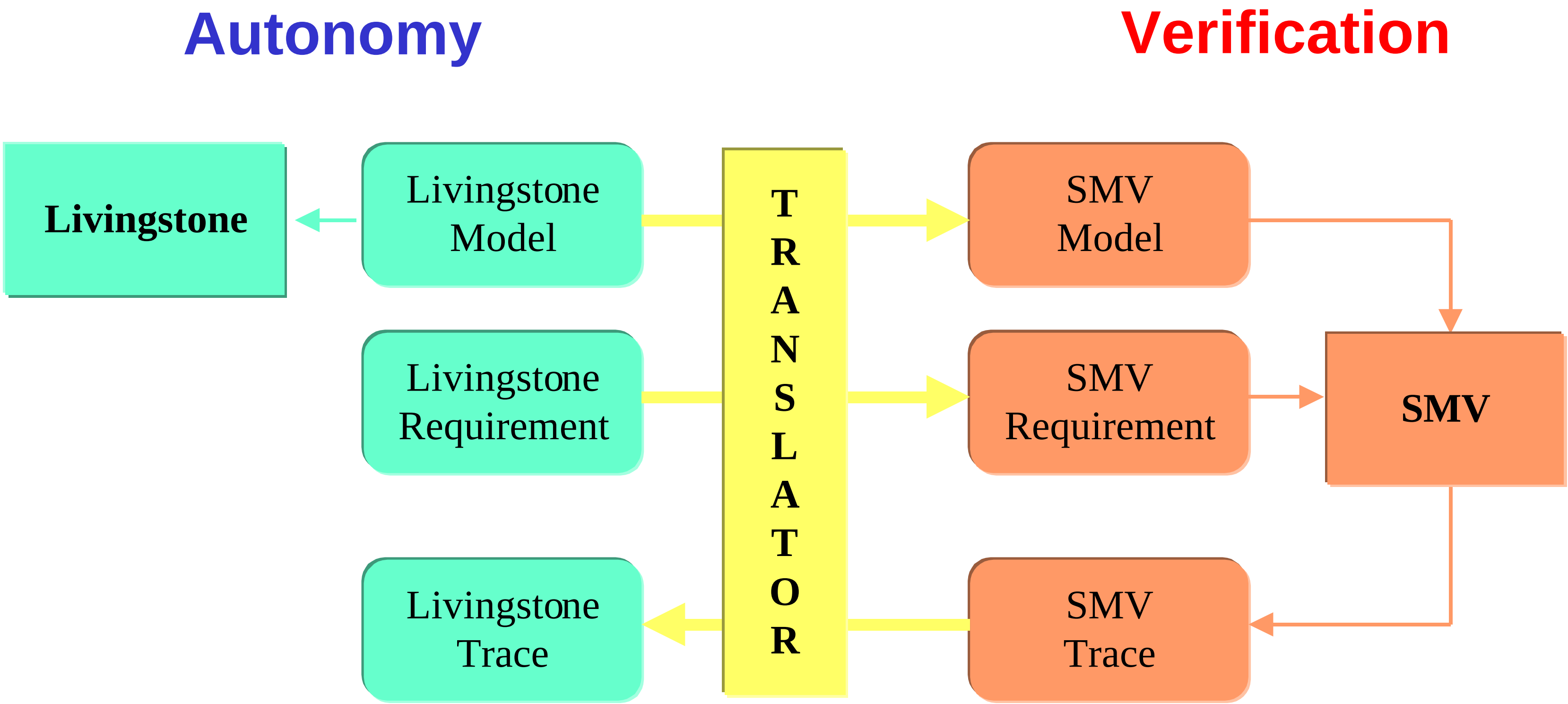
From Livingstone to SMV

Formal Verification for Autonomous Spacecrafts

Charles Pecheur (RIACS / NASA Ames)

Reid Simmons (Carnegie Mellon University)

Overview


Autonomy
Verification
Livingstone
Livingstone Model
Livingstone Requirement
Livingstone Trace
TRANSLATOR
SMV Model
SMV Requirement
SMV Trace
SMV

Information Sciences & Technology
Ames Research Center

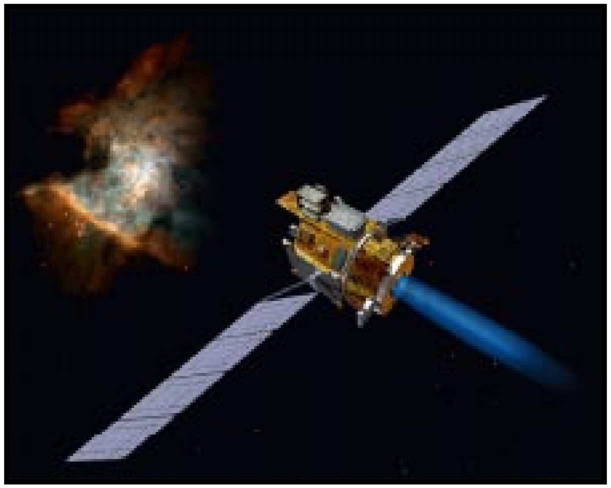
Autonomy

Past:

Time- stamped control sequences

Future:

On-board intelligence



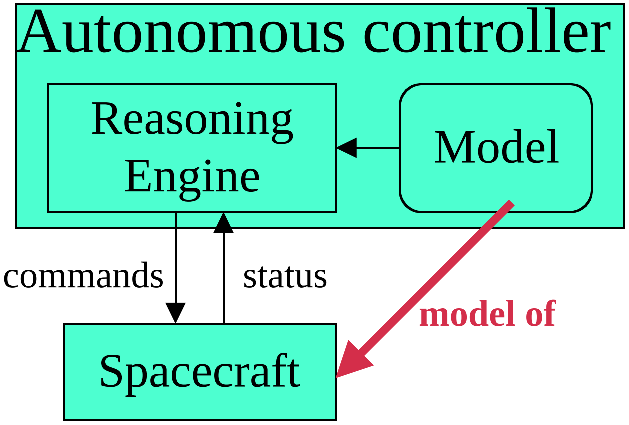
+ Can respond to unanticipated scenarios!

– How do we verify all those scenarios?

Concurrency => testing is not enough.

Model-Based Autonomy

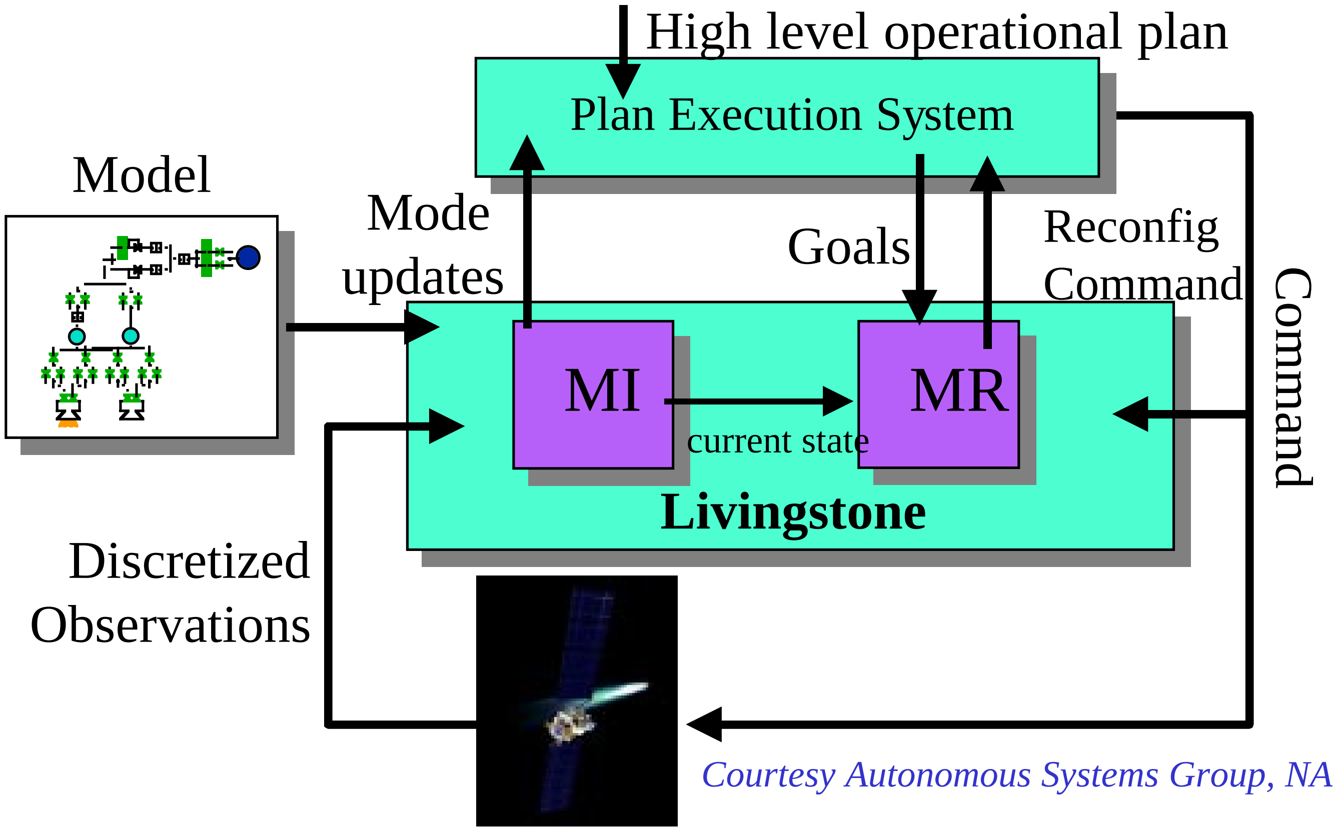
- Based on AI technology
- General reasoning engine + application-specific model
- Use model to respond to unanticipated situations



=> Verify the model !

The Livingstone MIR

Remote Agent's model-based fault recovery sub-system



High level operational plan

Plan Execution System

Model

Mode updates

Goals

Reconfig Command

Command

MI

MR

current state

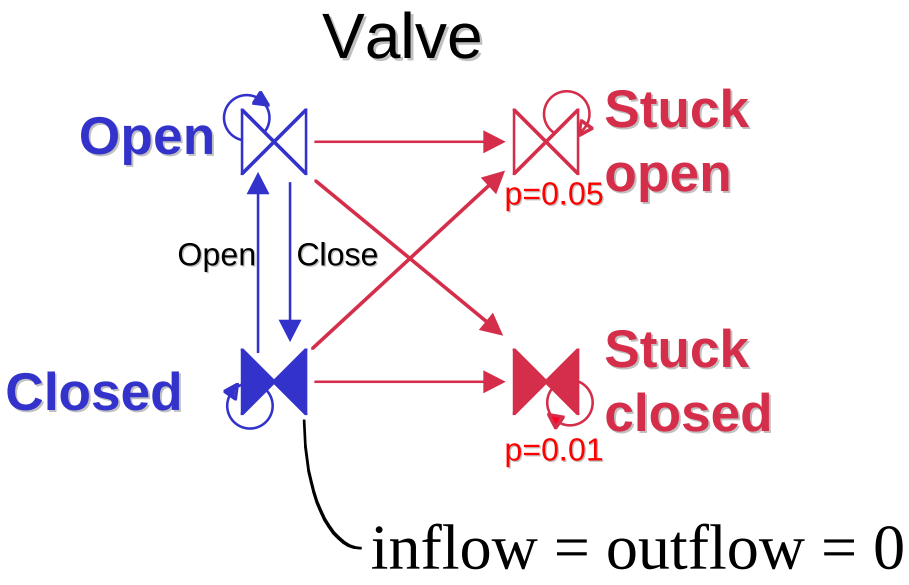
Livingstone

Discretized Observations

Courtesy Autonomous Systems Group, NASA Ames

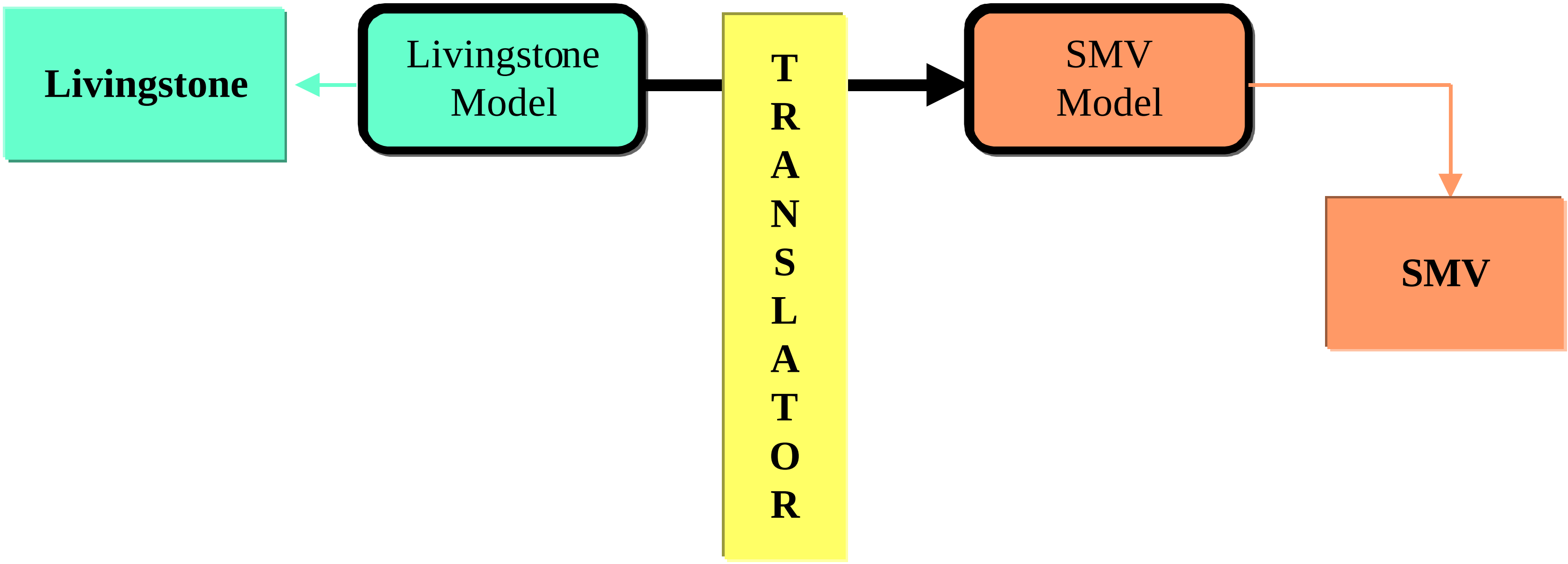
Livingstone Models

- Models = concurrent transition systems
- Qualitative values => finite state
- Nominal/fault modes
- Probabilities on faults



Courtesy Autonomous Systems Group, NASA Ames

Livingstone to SMV: Models



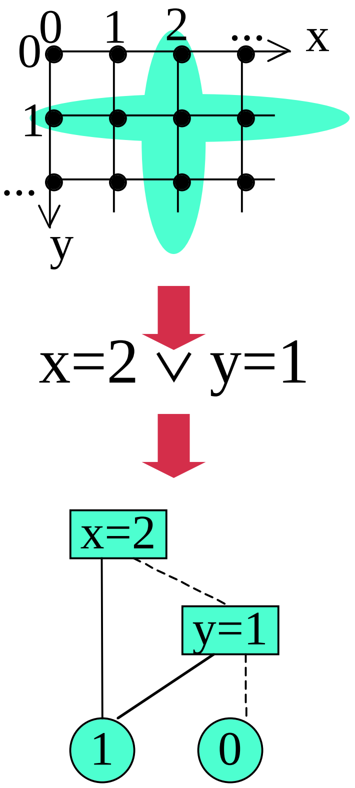
SMV

From Carnegie Mellon U. (Clarke, McMillan)

Does **Symbolic Model Checking**

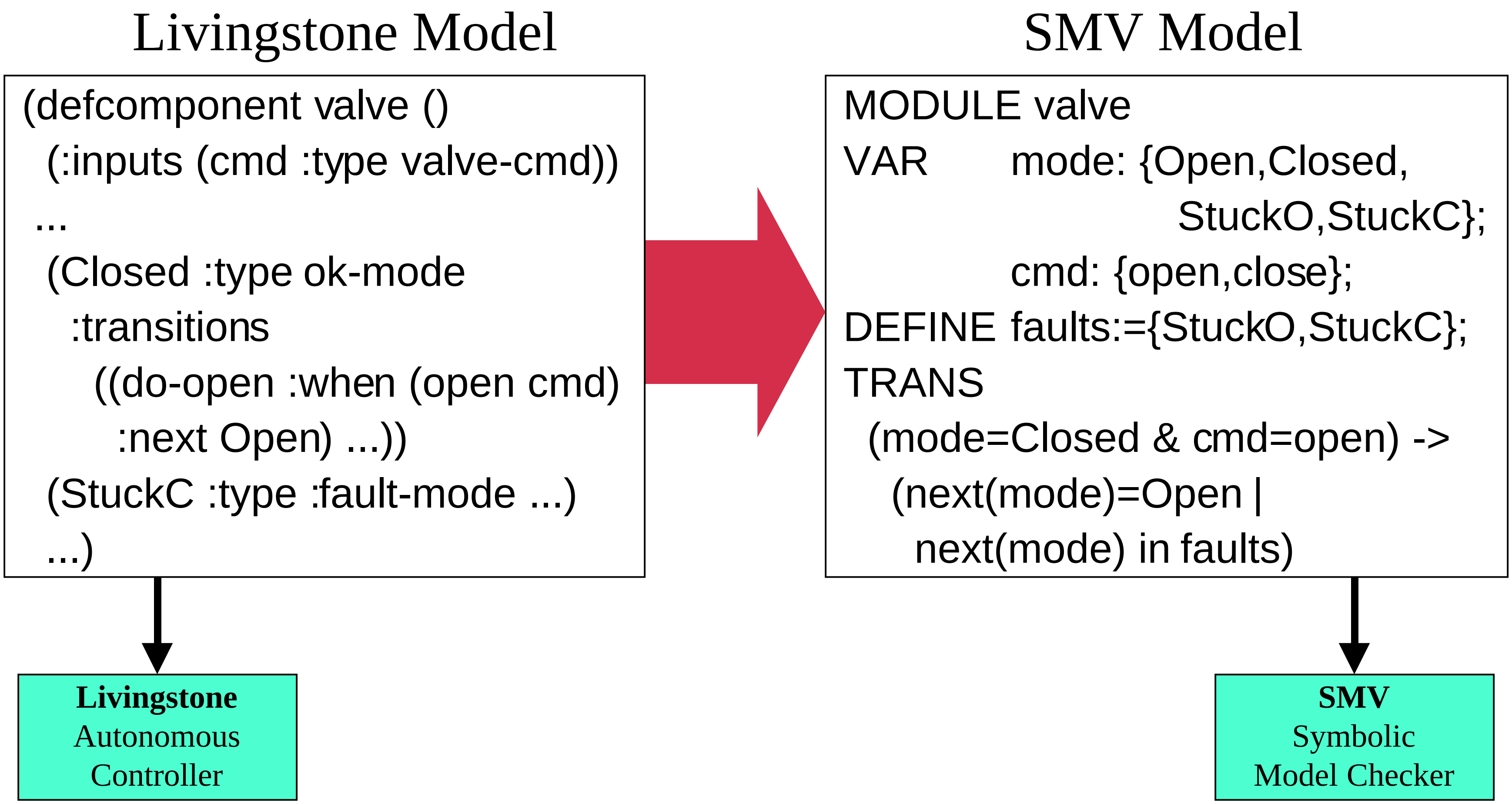
- Explore all states, BUT...
- Manipulates sets of states,
  Represented as boolean formulas,
  Encoded as Binary Decision Diagrams.
- BDD computations:
  - Good in average but exponential in worst case.
  - Computation time depends on BDD size
    => number of variables, complexity of formulas,
    but not directly state space size.

=> Can handle very large state spaces (10^{50+}).



$x=2 \vee y=1$

Translating Models



Livingstone Model

```
(defcomponent valve ()
  (:inputs (cmd :type valve-cmd))
 ...
  (Closed :type ok-mode
    :transitions
      ((do-open :when (open cmd)
        :next Open) ...))
  (StuckC :type :fault-mode ...)
  ...)
```

Livingstone Autonomous Controller

SMV Model

```
MODULE valve
VAR     mode: {Open,Closed,
                   StuckO,StuckC};
        cmd: {open,close};
DEFINE faults:={StuckO,StuckC};
TRANS
  (mode=Closed & cmd=open) ->
    (next(mode)=Open |
      next(mode) in faults)
```

SMV Symbolic Model Checker

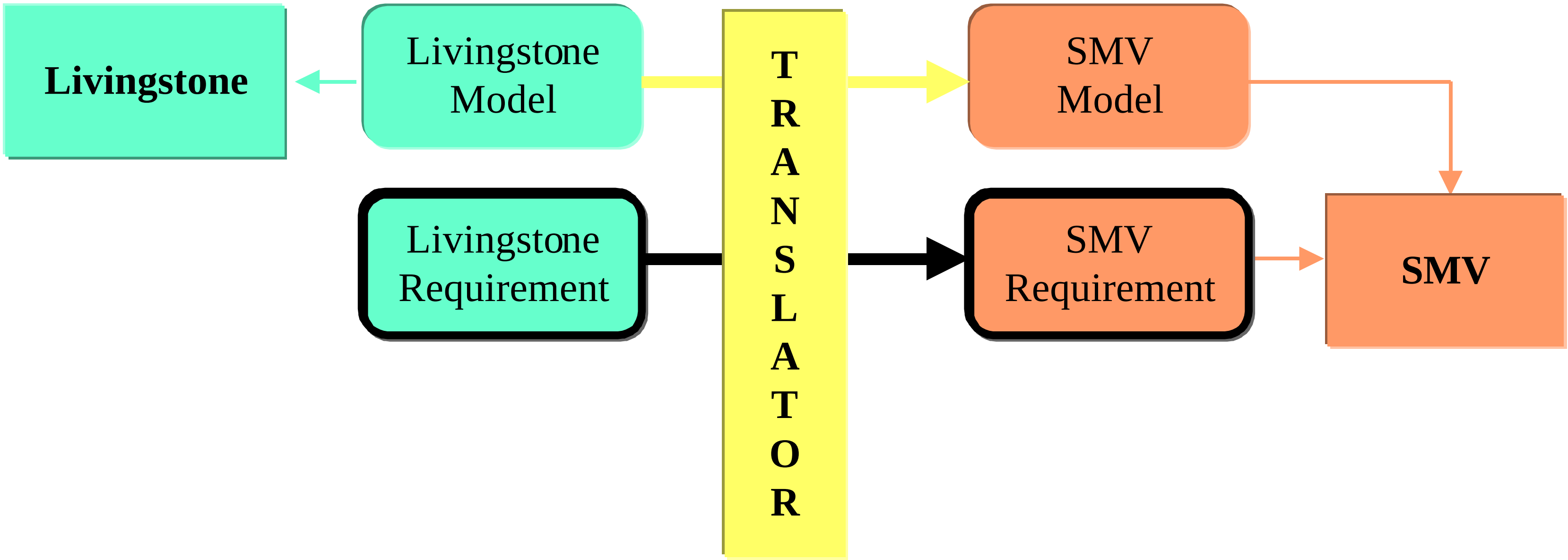
Implementation Notes

- 4K lines of Lisp
- Similar semantics (synchronous transition systems)

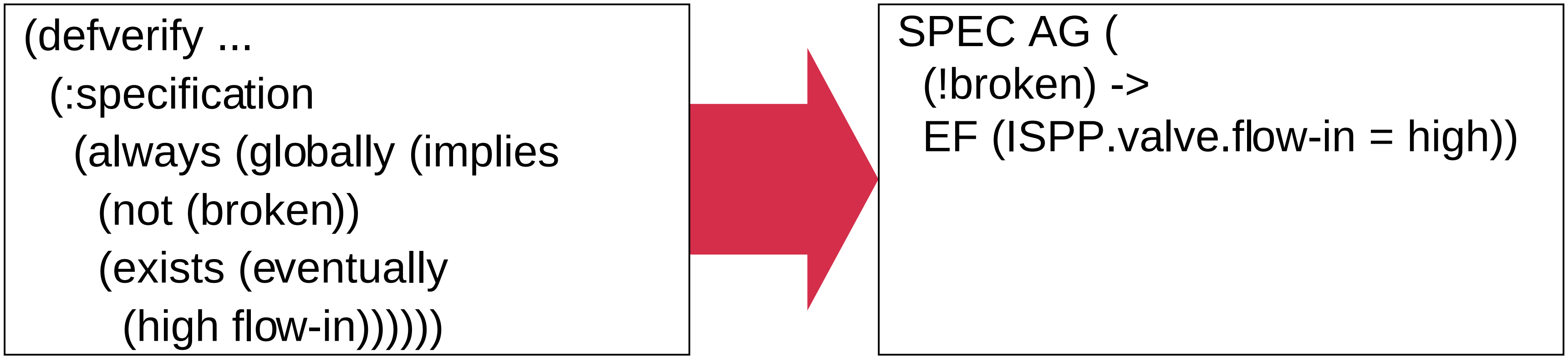
  => translation is fairly straightforward and one-to-one.
- Different naming and scoping rules

  => complex part is translation of variable names.
  Build and use lexicon of Livingstone vs. SMV variables.

Livingstone to SMV: Requirements



Translating Requirements



Livingstone Requirement

```
(defverify ...
  (:specification
    (always (globally (implies
      (not (broken))
      (exists (eventually
        (high flow-in))))))
```

SMV Requirement

```
SPEC AG (
  (!broken) ->
  EF (ISPP.valve.flow-in = high))
```

- Declaration (defverify ...) added to the Livingstone model.
- Temporal logic formulas (CTL) in Livingstone syntax
  + auxiliary predicates and patterns.

Auxiliary Predicates

(broken heater) = heater is in a failed state

(failed heater) = on last transition, heater failed

NB: failed more precise but requires extra SMV variable
=> SMV runs more slowly => optional

(multibroken 2) = at least two components are failed

(multicommand 2) = at least two commands are activated

(brokenproba 3) = combined probability of currently failed components is at least "of order" 3

NB: based on summation of approximate orders of magnitude
e.g. *n* stands for p=10^{-n}

Pre-Defined Patterns

(:specification :completeness ispp)

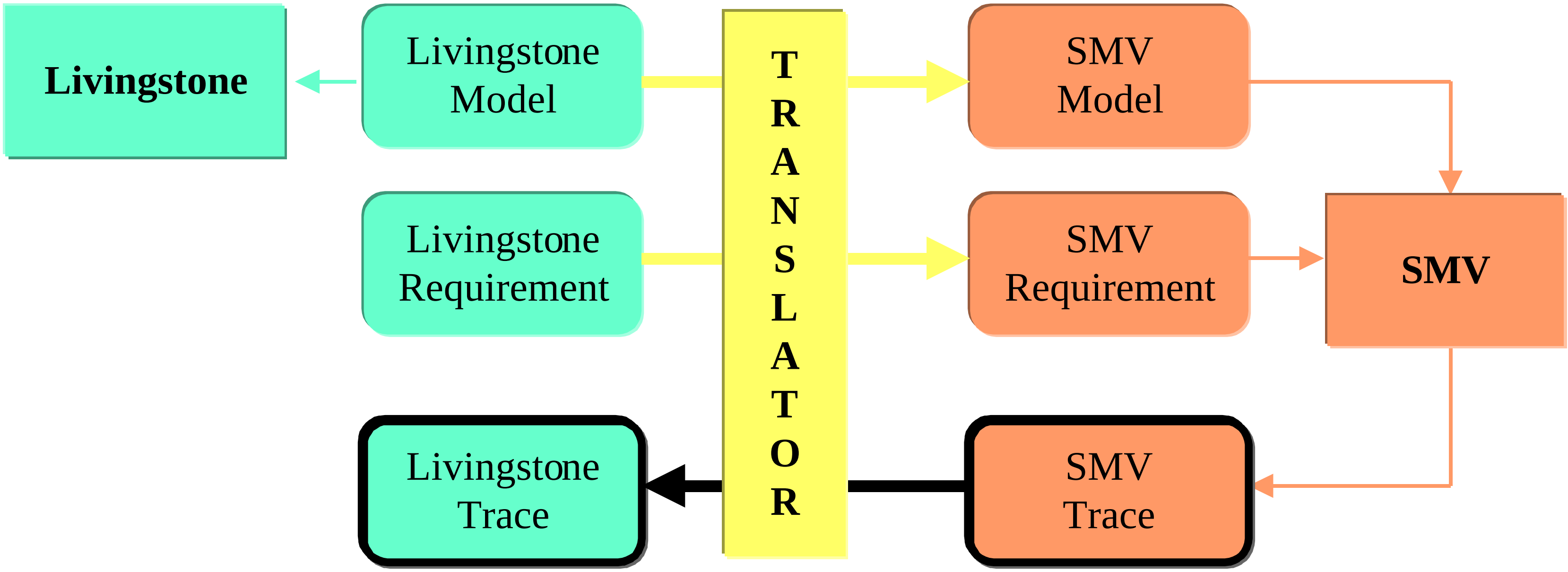
(:specification :disjointness ispp)

For each mode of each component of ispp, the conditions of all transitions are resp. complete and disjoint.

(:specification :reachability ispp)

All modes of all components of ispp are reachable from all initial states (variant :path-reachability from one state to another).

SMV to Livingstone: Diagnostic Traces



Closing the Loop

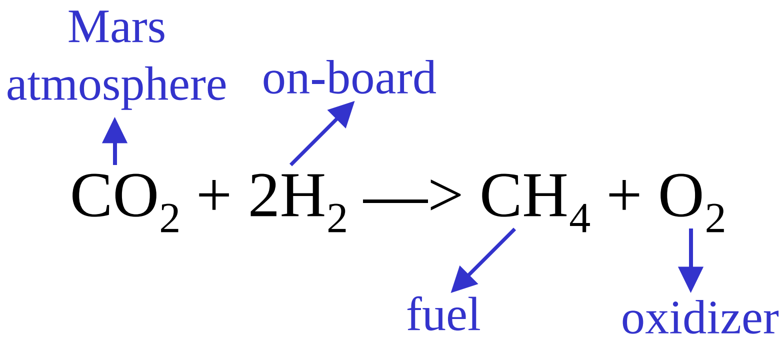
- Diagnostic traces = sequences of states.
- Translation uses lexicon backwards.
- Completes the Livingstone ↔ SMV bridge
  => isolates Livingstone users from SMV syntax.
- In progress (CMU):
  generate causal explanations of traces.

Application
In-Situ Propellant Production

- Use atmosphere from Mars to make fuel for return flight.
- Livingstone controller developed at NASA KSC.
- Components are tanks, reactors, valves, sensors...
- Exposed improper flow modeling.
- Latest model is 10^{50} states.

See poster!

(Peter Engrand)



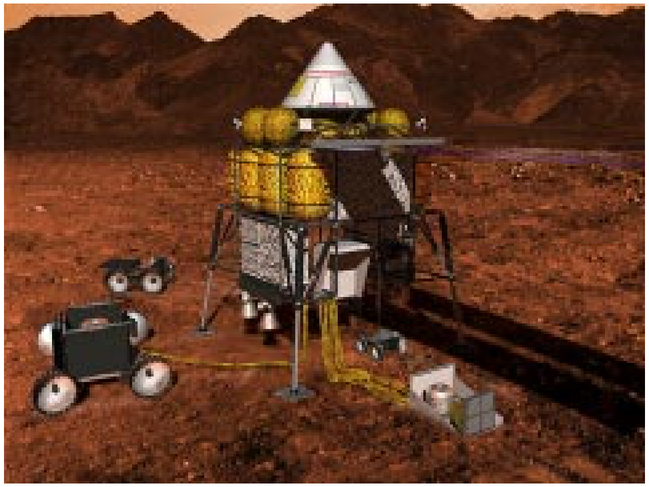
Mars atmosphere

on-board

$$CO_2 + 2H_2 \longrightarrow CH_4 + O_2$$

fuel

oxidizer



Conclusions

Symbolic model checking for models used in autonomous fault recovery system.

- Works well because:
  - Models are already abstract,
  - Similar semantics.
- Full forward and backward translation

  => shields Livingstone users from SMV details.

To Probe Further

- Improved accuracy for V&V (w.r.t. testing) ?
  - Complements (rather than replaces) testing.
- Methodology, what to look for:
  - Not deadlocks.
  - Consistency/completeness.
  - Responsiveness: can a failure be observed?

  Tools are available, needs more user experience.