

# System Verification

Alexandra Silva

<sup>1</sup>CWI, The Netherlands

Program Correctness 2009

# Motivation

ICT (Information and Communication Technology) systems are everywhere and becoming more complex.

They are essential to the survival of companies so they have to be reliable

It is all about money and safety!

- **Intel** 475 million USD
- **Ariane-5** 36 seconds 1.1 million USD
- **Denver's airport** 9 months: 1.1 million USD per day
- **Airline** : 24h failure of online ticket reservation will provoke bankruptcy
- **Therac-25** 6 people died

# Motivation

ICT (Information and Communication Technology) systems are everywhere and becoming more complex.

They are essential to the survival of companies so they have to be reliable

It is all about money and safety!

- **Intel** 475 million USD
- **Ariane-5** 36 seconds 1.1 million USD
- **Denver's airport** 9 months: 1.1 million USD per day
- **Airline** : 24h failure of online ticket reservation will provoke bankruptcy
- **Therac-25** 6 people died

# Motivation

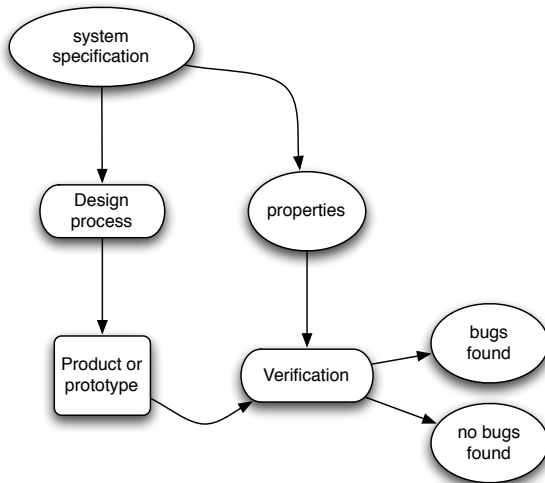
ICT (Information and Communication Technology) systems are everywhere and becoming more complex.

They are essential to the survival of companies so they have to be reliable

It is all about money and safety!

- **Intel** 475 million USD
- **Ariane-5** 36 seconds 1.1 million USD
- **Denver's airport** 9 months: 1.1 million USD per day
- **Airline** : 24h failure of online ticket reservation will provoke bankruptcy
- **Therac-25** 6 people died

# System Verification



**In this course:** Verification = Model Checking

# Alternate verification techniques

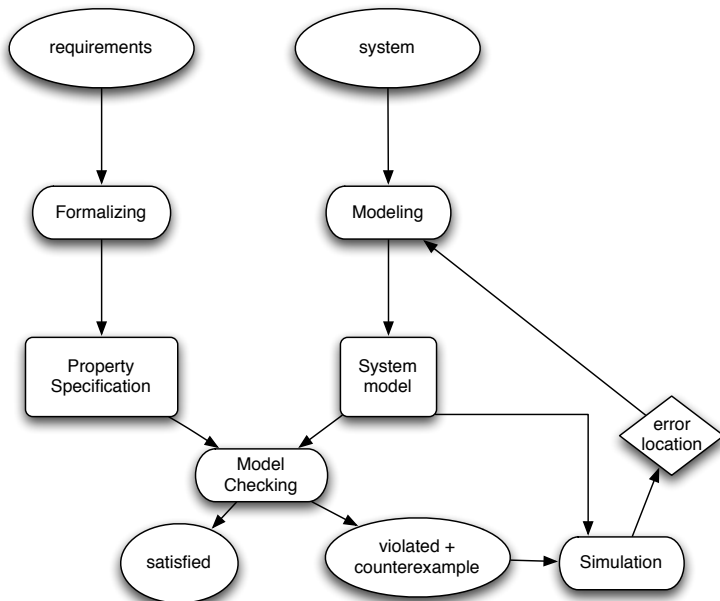
## Software Verification

- Peer Reviewing
- Testing
- Theorem Proving

## Hardware Verification

- Emulation
- Simulation
- Structural Analysis

# Model Checking Process



# Some successful cases

- IEEE standards
- Deep space 1 spacecraft controller (6 errors)
- Control of a storm surge barrier (Rotterdam)

Most of the errors arise from **concurrency**.

```
proc Inc = while true do if x < 200 then x := x + 1 fi od
proc Dec = while true do if x > 0 then x := x - 1 fi od
proc Reset = while true do if x = 200 then x := 0 fi od
```

Is  $x$  always between 0 and 200?

# Some successful cases

- IEEE standards
- Deep space 1 spacecraft controller (6 errors)
- Control of a storm surge barrier (Rotterdam)

Most of the errors arise from **concurrency**.

```
proc Inc = while true do if  $x < 200$  then  $x := x + 1$  fi od  
proc Dec = while true do if  $x > 0$  then  $x := x - 1$  fi od  
proc Reset = while true do if  $x = 200$  then  $x := 0$  fi od
```

Is  $x$  always between 0 and 200?

# Strengths of model checking

- General verification approach, applicable to a wide range of applications, s.a. **embedded systems**, **software engineering** and **hardware design**.
- It provides **diagnostic information** in case a property is invalidated: useful for **debugging**.
- It is a potential "push button" technology: it does not require a high degree of user interaction or expertise.
- Supports partial verification: focus on essential properties first.
- It is gaining popularity in industry : several hardware companies have verification labs (Intel, Philips, . . .).
- It can easily be integrated in existing development cycles.

# Weaknesses of model checking

- It is mainly appropriate for **control-intensive** applications and less suited for data-intensive applications.
- It verifies a **system model** and not the actual system. Thus, it needs to be complemented by testing.
- It checks only stated requirements: no completeness is guaranteed.
- State explosion : the model might be too big for the available memory.
- Its usage requires some expertise to find appropriate abstractions that lead to compact models.

*Model checking is an automated technique that, given a **finite-state model** of a system and a **formal property**, systematically checks whether this property holds for a given state in the model.*

- Lecture 2 Transition systems and LTL
- Lecture 3 LTL
- Lecture 4 CTL
- Lecture 5 and 6 Model checking algorithms

*Model checking is an automated technique that, given a **finite-state model** of a system and a **formal property**, systematically checks whether this property holds for a given state in the model.*

- [Lecture 2](#) Transition systems and LTL
- [Lecture 3](#) LTL
- [Lecture 4](#) CTL
- [Lecture 5 and 6](#) Model checking algorithms