Universität Stuttgart

DaimlerChrysler

Verification in the Design Process of Large Real-Time Systems: A Case Study

Pascal Montag & Dirk Nowotka

Overview

- Motivation
- Goals
- Introduction to Timed Automata
- Using Timed Automata in the Design Process
- Conclusion

Motivation

- Increasing extent of areas of software use
  - Increase in safety relevant applications → Need for verification methods
- Increasing complexity of software systems
  - Distributed systems → Dependencies beyond ECU borders
  - Real-Time systems → Timing constraints are hard to verify
- Increasing need of automated verification
- Approach of Timed Automata
  - Lack of appropriate case studies → No large systems so far

Overview

- Motivation
- Goals
- Introduction to Timed Automata
- Using Timed Automata in the Design Process
- Conclusion

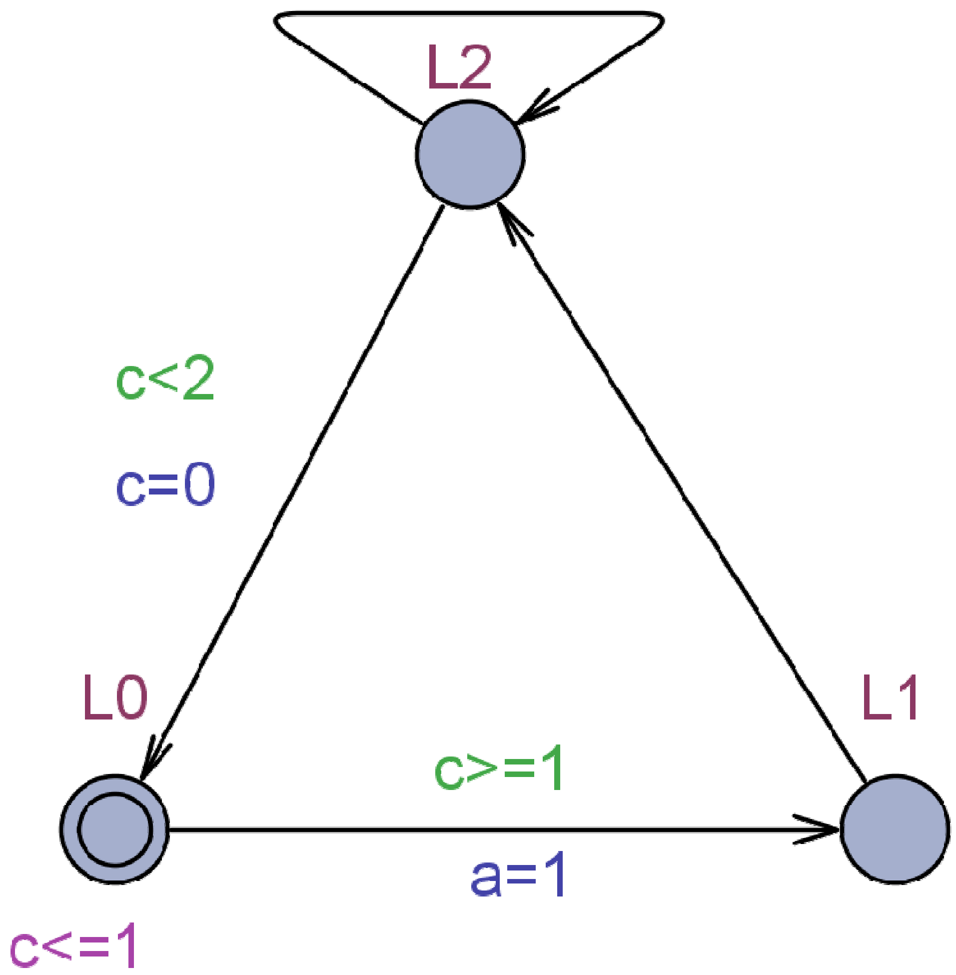
Goals

- Modelling of a large Real-Time system
  - System should be
    - Distributed (real world of automotive systems)
    - Safety relevant (results have to be useful)
    - Time dependent (verification should not be trivial)
- Verification of safety relevant properties

Overview

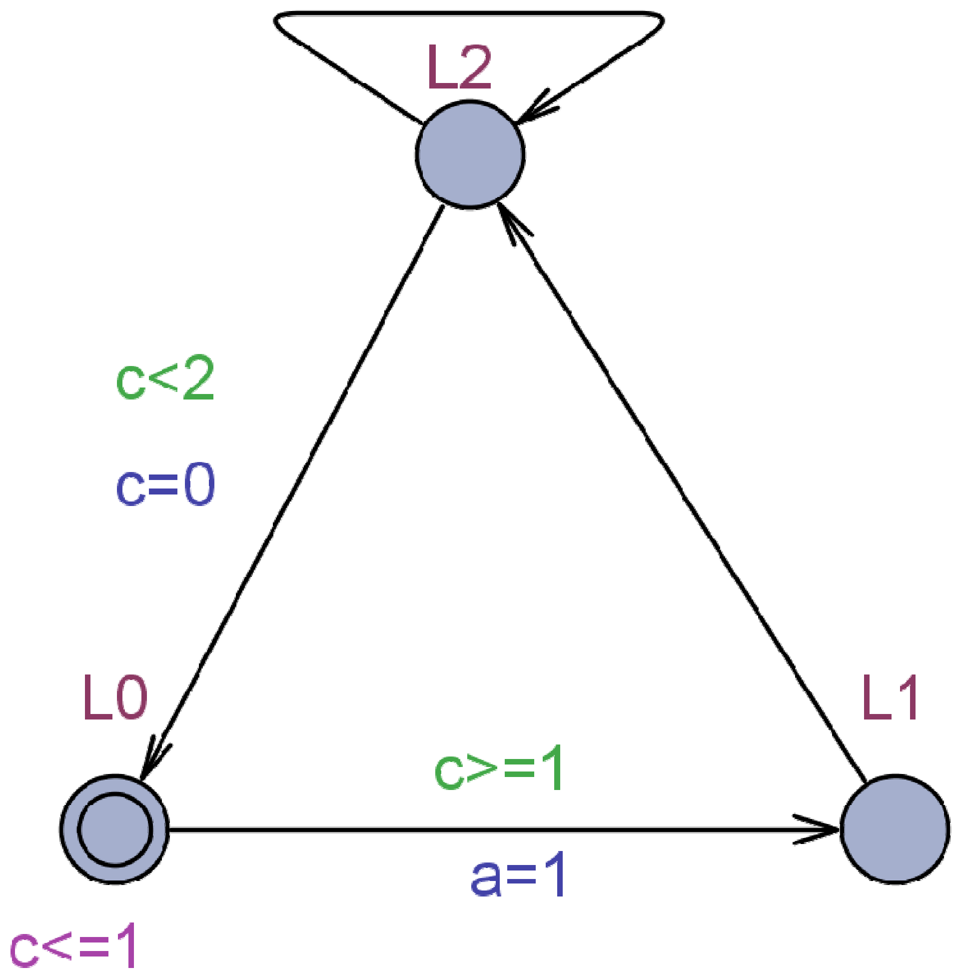
- Motivation
- Goals
- Introduction to Timed Automata
- Using Timed Automata in the Design Process
- Conclusion

Introduction to Timed Automata – Definition by Example



Introduction to Timed Automata – UPPAAL Specification Language

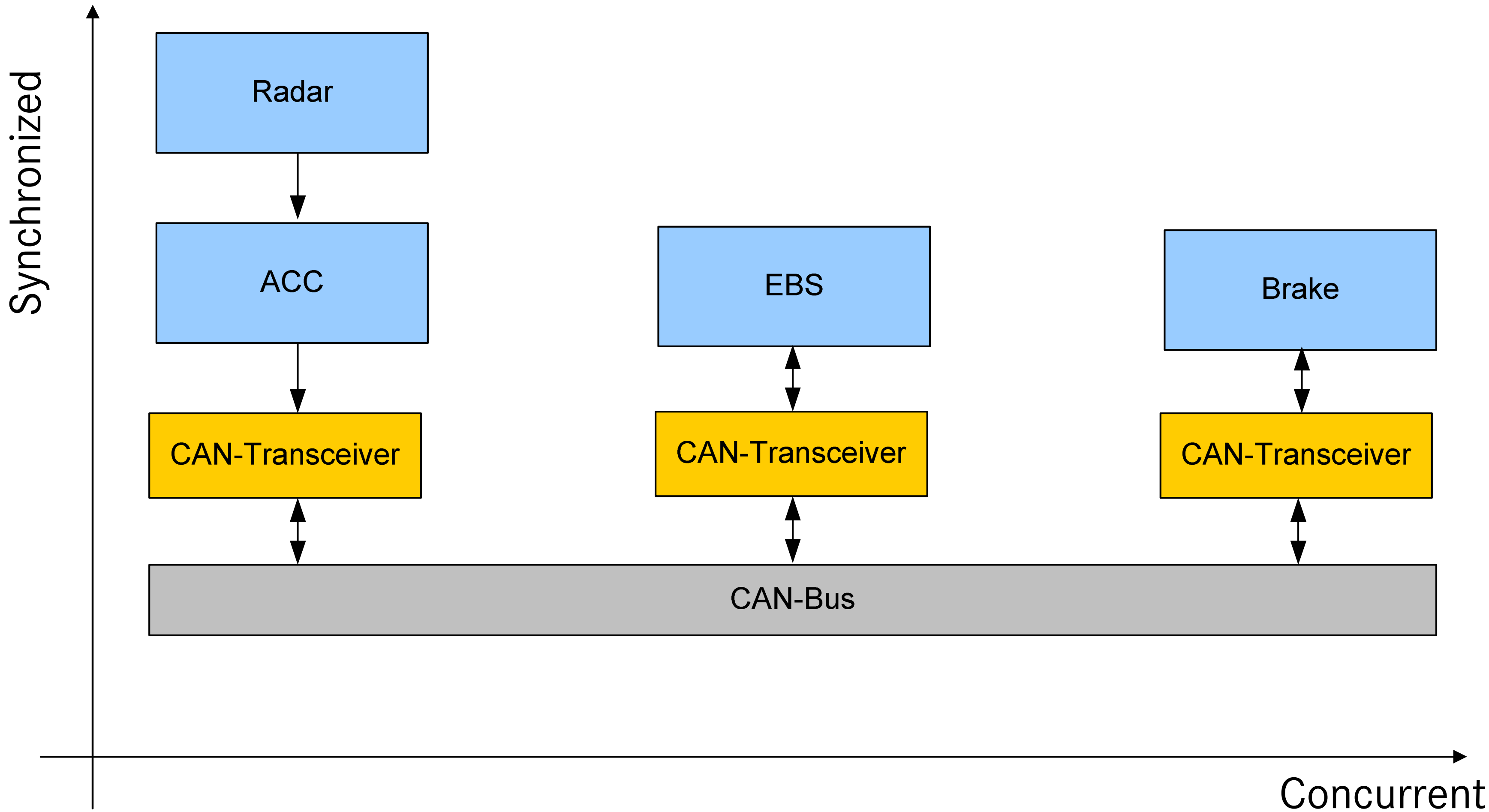
- A[] not deadlock
- L2 → L0
- A[] L2 imply c>=1
- E<> L2 and c<1



Overview

- Motivation
- Goals
- Introduction to Timed Automata
- Using Timed Automata in the Design Process
  - The Case Study System: An Emergency Brake Assistant
  - Properties to Be Verified
  - First Step: Basic Model Specification
  - Verifying Properties for the Overall System
  - Refinement Step
- Conclusion

The Case Study System: An Emergency Brake Assistant



Properties to Be Verified

1. Will the emergency brake be activated after an emergency situation has been sensed for a maximum time of 30ms?

   Radar.Close → (Brake.EmergencyBrake *and* CloseTimer <= 30ms)

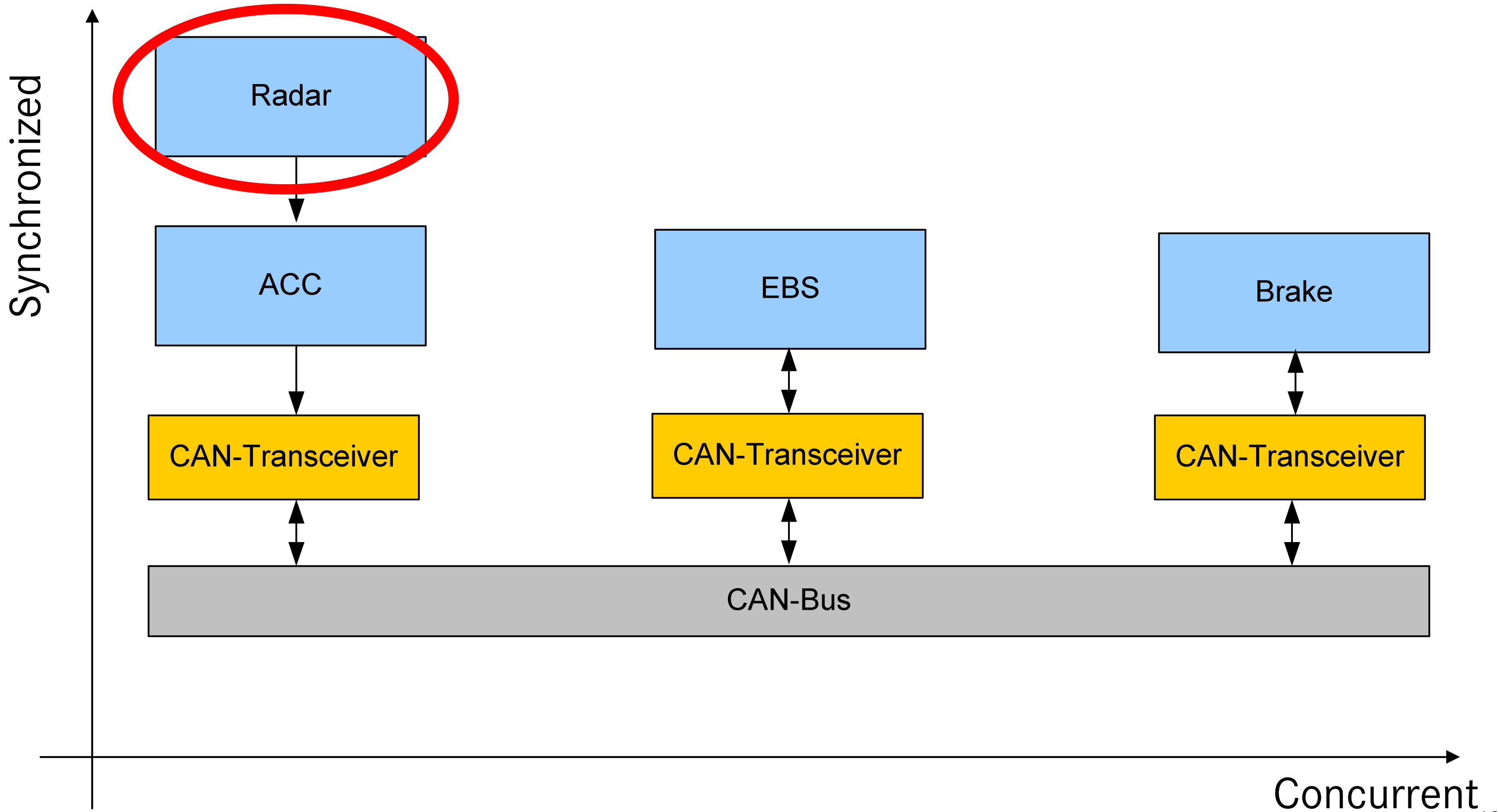
2. Do ECU tasks consume less time than their period lengths (deadline)?

   A[] (ECU.TaskFinished *imply* ECU.Timer <= Deadline)

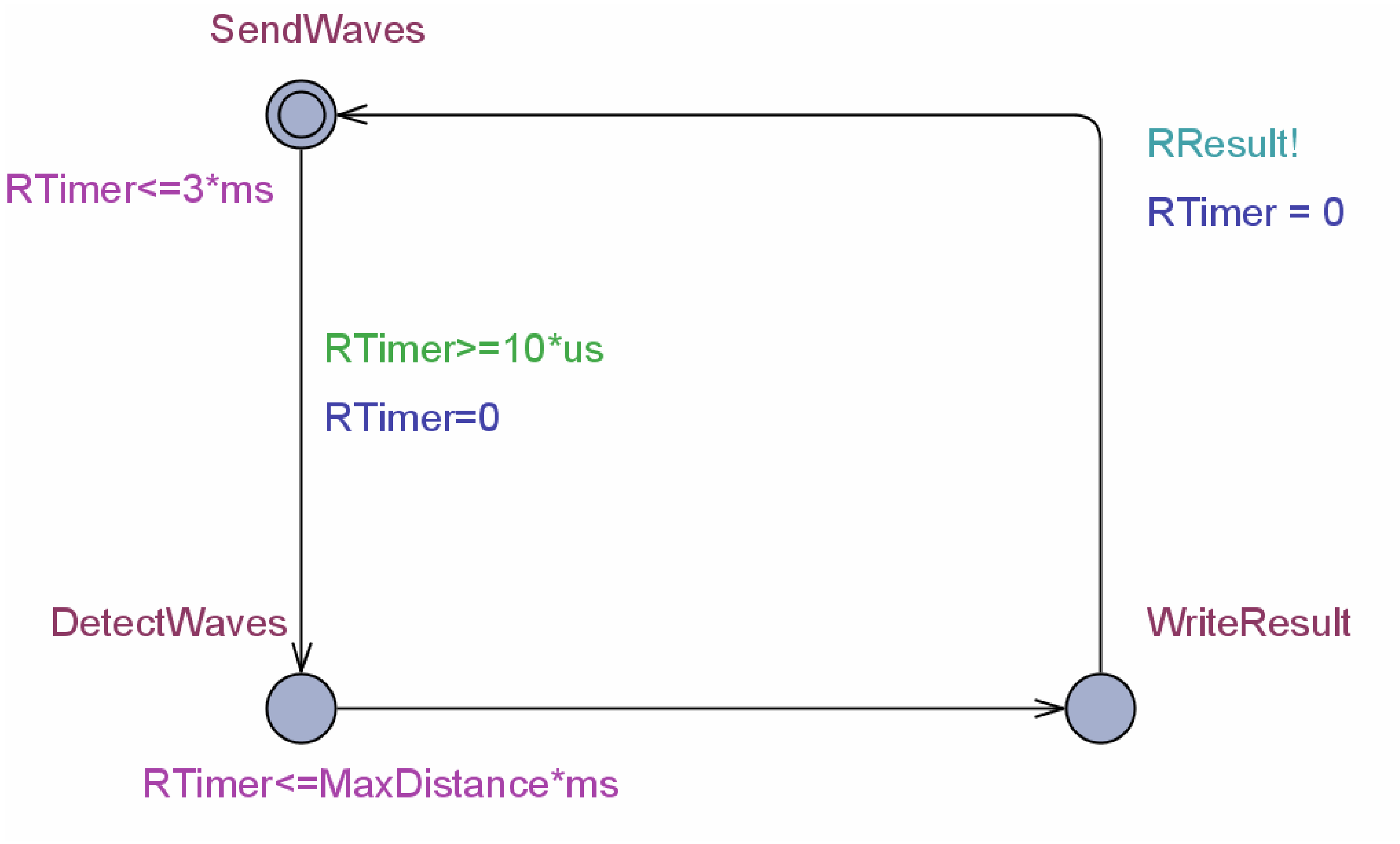
3. Is the system deadlock free?

   A[] (*not* deadlock)

First Step: Basic Model Specification Exemplified on Radar Model Part



First Step: Basic Model Specification Exemplified on Radar Model Part



Verifying Properties for the Overall System

1. Radar.Close → (Brake.EmergencyBrake and (CloseTimer <= 30ms))
   - Satisfied (CloseTimer does not exceed 27.83ms)
2. A[] (not deadlock)
   - Satisfied
3. A[] (ACC.TaskFinished imply (ACCTimer <= 5ms))
   - Not satisfied (ACCTimer can reach 5.344ms)
4. A[] (EBS.TaskFinished imply (EBSTimer <= 10ms))
   - Satisfied
5. A[] (Brake.TaskFinished imply (BrakeTimer <= 5ms))
   - Satisfied

Verifying Properties for the Overall System

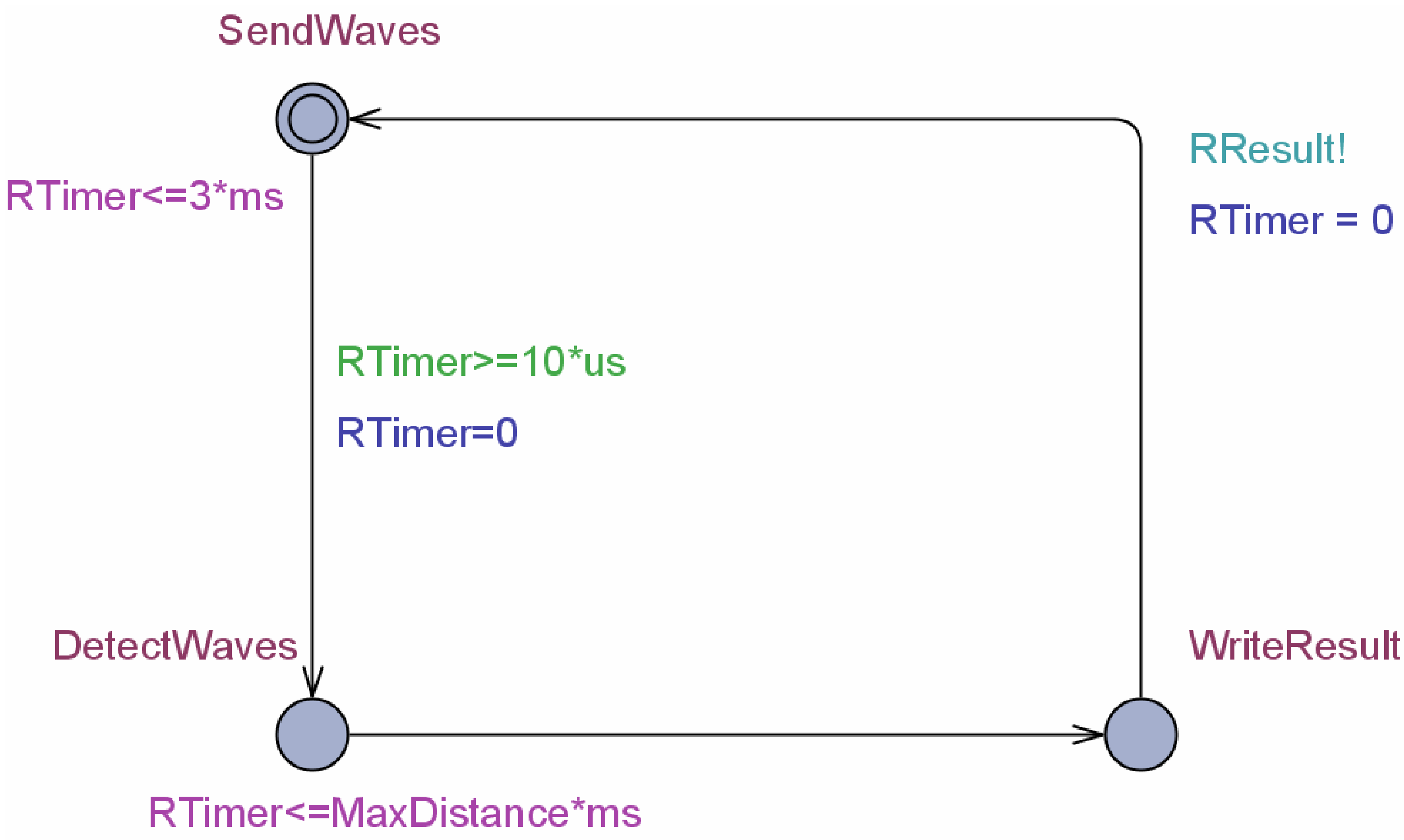
- Design errors detected in the first specification step
  - Analysis of counter example trace indicates reasons
  - Change of basic model design (e.g. priorities, deadlines, periods)
- Fixing design errors allows verification of all properties

→ Proceed to next refinement stage

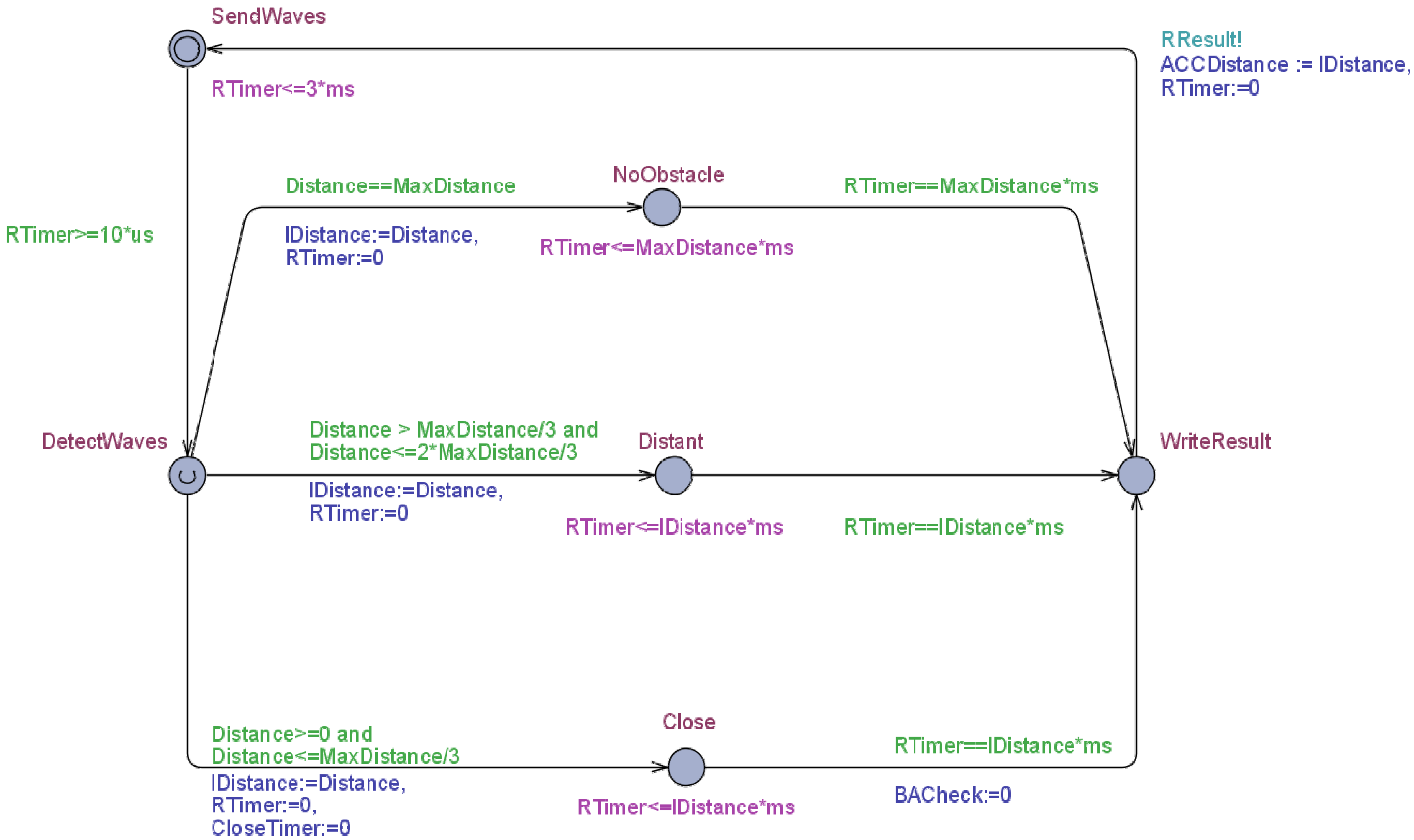
Overview

- Motivation
- Goals
- Introduction to Timed Automata
- Using Timed Automata in the Design Process
  - The Case Study System: An Emergency Brake Assistant
  - Properties to Be Verified
  - First Step: Basic Model Specification
  - Verifying Properties for the Overall System
  - Refinement Step
- Conclusion

Refinement Step



Refinement Step



Refinement Step

- UPPAAL fails in proving the same properties as for basic model
- Possible verification
  - Find and prove simulation relation on basic and refined model
    - Show that previously verified properties are still valid
  - Prove properties by (partially) using
    - Simulator model parts
    - Basic model parts
  - Use smaller refinement steps
  - Prove only local model part properties

Overview

- Motivation
- Goals
- Introduction to Timed Automata
- Using Timed Automata in the Design Process
- Conclusion

Conclusion

- Need for verification is increasing ✔
  - → Method for early stage use of formal methods presented
  - → Approach to verify large systems as far as possible
- The state explosion problem sets boundaries ✔
  - → Basic idea: using simple models at an early stage
  - → Refine these models at later stages
- Lack of real world examples ✔
  - → Demonstration of feasibility using automotive example

Bottom Line

Formal methods are useful in the design of large systems.

Thank you for your attention.