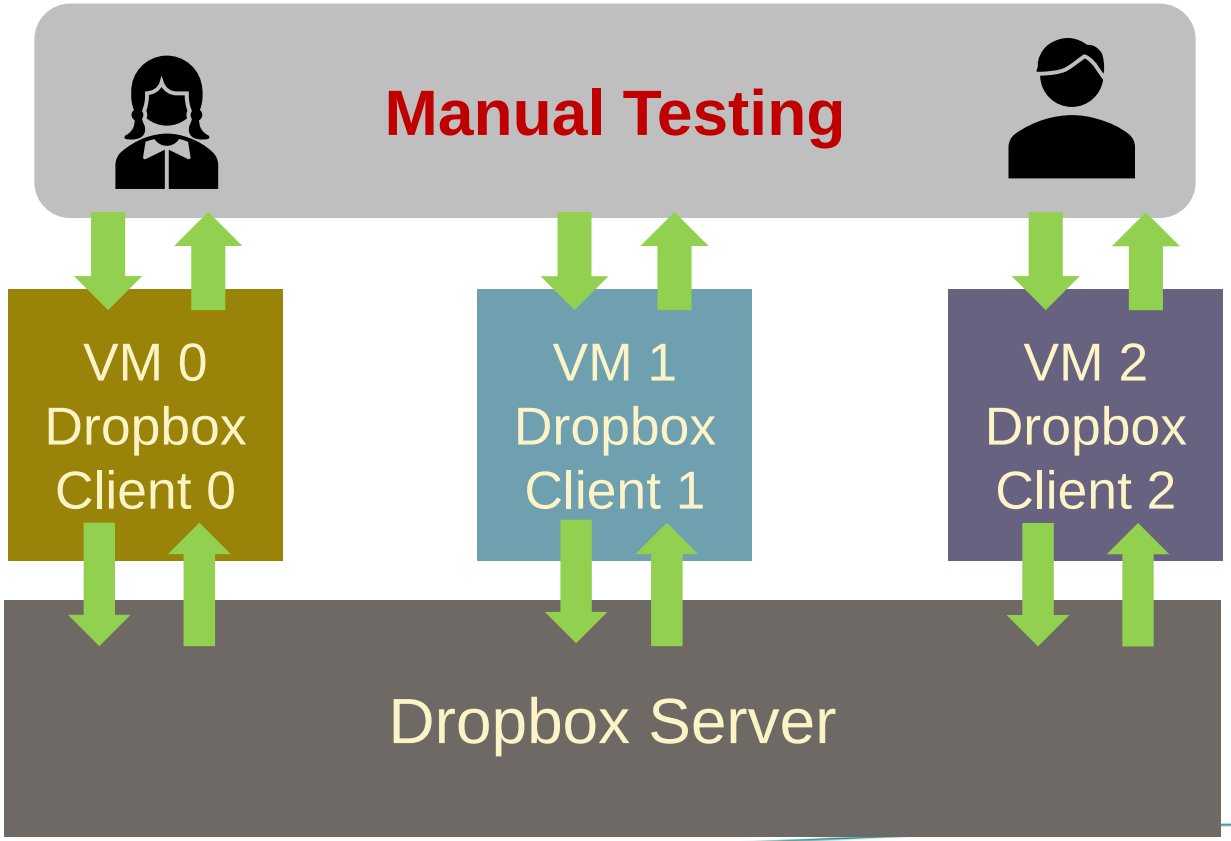
$\neg\, fresh_N \wedge clean_N \rightarrow$
$::= serverVal$
$= True$

HIDE [Up,Down] IN dropbox NI

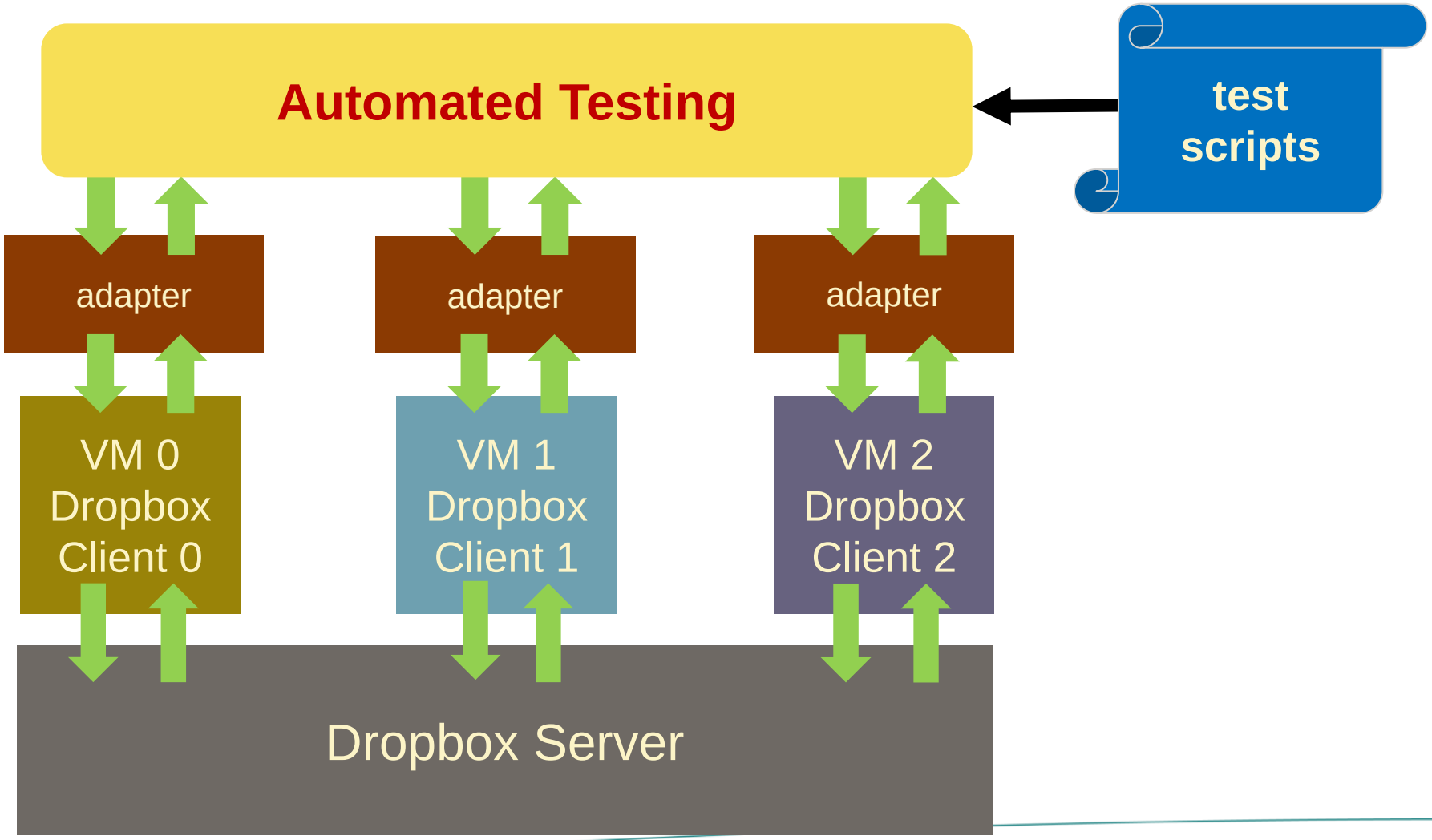
Testing Dropbox



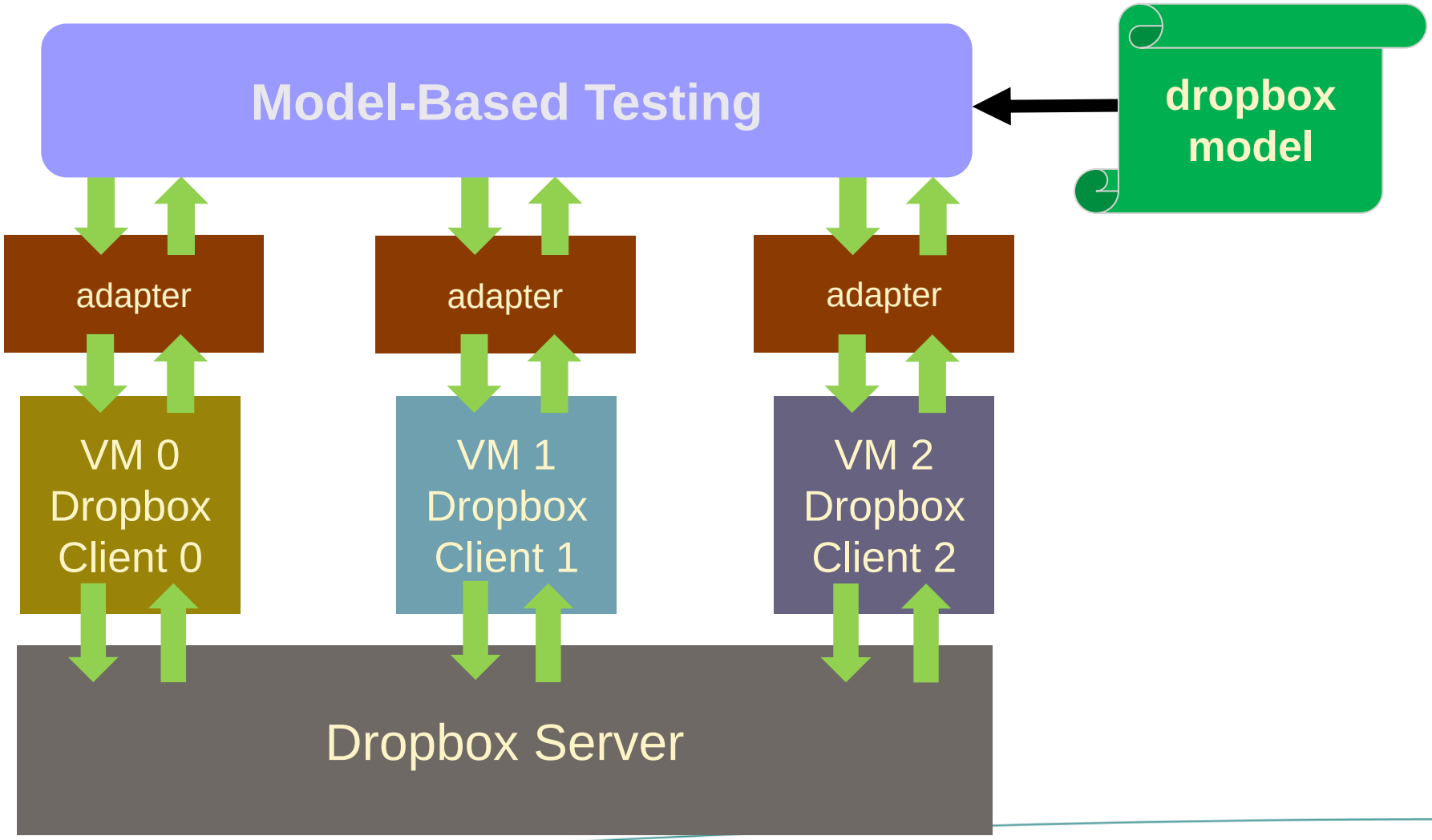
Testing Dropbox



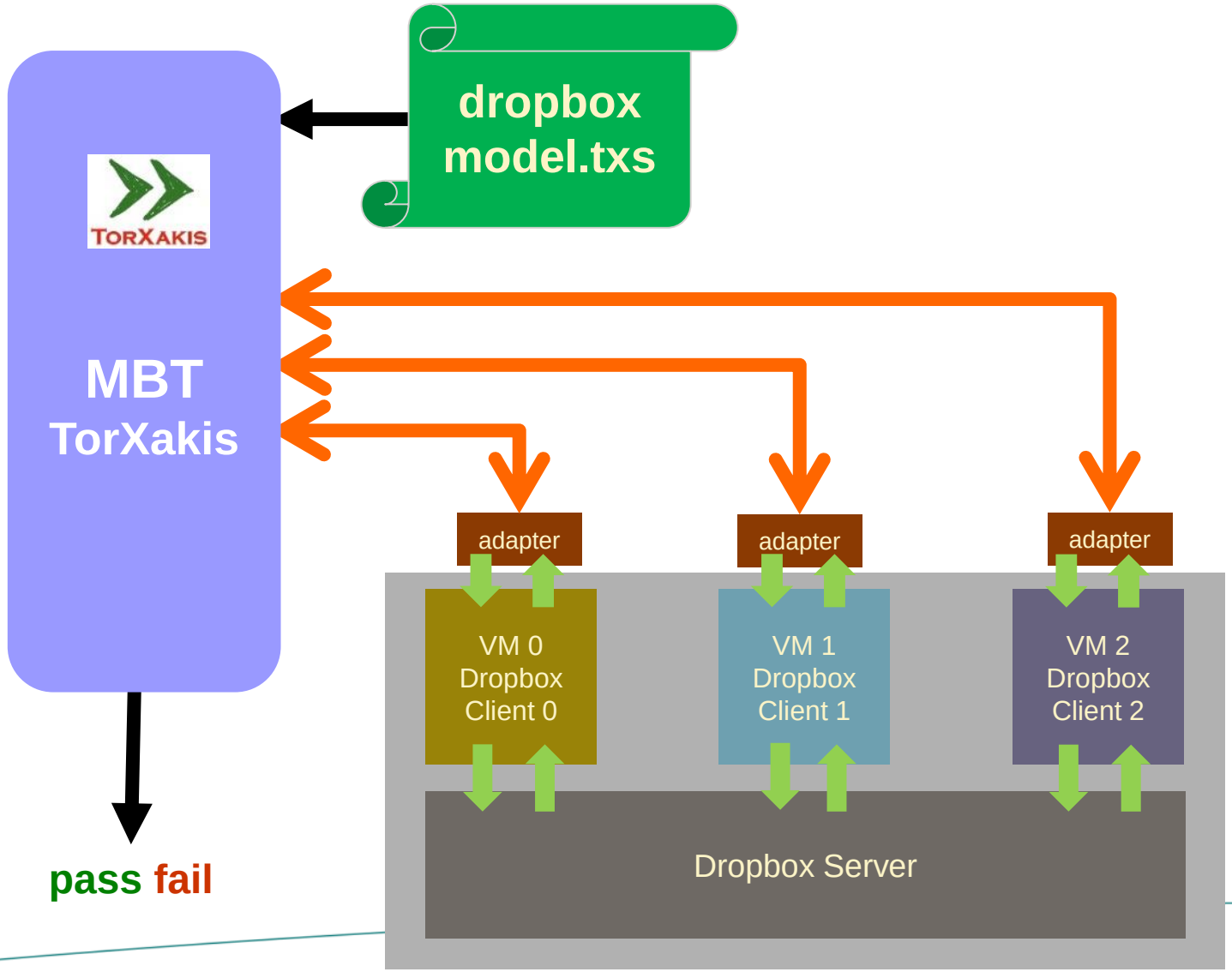
Testing Dropbox



Testing Dropbox



TorXakis : Dropbox



Dropbox Test Run

```
TXS >> ...5: IN:   In1  ! Write(Value("P"))
TXS >> ...6: OUT:  Out1 ! File(Value("$"))
TXS >> ...7: IN:   In0  ! Write(Value("SHK"))
TXS >> ...8: OUT:  Out0 ! File(Value("$"))
TXS >> ...9: IN:   In1  ! Read
TXS >> ..10: OUT:  Out1 ! File(Value("P"))
```

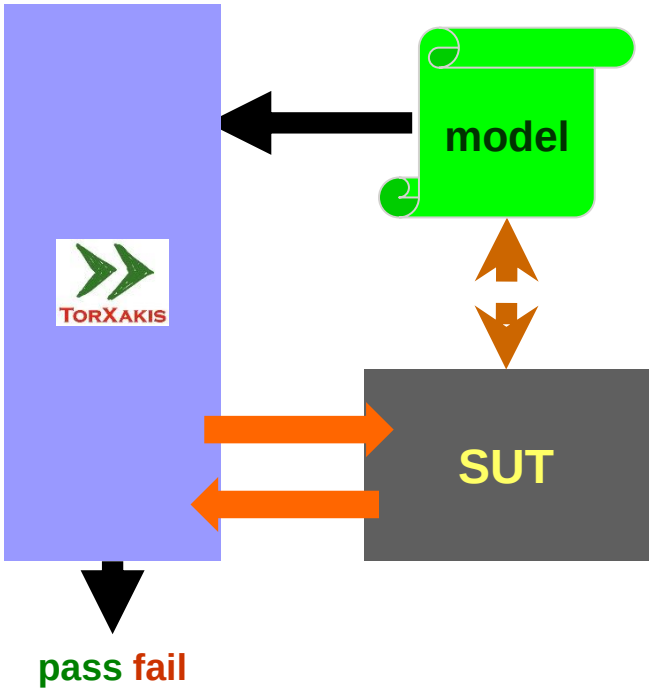
Dropbox Test Run

```
TXS >> ..11: IN:   In0  ! Write(Value("X"))
TXS >> ..12: OUT:  Out0 ! File(Value("SHK"))
TXS >> ..13: IN:   In2  ! Write(Value("A"))
TXS >> ..14: OUT:  Out2 ! File(Value("$"))
TXS >> ..15: IN:   In2  ! Write(Value("SP"))
TXS >> ..16: OUT:  Out2 ! File(Value("A"))
TXS >> ..17: IN:   In1  ! Write(Value("BH"))
TXS >> ..18: OUT:  Out1 ! File(Value("P"))
TXS >> ..19: IN:   In2  ! Read
TXS >> ..20: OUT:  Out2 ! File(Value("SP"))
TXS >> ..21: IN:   In0  ! Read
TXS >> ..22: OUT:  Out0 ! File(Value("X"))
TXS >> ..23: IN:   In2  ! Write(Value("PXH"))
TXS >> ..24: OUT:  Out2 ! File(Value("X"))
```

Dropbox Test Run

```
TXS >>  ..77: IN:   In0  ! Stabilize
TXS >>  ..78: OUT:  Out0 ! File(Value("P"))
TXS >>  ..79: OUT:  Out0 ! File(Value("L"))
TXS >>  ..80: OUT:  Out0 ! File(Value("TK"))
TXS >>  ..81: OUT:  Out0 ! File(Value("P"))
TXS >>  Expected:
( { Out0[ $"Out0"$1266 ] }
, ( IF isFile($"Out0"$1266)
    THEN isValueInList(["PH","H"],value($"Out0"$1266))
    ELSE False
    FI
  )
)
TXS >>  FAIL:  Out0 ! File(Value("P"))
```

TorXakis



The next step in

Model-Based Testing