



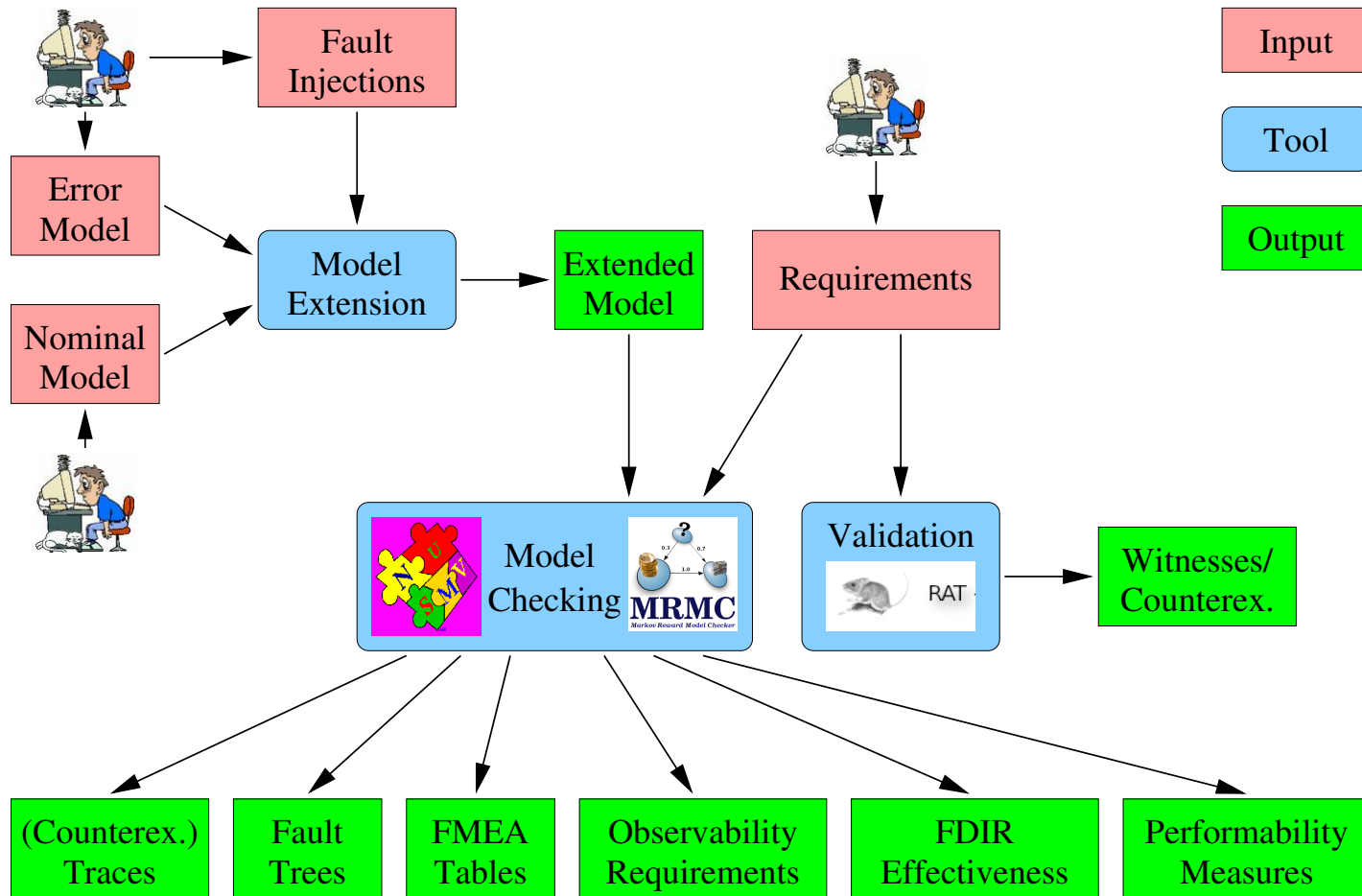
# Error Modeling in COMPASS

**Harold Bruintjes** ([h.bruintjes@cs.rwth-aachen.de](mailto:h.bruintjes@cs.rwth-aachen.de))

**COMPASS Workshop @ ESA/ESTEC**  
**22 October 2015**



# COMPASS Methodology



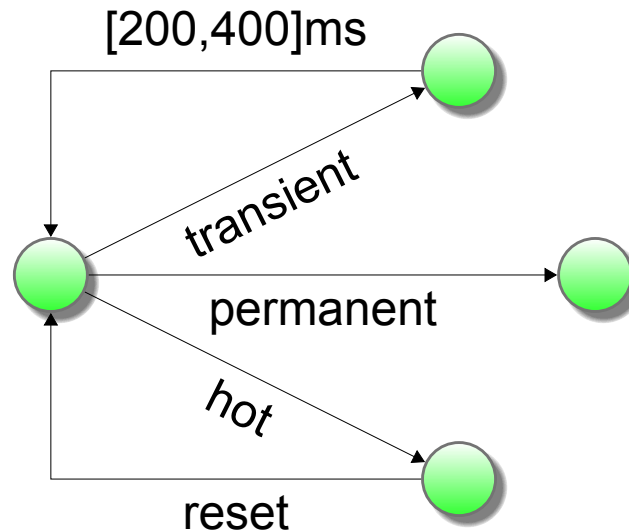
# Error Models

---

## Example

Error model with transient, hot and permanent faults/failures

- Transient: Recovery after some delay;
- Hot: Recovery after restart/reset of device;
- Permanent: Recovery not possible, device lost.



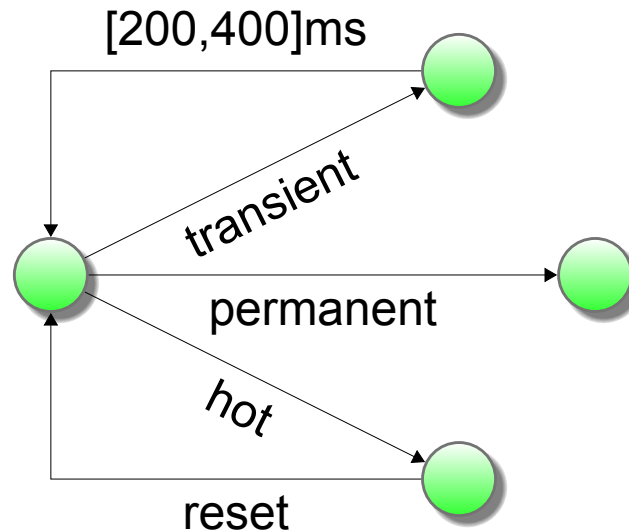
# Error Models

---

## Example

Error model with transient, hot and permanent faults/failures

- Transient: Recovery after some delay;
- Hot: Recovery after restart/reset of device;
- Permanent: Recovery not possible, device lost.



---

Probabilistic error occurrences : Faults happen after some random delay.

# Error Models

---

```
-- Error model with transient, hot and permanent failures
error model gpsError
  features
    nok : out error propagation;
end gpsError;

error model implementation gpsError.i
```

```
end gpsError.i;
```

# Error Models

---

```
-- Error model with transient, hot and permanent failures
error model gpsError
  features
    nok : out error propagation;
end gpsError;

error model implementation gpsError.i
  events
    transient_fault      : error event occurrence poisson 0.001 per hour;
    hot_fault            : error event occurrence poisson 0.001 per day;
    permanent_fault     : error event occurrence poisson 0.001 per day;

end gpsError.i;
```

# Error Models

---

```
-- Error model with transient, hot and permanent failures
error model gpsError
  features
    nok : out error propagation;
end gpsError;

error model implementation gpsError.i
  events
    transient_fault      : error event occurrence poisson 0.001 per hour;
    hot_fault            : error event occurrence poisson 0.001 per day;
    permanent_fault     : error event occurrence poisson 0.001 per day;
  states
    ok                   : initial state;
    transient_failure_prop : error state;
    transient_failure    : error state urgent in 400 msec;
    hot_failure_prop     : error state;
    hot_failure          : error state;
    permanent_failure_prop : error state;
    permanent_failure    : error state;

end gpsError.i;
```

# Error Models

---

```
-- Error model with transient, hot and permanent failures
error model gpsError
  features
    nok : out error propagation;
end gpsError;

error model implementation gpsError.i
  events
    transient_fault      : error event occurrence poisson 0.001 per hour;
    hot_fault            : error event occurrence poisson 0.001 per day;
    permanent_fault     : error event occurrence poisson 0.001 per day;
  states
    ok                   : initial state;
    transient_failure_prop : error state;
    transient_failure    : error state urgent in 400 msec;
    hot_failure_prop     : error state;
    hot_failure          : error state;
    permanent_failure_prop : error state;
    permanent_failure    : error state;
  transitions
    ok                   -[ transient_fault ]-> transient_failure_prop;
    transient_failure_prop -[ nok ]-> transient_failure;
    transient_failure    -[ nok within 200 msec to 400 msec ]-> ok;
    ok                   -[ hot_fault ]-> hot_failure_prop;
    hot_failure_prop     -[ nok ]-> hot_failure;
    hot_failure          -[ @activation ]-> ok;
    transient_failure    -[ @activation ]-> ok;
    ok                   -[ permanent_fault ]-> permanent_failure_prop;
    permanent_failure_prop -[ nok ]-> permanent_failure;
end gpsError.i;
```



# Tool Support

The screenshot displays the COMPASS Toolset interface with the following sections:

- Loaded Files:** A table with columns 'Filename' containing 'sensorfilter.slim' and 'sensorfilterErr.slim'. Buttons: 'Reload All', 'Remove', 'Add'.
- FDIR Components:** A table with columns 'Implementation' and 'Filename'. Row: '✓ \_\_default\_\_::Monitor.Impl sensorfilter.slim'.
- Root:** A table with columns 'Implementation' and 'Filename'. Row: '✓ \_\_default\_\_::Acquisition.Impl sensorfilter.slim'.
- Fault Injections:** A table with columns 'Use', 'Error Implementation', 'Error State', and 'Effect'.

Use	Error Implementation	Error State	Effect
<input checked="" type="checkbox"/>	__default__::SensorFailures.Impl	Dead	sensors.sensor1.output := 50
<input checked="" type="checkbox"/>	__default__::SensorFailures.Impl	Glitched	sensors.sensor1.output := output + 10
<input checked="" type="checkbox"/>	__default__::SensorFailures.Impl	Drifted1	sensors.sensor1.output := output + 1
<input checked="" type="checkbox"/>	__default__::SensorFailures.Impl	Drifted2	sensors.sensor1.output := output + 2
<input checked="" type="checkbox"/>	__default__::SensorFailures.Impl	Dead	sensors.sensor2.output := 50
<input checked="" type="checkbox"/>	__default__::SensorFailures.Impl	Glitched	sensors.sensor2.output := output + 10
<input checked="" type="checkbox"/>	__default__::SensorFailures.Impl	Drifted1	sensors.sensor2.output := output + 1
<input checked="" type="checkbox"/>	__default__::SensorFailures.Impl	Drifted2	sensors.sensor2.output := output + 2

Buttons: 'Remove', 'Clone', 'Add'.
- Output Console:** A text area showing:

```
Compiling 'sensorfilter.slim'... OK
Compiling 'sensorfilterErr.slim'... OK
Loading fault injections 'sensorfilter.fixml'... OK
> Loaded 8 of 8 fault injections.
```

Buttons: 'Compiler', 'Logging', 'Extended Model', 'Metrics'.

# Tool Support

The screenshot displays the COMPASS Toolset interface. The main window is titled "COMPASS Toolset" and features a menu bar with "File", "Edit", "View", "Activities", and "Help". Below the menu bar are tabs for "Model", "Properties", "Mission", "TFPG", "Validation", "Correctness", "Perfomability", "Safety", and "FDIR".

The interface is divided into several panels:

- Loaded Files:** A table with columns "Filename" and "Implementation". It lists "sensorfilter.slim" and "sensorfilterErr.slim". Buttons for "Reload All", "Remove", and "Add" are present.
- FDIR Components:** A table with columns "Implementation" and "Filename". It shows a checked entry for "\_\_default\_\_::Monitor.Impl" linked to "sensorfilter.slim".
- Root:** A table with columns "Implementation" and "Filename". It shows a checked entry for "\_\_default\_\_::Acquisition.Impl" linked to "sensorfilter.slim".
- Output Console:** A text area showing compilation and loading messages, such as "Compiling 'sensorfilter.slim'... OK" and "Loading fault injections 'sensorfilter.fixml'... OK".
- Fault Injections:** A table with columns "Use", "Error Implementation", "Error State", and "Effect". It lists several entries, all checked, with states like "Dead" and "Glitched", and effects like "sensors.sensor1.output := 50".

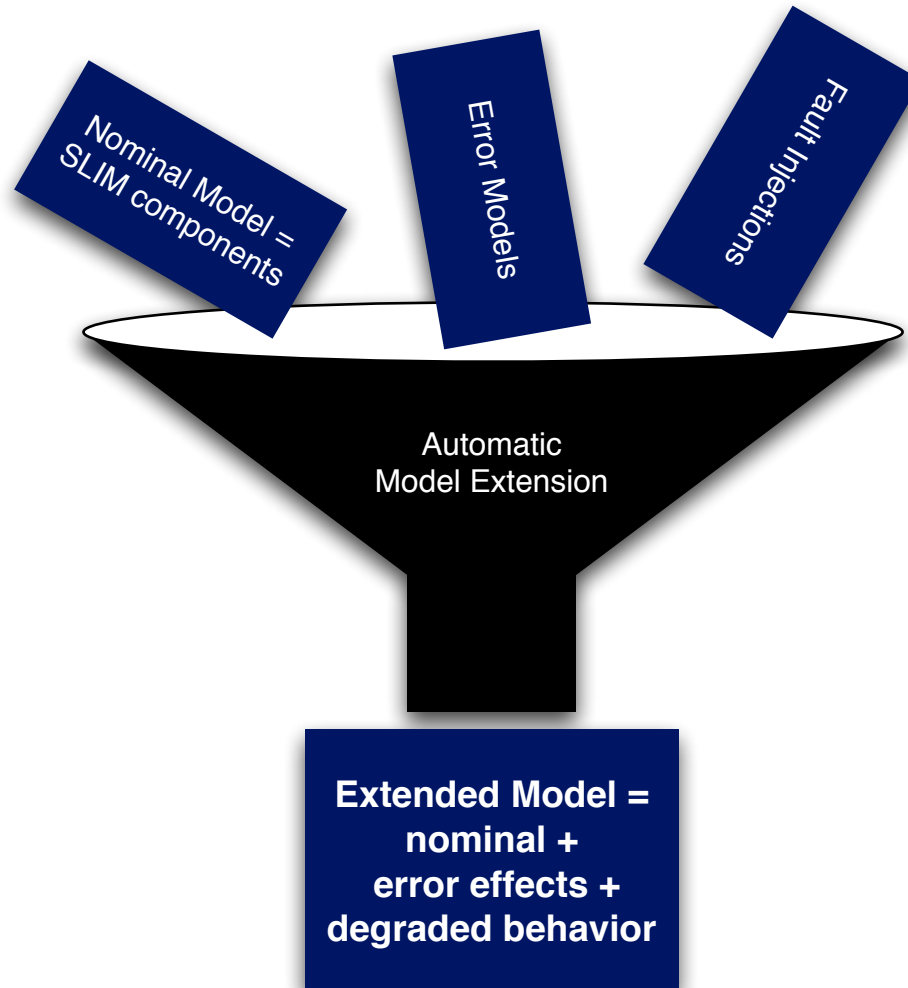
A "New Fault Injection" dialog box is open in the foreground, showing the configuration for a new injection:

- Error Model:** Implementation: "\_\_default\_\_::SensorFailures.Imp", State: "Drifted2".
- Nominal Model:** Instance: "filters", Data element: "output", Effect: "33".

Buttons for "Discard" and "Inject" are at the bottom of the dialog.

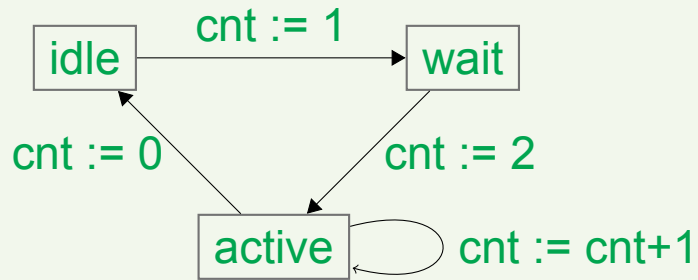
# Integrating Erroneous and Nominal Behaviour

---

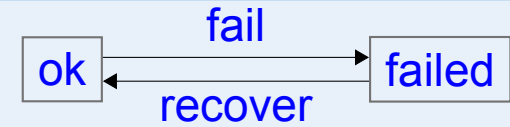


# Model Extension by Example

## Nominal behaviour

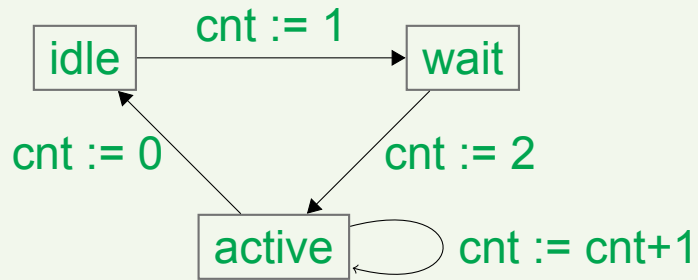


## Error behaviour



# Model Extension by Example

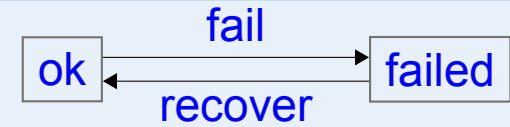
## Nominal behaviour



## Fault injection

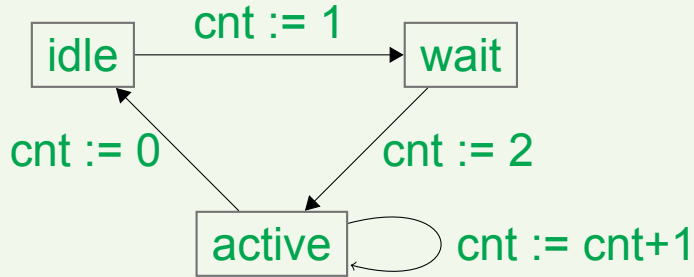
failed:  $cnt := -1$

## Error behaviour



# Model Extension by Example

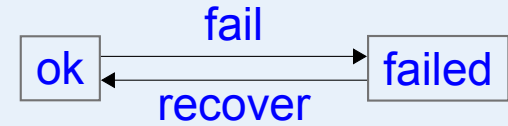
## Nominal behaviour



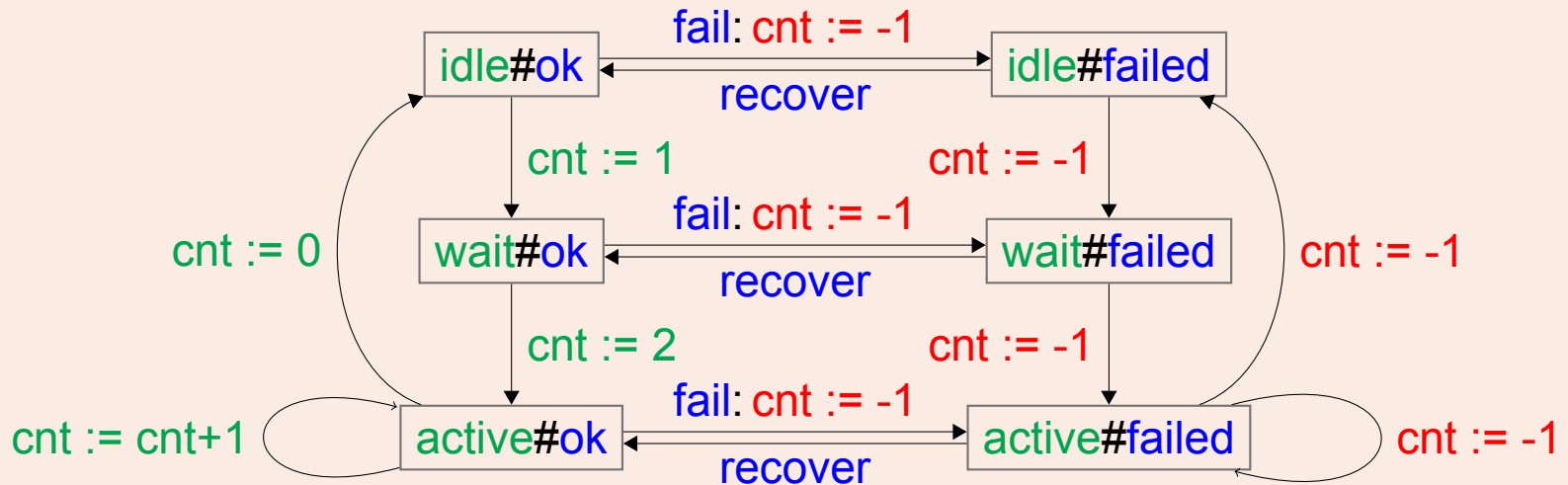
## Fault injection

failed: cnt := -1

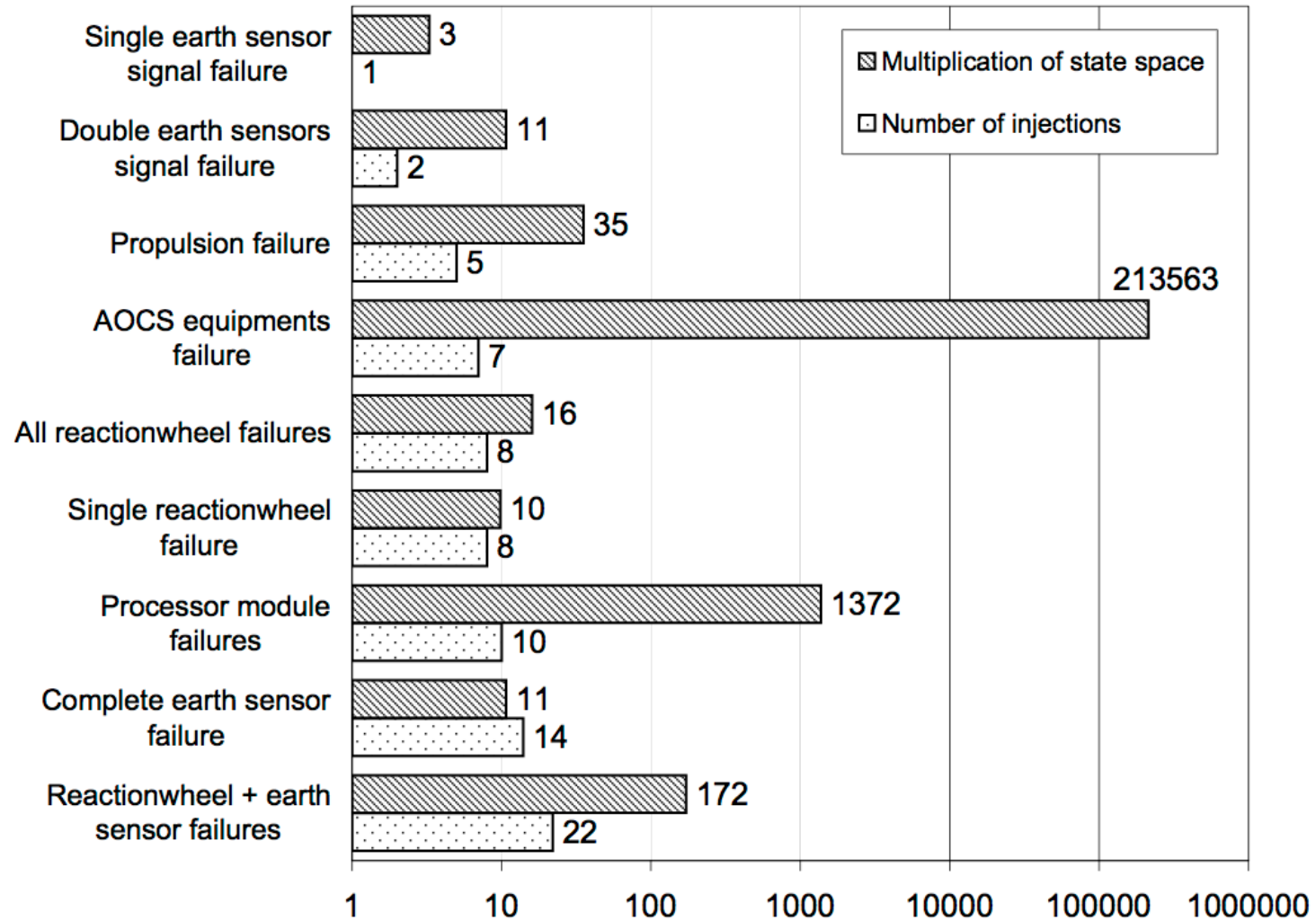
## Error behaviour



## Extended model



# Effect of Fault Injections (from Satellite Case Study [ICSE 2012])



# Contribution to AADL Error Model Annex (V2): Error State History

---

- EMA:

`initial error state | error state`



# Contribution to AADL Error Model Annex (V2): Error State History

---

- EMA:

initial error state | error state

- COMPASS:

initial state | activation state | error state

- **initial state**: like **initial error state**, but **supporting error state history**
  - after reactivation of component, error model resumes error state from last deactivation
  - appropriate for hardware components
- **activation state**: like **initial error state**, but **without error state history**
  - after reactivation of component, error model reset to activation state
  - appropriate for software components
- **error state**: like EMA **error state**

# Possible Future Extensions

---

## Current fault injection mechanism

$s: d := f$  means that while being in error state  $s$ , the assignment  $d := f$  is active, where  $d$  is a data element and  $f$  the fault effect.

# Possible Future Extensions

---

## Current fault injection mechanism

$s: d := f$  means that while being in error state  $s$ , the assignment  $d := f$  is active, where  $d$  is a data element and  $f$  the fault effect.

## Possible extensions to increase expressiveness and usability

- **Transient** fault effects
  - only when entering state  $s$ , the assignment  $d := f$  is performed
- **Error-triggered** nominal transitions
  - error transition  $s \rightarrow s'$  causes mode transition  $m \rightarrow m'$
- **Disabling** of nominal transitions
  - while being in error state  $s$ , nominal event  $e$  is disabled

# The End

---

