The CYBERNETIX and TERMA Case Studies in μCRL

Yaroslav S. Usenko

University of Twente, The Netherlands

with the input from Holger Hermanns, Tomas Krilavičius and Theo Ruys

6 May 2003

MoDeST Framework

A Modeling and Description Language for Stochastic and Timed Systems.

MoDeST Framework

A Modeling and Description Language for Stochastic and Timed Systems.

- A Language Framework developed at the FMG group of Twente University.

MoDeST Framework

A Modeling and Description Language for Stochastic and Timed Systems.

- A Language Framework developed at the FMG group of Twente University.
- A general specification meta-language for describing the behavior of discrete event systems.

MoDeST Framework

A Modeling and Description Language for Stochastic and Timed Systems.

- A Language Framework developed at the FMG group of Twente University.
- A general specification meta-language for describing the behavior of discrete event systems.
- Will include primitives for describing timed, probabilistic, stochastic and hybrid systems.

MoDeST Framework

A Modeling and Description Language for Stochastic and Timed Systems.

- A Language Framework developed at the FMG group of Twente University.
- A general specification meta-language for describing the behavior of discrete event systems.
- Will include primitives for describing timed, probabilistic, stochastic and hybrid systems.
- Translations to the well-known formalisms like UPPAAL, SPIN, CADP, MÖBIUS, μCRL, etc., should enable efficient analysis.

The CYBERNETICS Case Study

- Has been provided by CYBERNETIX (France).

The CYBERNETICS Case Study

- Has been provided by Cybernetix (France).
- Deals with a card personalization conveyor.

The CYBERNETICS Case Study

- Has been provided by CYBERNETIX (France).
- Deals with a card personalization conveyor.
- The goal is in finding an optimal schedule.

The CYBERNETICS Case Study

- Has been provided by Cybernetix (France).
- Deals with a card personalization conveyor.
- The goal is in finding an optimal schedule.
- An idea is in trying model checking techniques for timed systems.

Some Approaches to the CYBERNETICS CS (related to MoDeST)

- An SMV model by Biniam Gebremichael (KUN, The Netherlands).

Some Approaches to the CYBERNETICS CS (related to MoDeST)

- An SMV model by Biniam Gebremichael (KUN, The Netherlands).
- An UPPAAL model by Tomas Krilavičius (UT, The Netherlands).

Some Approaches to the CYBERNETICS CS (related to MoDeST)

- An SMV model by Biniam Gebremichael (KUN, The Netherlands).
- An UPPAAL model by Tomas Krilavičius (UT, The Netherlands).
- A similar model in SPIN by Theo Ruys (UT, The Netherlands).

Some Approaches to the CYBERNETICS CS (related to MoDeST)

- An SMV model by Biniam Gebremichael (KUN, The Netherlands).
- An UPPAAL model by Tomas Krilavičius (UT, The Netherlands).
- A similar model in SPIN by Theo Ruys (UT, The Netherlands).
- Two μCRL models, similar to the models of Biniam and Tomas, respectively.

What does similar mean?

Why are these models related to MoDeST?

μCRL Language

μCRL is based on process algebra and algebraic (equational) data types.

Specification structure:

- data types definitions (**sort**, **func**, **map**, **rew**)

μCRL Language

μCRL is based on process algebra and algebraic (equational) data types.

Specification structure:

- data types definitions (**sort**, **func**, **map**, **rew**)
- actions and communication functions definitions (**act**, **comm**)

μCRL Language

μCRL is based on process algebra and algebraic (equational) data types.

Specification structure:

- data types definitions (**sort**, **func**, **map**, **rew**)
- actions and communication functions definitions (**act**, **comm**)
- process definitions (**proc**): equations involving:
  $p ::= \mathsf{a}(\overrightarrow{t}) \mid \delta \mid \mathsf{Y}(\overrightarrow{t}) \mid p + p \mid p \cdot p \mid p \,\|\, p \mid \sum_{d:D} p \mid p \lhd c \rhd p \mid \tau_I(p) \mid \partial_H(p) \mid \rho_R(p)$

μCRL Language

μCRL is based on process algebra and algebraic (equational) data types.

Specification structure:

- data types definitions (**sort**, **func**, **map**, **rew**)
- actions and communication functions definitions (**act**, **comm**)
- process definitions (**proc**): equations involving:
  $p ::= \mathsf{a}(\overrightarrow{t}) \mid \delta \mid \mathsf{Y}(\overrightarrow{t}) \mid p + p \mid p \cdot p \mid p \parallel p \mid \sum_{d:D} p \mid p \lhd c \rhd p \mid \tau_I(p) \mid \partial_H(p) \mid \rho_R(p)$
- initial state (**init**).

μCRL Language

μCRL is based on process algebra and algebraic (equational) data types.

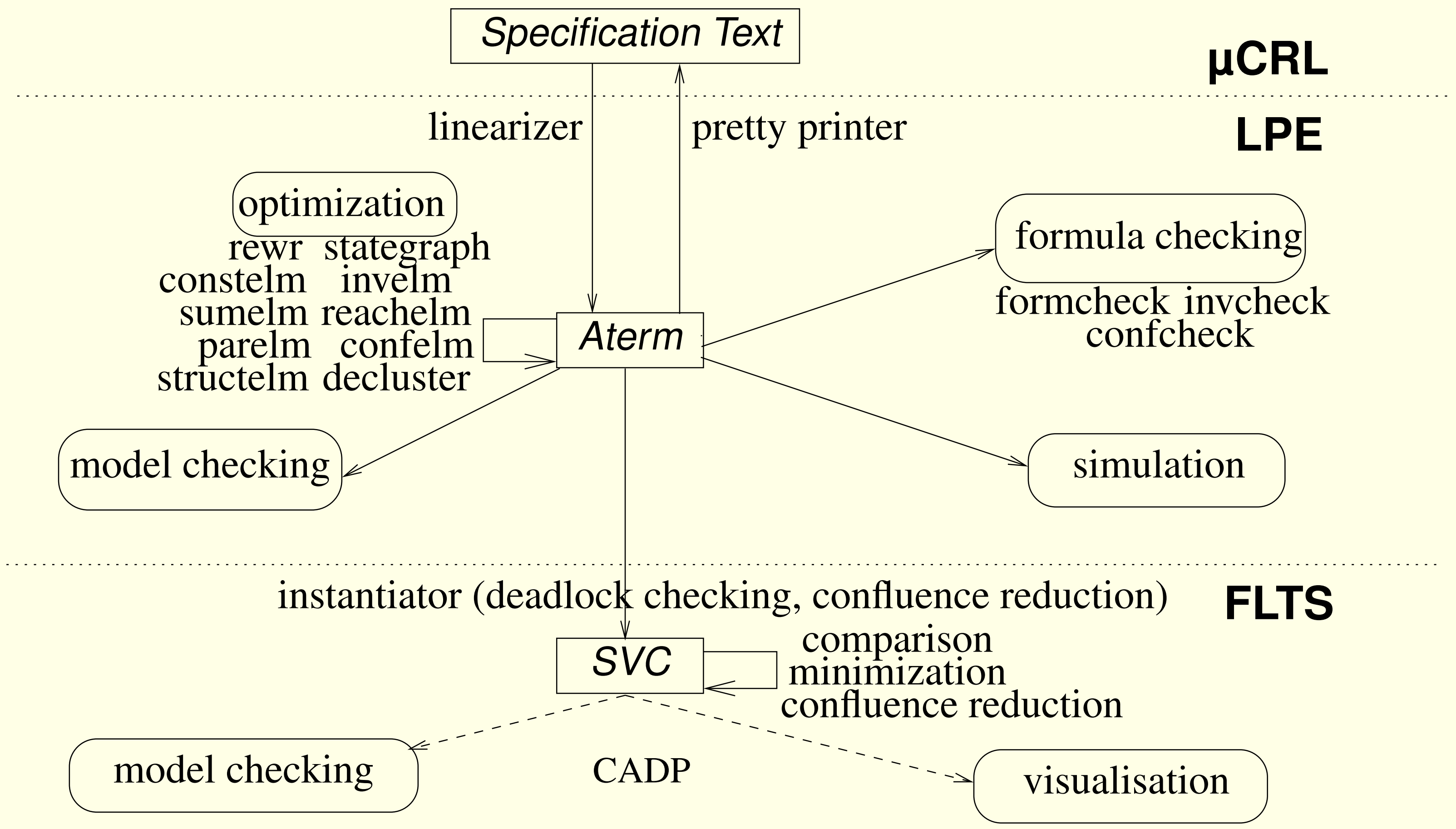
Specification structure:

- data types definitions (**sort**, **func**, **map**, **rew**)
- actions and communication functions definitions (**act**, **comm**)
- process definitions (**proc**): equations involving:
  $p ::= \mathsf{a}(\overrightarrow{t}) \mid \delta \mid \mathsf{Y}(\overrightarrow{t}) \mid p+p \mid p\cdot p \mid p \,\|\, p \mid \sum_{d:D} p \mid p \lhd c \rhd p \mid \tau_I(p) \mid \partial_H(p) \mid \rho_R(p)$
- initial state (**init**).

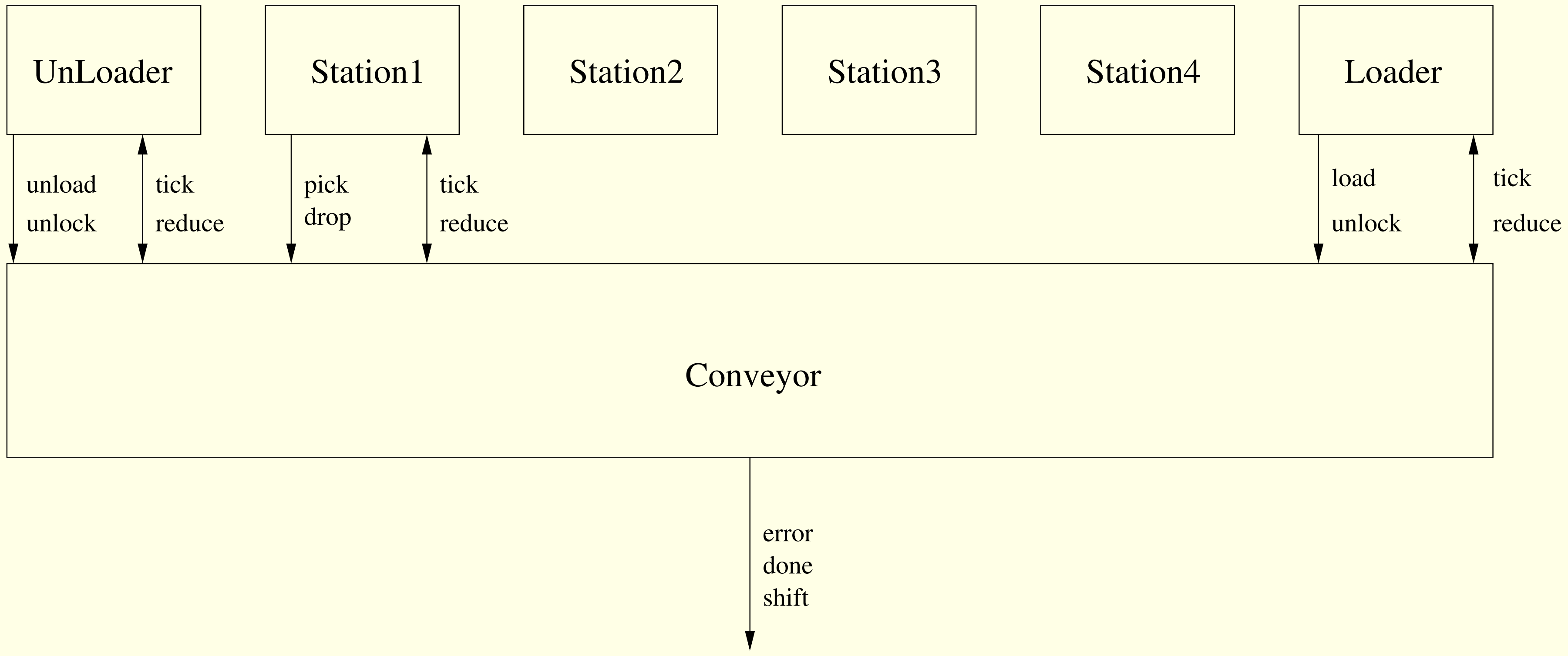
Extensions to process algebra:

- action parameterized by data ($\mathsf{a}(d) \mid \mathsf{b}(e) \approx \mathsf{c}(d) \lhd d = e \rhd \delta$),
- $\sum_{d:D} p$ and $x \lhd c \rhd y$
- systems of parameterized recursion equations.

Overview of the μCRL Toolset



The CYBERNETIX CS in μCRL



<u>What are the differences between the models of Beniam and Tomas?</u>

Modeling Time in μCRL

- The passage of time is modeled by an action tick (one time unit has passed).

Modeling Time in μCRL

- The passage of time is modeled by an action tick (one time unit has passed).
- All processes in the system synchronize to this action by means of multi-party communication.

Modeling Time in μCRL

- The passage of time is modeled by an action tick (one time unit has passed).
- All processes in the system synchronize to this action by means of multi-party communication.
- Technically, this is achieved by using action renaming and synchronous communication.

Analysis of the μCRL Models

- Both models have been analyzed by means of performing a breadth-first search for a deadlock. (a shortest path to a deadlock is an optimal schedule)

Analysis of the μCRL Models

- Both models have been analyzed by means of performing a breadth-first search for a deadlock. (a shortest path to a deadlock is an optimal schedule)
- For the model similar to Beniam's we could analyze the system with 4 personalization station and 6 card. No comparison of the resulting schedules (October 2002).

Analysis of the μCRL Models

- Both models have been analyzed by means of performing a breadth-first search for a deadlock. (a shortest path to a deadlock is an optimal schedule)
- For the model similar to Beniam's we could analyze the system with 4 personalization station and 6 card. No comparison of the resulting schedules (October 2002).
- For the model similar to Tomas' we could analyze all the systems presented by Tomas and Theo on the previous AMETIST meetings in Twente and Dortmund, respectively.

Analysis of the μCRL Models

- Both models have been analyzed by means of performing a breadth-first search for a deadlock. (a shortest path to a deadlock is an optimal schedule)
- For the model similar to Beniam's we could analyze the system with 4 personalization station and 6 card. No comparison of the resulting schedules (October 2002).
- For the model similar to Tomas' we could analyze all the systems presented by Tomas and Theo on the previous AMETIST meetings in Twente and Dortmund, respectively.
- For the case of 4 stations and 8 cards we could find an optimal schedule which is 1 tick better than the one found by Theo (46 instead of 47 ticks).

Finding an Optimal Loop

- By an ad-hoc analysis of the problem it became apparent that the system repeats itself after processing a number of cards.

Finding an Optimal Loop

- By an ad-hoc analysis of the problem it became apparent that the system repeats itself after processing a number of cards.
- The only difference is that all card numbers in the system are bigger by 1, and the current time value is also bigger.

Finding an Optimal Loop

- By an ad-hoc analysis of the problem it became apparent that the system repeats itself after processing a number of cards.
- The only difference is that all card numbers in the system are bigger by 1, and the current time value is also bigger.
- In the model we do not store the global time.

Finding an Optimal Loop

- By an ad-hoc analysis of the problem it became apparent that the system repeats itself after processing a number of cards.
- The only difference is that all card numbers in the system are bigger by 1, and the current time value is also bigger.
- In the model we do not store the global time.
- After each loaded card we decrease each card number in the system (on the conveyor, in the stations, the next card to be loaded) by 1. This is done by synchronizing all processes on reduce action.

Finding an Optimal Loop (cont.)

- We generated a Labeled Transition System (LTS) containing all behaviors of the model with 4 cards. It has 9.9M states and 17.5M transitions. (Generated in approx 1h on an Athlon 1.4GHz machine with 2Gb of RAM).

Finding an Optimal Loop (cont.)

- We generated a Labeled Transition System (LTS) containing all behaviors of the model with 4 cards. It has 9.9M states and 17.5M transitions. (Generated in approx 1h on an Athlon 1.4GHz machine with 2Gb of RAM).
- Finding an optimal (maximal number of load actions per one tick action) loop that starts and ends with a load action in this LTS, and finding an optimal path to it, could give us optimal schedules for any card number.

Finding an Optimal Loop (cont.)

- We generated a Labeled Transition System (LTS) containing all behaviors of the model with 4 cards. It has 9.9M states and 17.5M transitions. (Generated in approx 1h on an Athlon 1.4GHz machine with 2Gb of RAM).
- Finding an optimal (maximal number of load actions per one tick action) loop that starts and ends with a load action in this LTS, and finding an optimal path to it, could give us optimal schedules for any card number.
- The future plans are finding these optimal loops and trying to analyze the model with 5 or more stations in this way.

Conclusions on the CYBERNETICS CS

- The general formulation of the case study is hard to tackle by the existing automated model checking techniques. Therefore it is not feasible to come up with an industrial tool that would solve such a problem out of the box.

Conclusions on the CYBERNETICS CS

- The general formulation of the case study is hard to tackle by the existing automated model checking techniques. Therefore it is not feasible to come up with an industrial tool that would solve such a problem out of the box.
- Fixing a particular concrete formulation of the case study allows the academic partners to compare the tools and techniques by trying to find concrete optimal schedules.

TERMA CS Results

- The TERMA CS has been provided by Terma AS (Denmark).

TERMA CS Results

- The TERMA CS has been provided by Terma AS (Denmark).
- Has to do with a scheduling of buffer flushes that does not lead to an overflow/underflow.

TERMA CS Results

- The TERMA CS has been provided by Terma AS (Denmark).
- Has to do with a scheduling of buffer flushes that does not lead to an overflow/underflow.

- The proposed schedule is: 1,2,3,4,7,8,9; 5,6,4,7,8,9. . .

TERMA CS Results

- The TERMA CS has been provided by Terma AS (Denmark).
- Has to do with a scheduling of buffer flushes that does not lead to an overflow/underflow.

- The proposed schedule is: 1,2,3,4,7,8,9; 5,6,4,7,8,9. . .
- The buffer sizes that are sufficient for this schedule are: 69 bytes for a single buffers, and 98 bytes for double buffers.

TERMA CS Results

- The TERMA CS has been provided by Terma AS (Denmark).
- Has to do with a scheduling of buffer flushes that does not lead to an overflow/underflow.

- The proposed schedule is: 1,2,3,4,7,8,9; 5,6,4,7,8,9. . .
- The buffer sizes that are sufficient for this schedule are: 69 bytes for a single buffers, and 98 bytes for double buffers.
- The verification of this schedule took 6.5 minutes on a 1.7GHz Pentium IV with 256 Mb of RAM. (534377 states).

TERMA CS Results

- The TERMA CS has been provided by Terma AS (Denmark).
- Has to do with a scheduling of buffer flushes that does not lead to an overflow/underflow.

- The proposed schedule is: 1,2,3,4,7,8,9; 5,6,4,7,8,9. . .
- The buffer sizes that are sufficient for this schedule are: 69 bytes for a single buffers, and 98 bytes for double buffers.
- The verification of this schedule took 6.5 minutes on a 1.7GHz Pentium IV with 256 Mb of RAM. (534377 states).