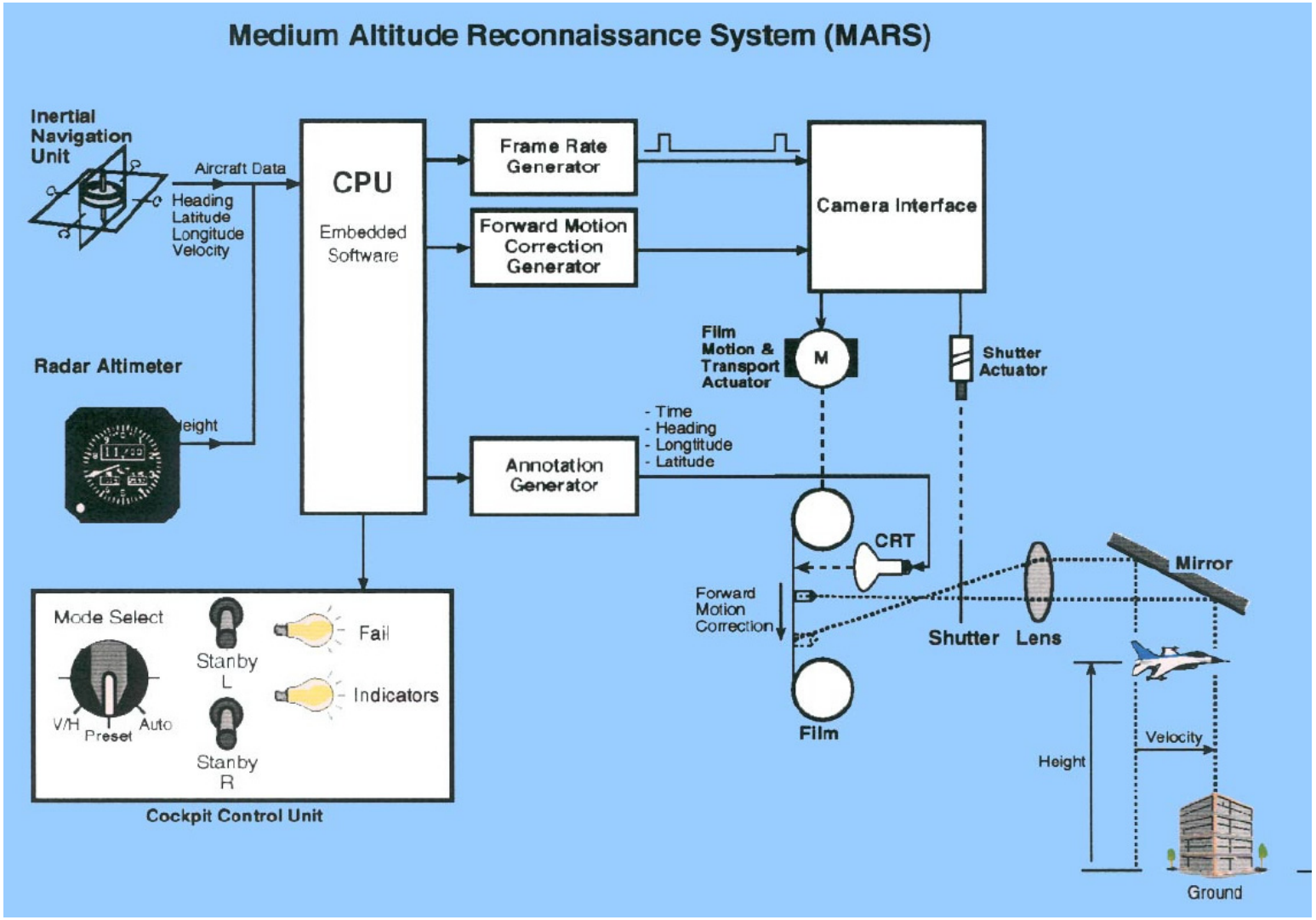
Timing analysis and validation with UML: the case of the embedded MARS bus manager

Iulian Ober
IRIT, Univ. of Toulouse
Susanne Graf
CNRS/VERIMAG, Grenoble
Yuri Yushtein
ESA/ESTEC Noordwijk
Ileana Ober
IRIT, Univ. of Toulouse

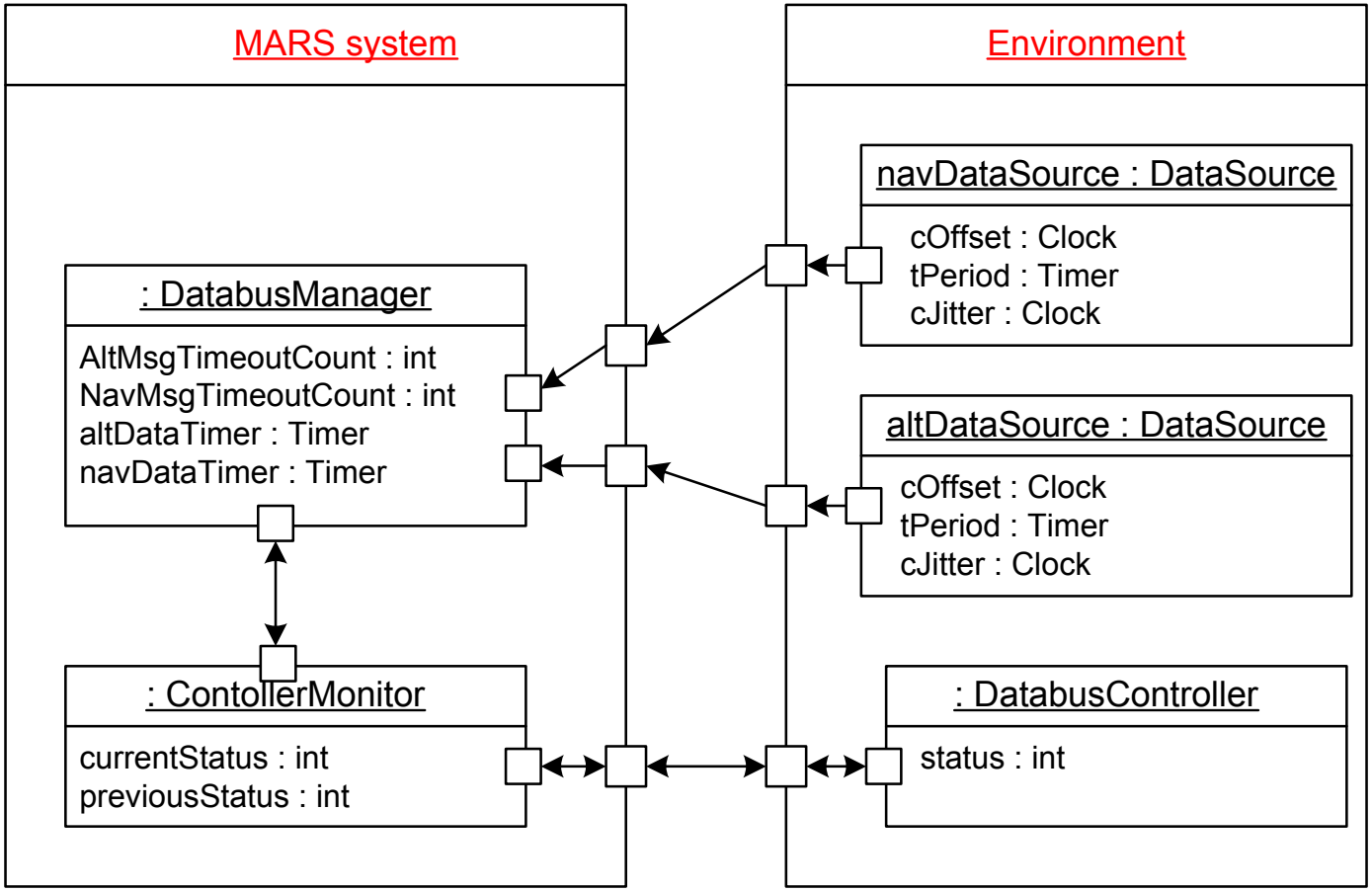
Overview

- MARS system :
  overview and requirements
- UML model and overview of tools
- Verification experiments
- Conclusions

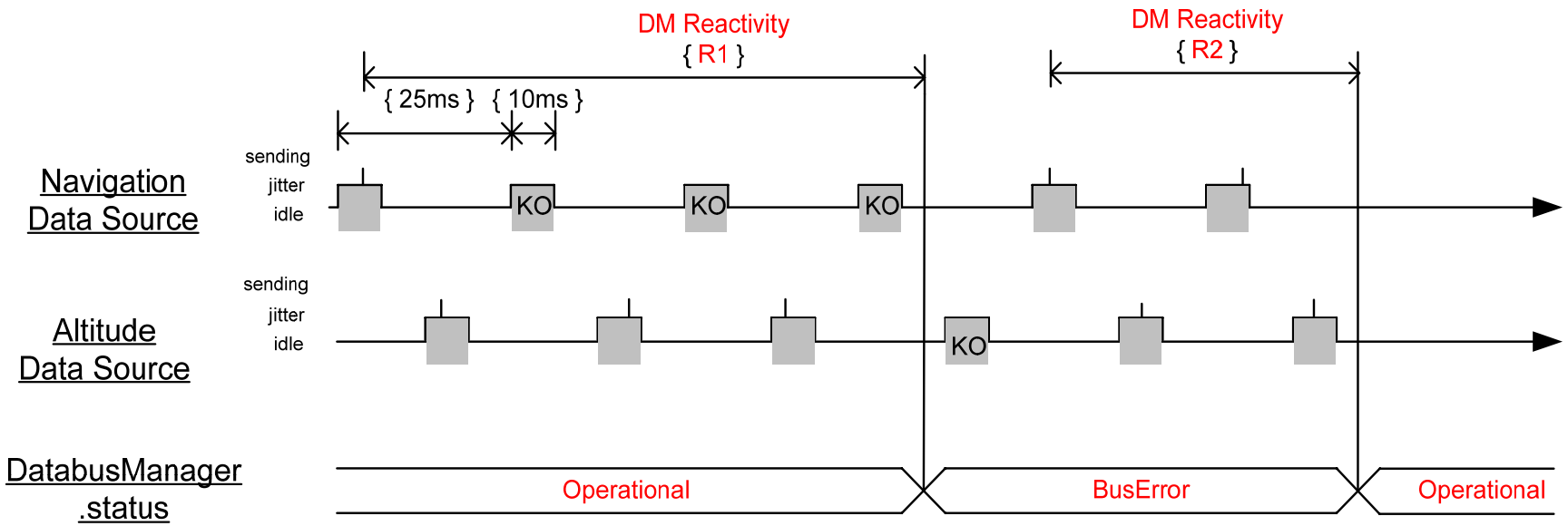
MARS overall architecture



Model architecture



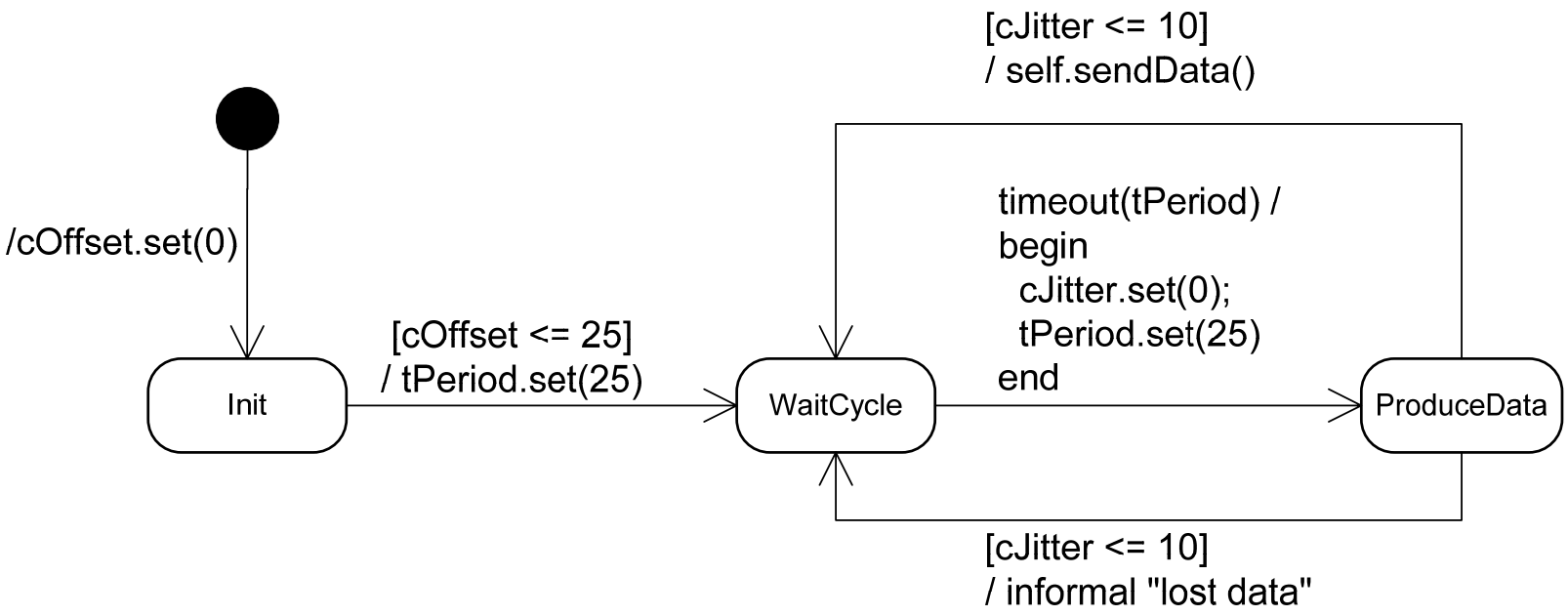
Data bus manager: timing requirements



UML and tools

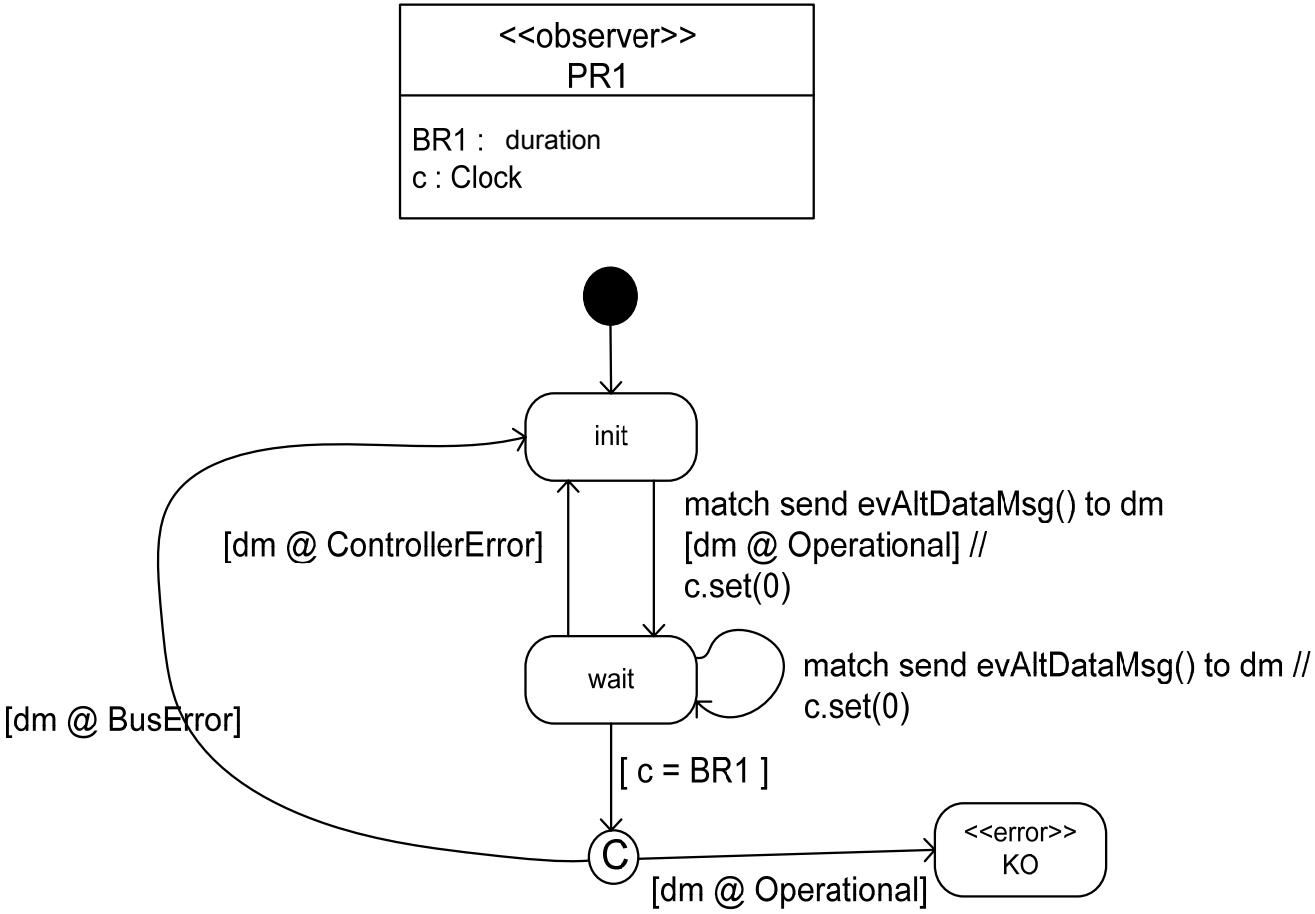
- OMEGA UML profile [http://www-omega.imag.fr/]
  (executable semantics)
  - active components, state machines
  - asynchronous signals & synchronous calls
  - precise timing (TA w/ urgency), dynamic priority
  - safety properties : observers
- IFx toolset [http://www-if.imag.fr/]
  - simulation
  - model checking
  - test generation,...

Environment model : data sources

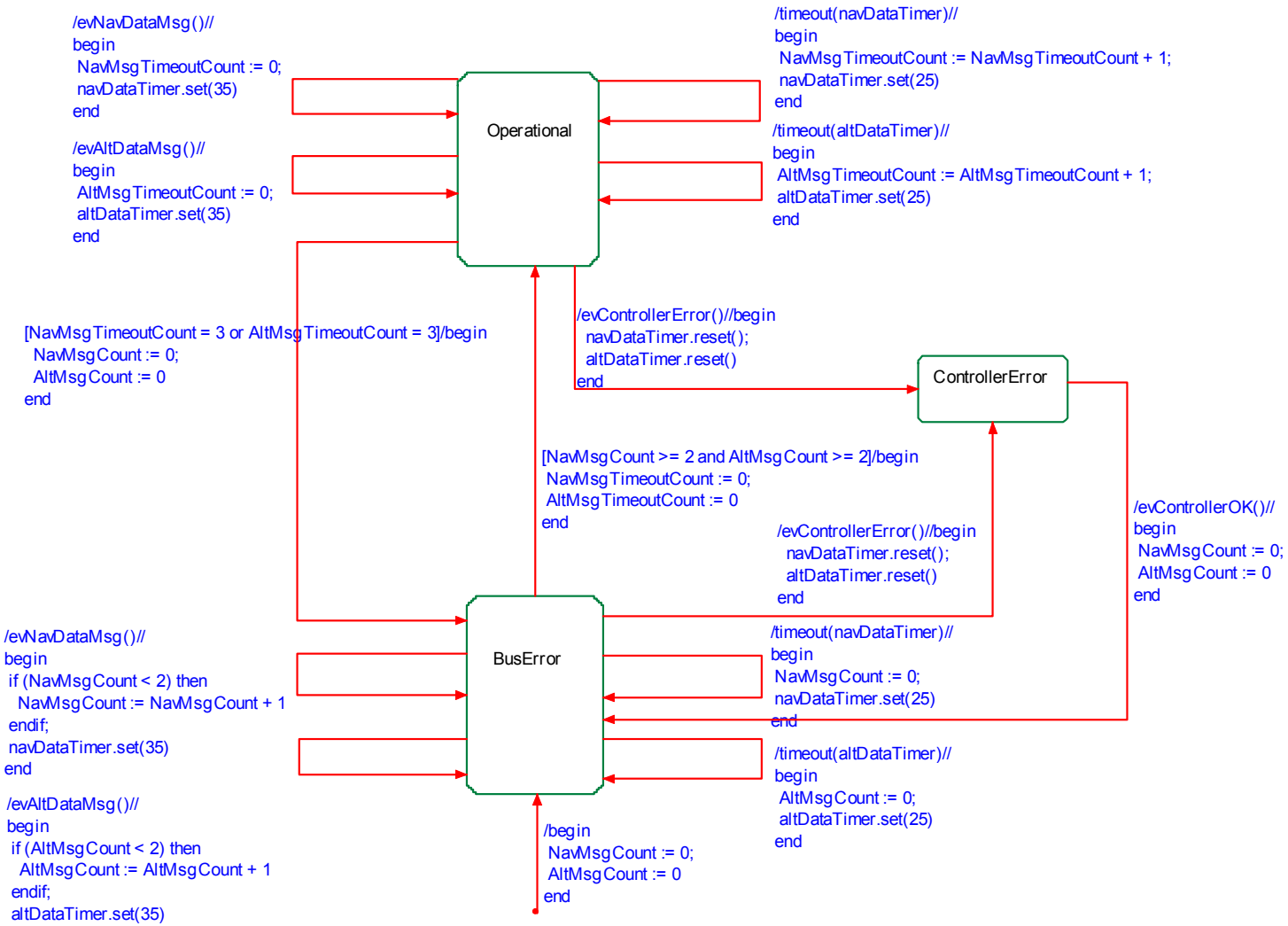


property : measuring reactivity R1

Absence of transmission is discovered within delay BR1

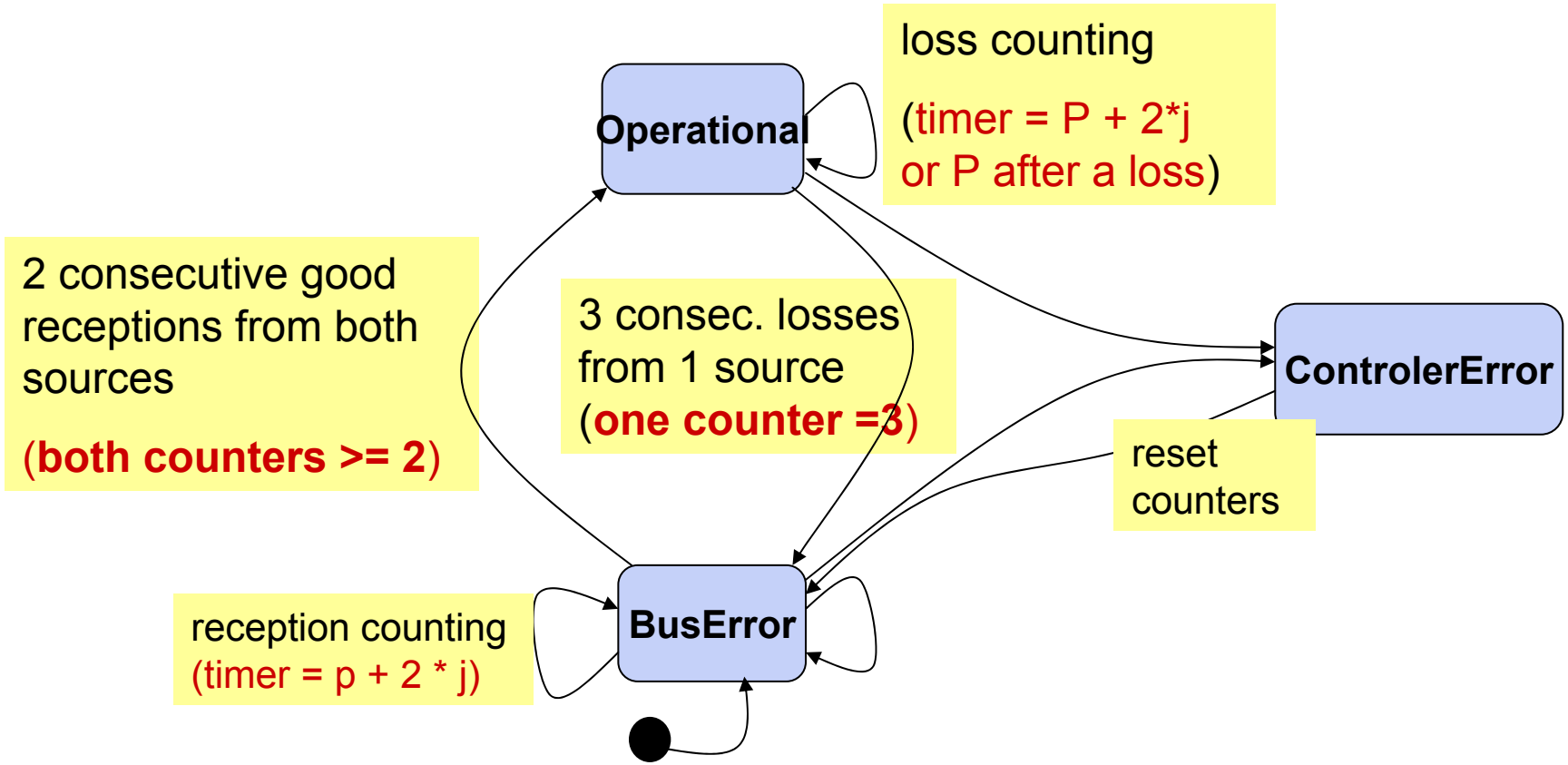


Model of the Databus Manager



Abstract model, Version 1: for every sender

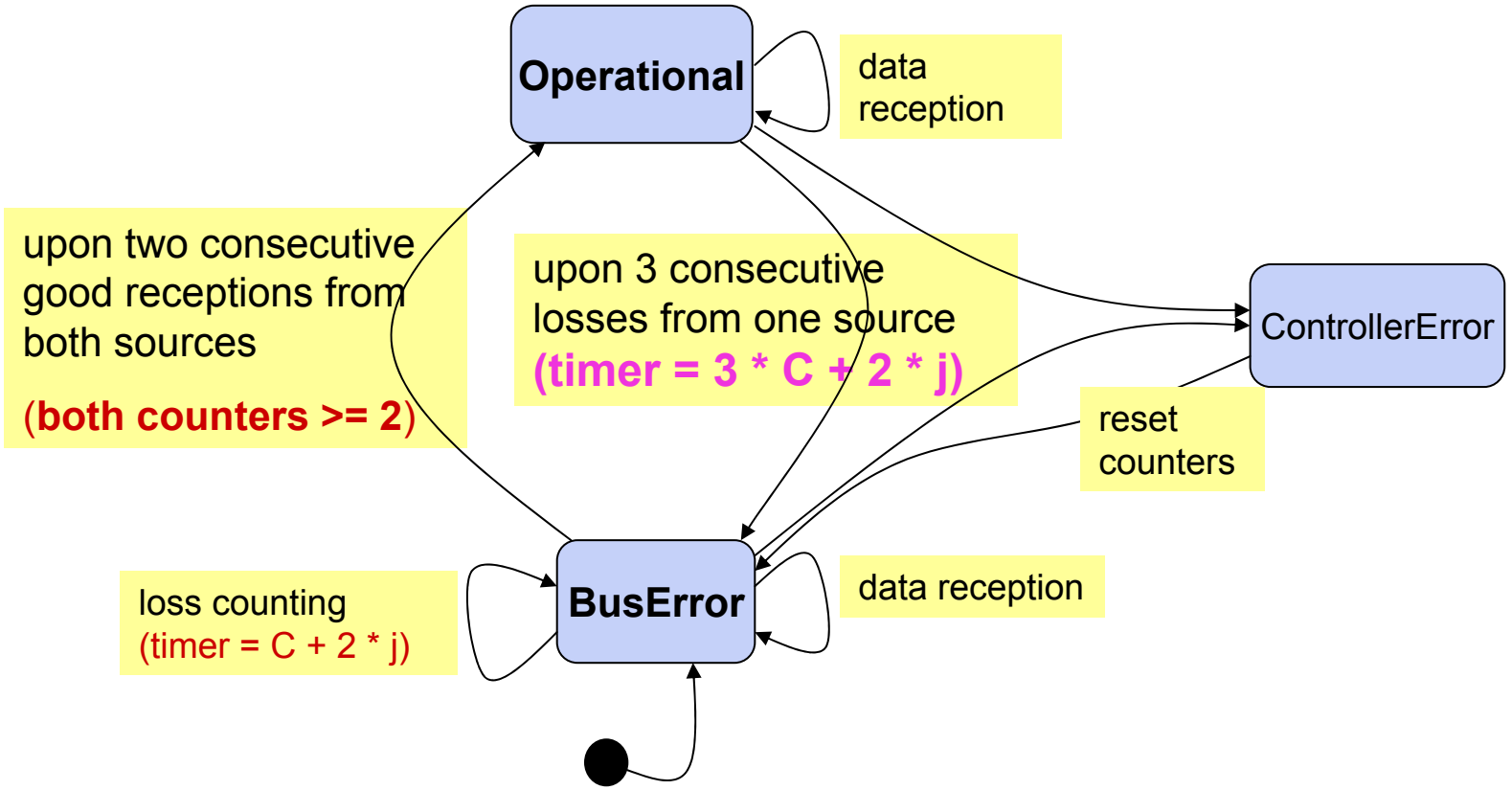
- Determine for every period “signal ok” or “signal loss” and count them
  - in state operational: count consecutive losses
  - in state BusError: count consecutive receptions
- Depending on counter values decide if status switch is needed



Verification shows: reactivity property holds

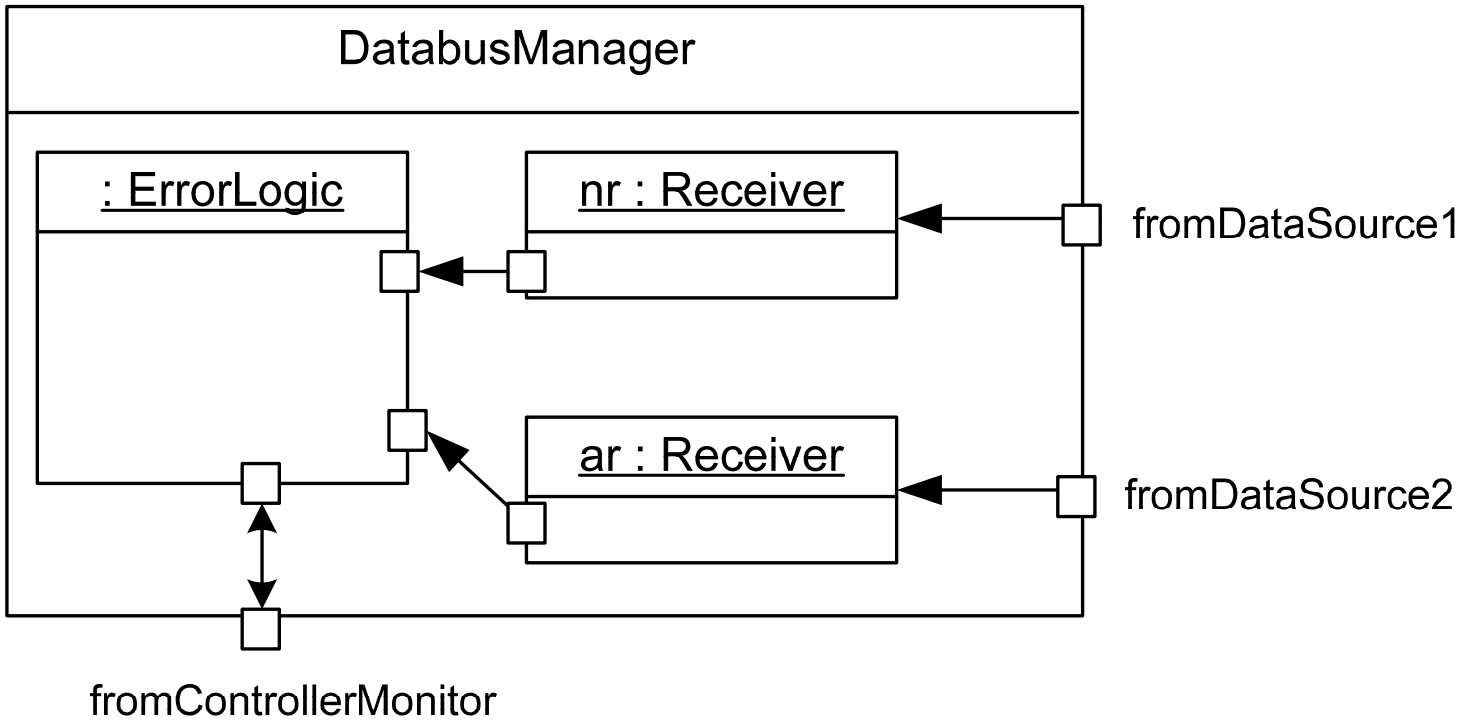
Version 2: Asymmetric

- In state ***operational***, look for a "long timeout" (3*C +2*j) – no need for counter
- in state ***BusError***, no change



Verification shows: this version is less reactive

compositional re-design
⇒ better abstraction results



- Error logic parametric in the number of receivers
- Project property to a single receiver
- Consider 1 concrete receiver and abstract all others

Verification parameters and results

Configuration	Number of states	Number of transitions	User time
Initial model with only one source (no CM polling) (*non-conservative*)	1084	1420	$< 1s$
Initial model with two synchronized sources (no CM polling) (*non-conservative*)	99355	151926	$36s$
Initial model with two de-synchronized sources (no CM polling) (*conservative* – does not finish)	> 1136768	> 1676126	$> 9m30s$
Abstract model, 10ms CM polling (*conservative* – does not finish)	> 1494864	> 701120	$> 8m12$
Abstract model with no CM polling (*non-conservative*)	118690	174871	$45s$
Abstract model with lazy CM polling (*conservative*)	155166	263368	$1m21s$

Conclusions

- Push button verification for whole systems – not there yet !
- Still useful for analysis of "hard points"
- Essential success factors :
  - "user-friendly" language and tools
  - use of abstractions
    - easy ones (~ automatic) : timing relaxation, …
    - harder ones : design refactoring ← "design for verification" skills