Modeling and Analyzing Real-Time Systems in Rewriting Logic (II)

Peter Ölveczky

University of Oslo

May 4, 2010

Based on joint work with Kyungmin Bae, Artur Boronat, Marco Caccamo, José Meseguer, Stian Thorvaldsen, and others

Context

Applications

"Concrete" Systems

Semantic Framework for Real-Time Systems

Formal Analysis for MDE Languages

Lecture Series

Last Thursday: Real-Time Maude tutorial

Today: Context and applications

Context

Applications

"Concrete" Systems

Semantic Framework for Real-Time Systems

Formal Analysis for MDE Languages

Content

Context

What is it Good For?

- "Concrete" Systems
- Semantic Framework for Real-Time Systems
- Formal Analysis for MDE Languages

Summary

Context

Applications

"Concrete" Systems

Semantic Framework for Real-Time Systems

Formal Analysis for MDE Languages

Formal Analysis: Model Checking vs Theorem Proving

Model Checking

- exhaustive analysis from single initial states
- automatic
- search for subtle bugs
- cannot prove correctness

Theorem Proving

- (inductive) analysis w.r.t. all initial states
- non-trivial user interaction
- prove correctness
- typically cannot find bugs

Formal Tools: Expressiveness vs Analytic Power

Expressive formalism

- can model large and advanced systems
- formal analysis harder
- properties in general undecidable

Restricted formalism

- hard/impossible to model large and complex systems
- more optimized analysis
- properties decidable

Formal Languages and Tools for Real-Time Systems

- Timed automata
  - finite automata extended with clocks
  - restricted transition guards
  - key properties decidable
  - tools: UPPAAL, Kronos, RED
- Timed Petri nets
  - limited tool support
- Timed process algebras
- IF, TE-LOTOS, etc:
  - separate formalisms for data types, dynamic behavior, and time
  - unclear or non-existing semantics
  - based on fixed communication primitives
- MOBY/RT
  - designs specified as PLC-automata
  - translated into timed automata for model checking

Real-Time Maude in Comparison

- Expressiveness and ease of specification
  - define any computable data type
  - dynamic object creation/deletion
  - unbounded data structures
  - different forms of communication
- Simple and intuitive formalism
- "Natural" model for real-time systems
  - object-oriented real-time systems
- Properties in general undecidable

The Main Question

Are there systems where Real-Time Maude's expressiveness is needed, and where Real-Time Maude analysis yields interesting results?

Classes of Applications

- "Concrete" systems/protocols
- Semantic framework for real-time systems
- Formal analysis tool for other languages

AER/NCA

AER/NCA :

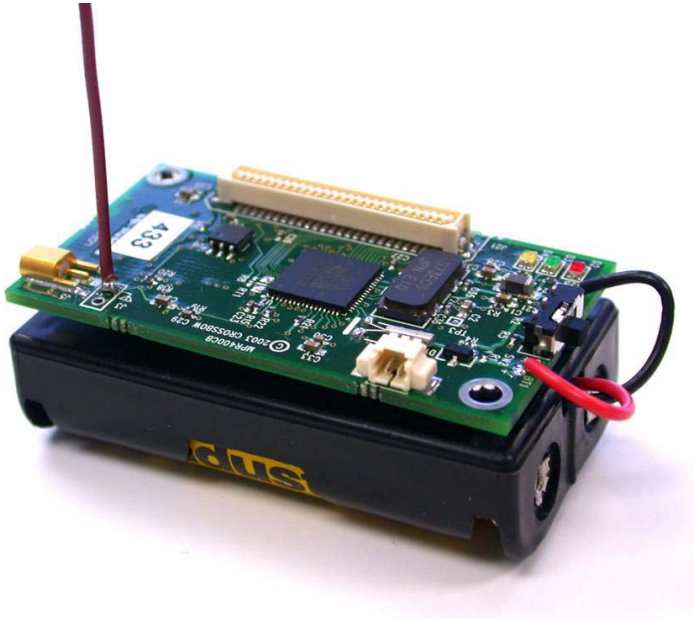
- Sophisticated multicast protocol for active networks
  - active routers buffer and repair packets
  - 50 pages of use cases
  - involves link capacity and propagation delay, packet sizes, etc.
- Easy to experiment with different sending rates, link capacities/speeds, ...
- Real-Time Maude analysis found all known design errors
- . . . and additional unknown serious design errors

CASH Scheduling Algorithm

CASH : State-of-the-art scheduling algorithm

- A job can use more or less time than allocated
- Unused execution times put in a queue for reuse
- Simulation: # elements in queue unbounded
- Modeled all possible jobs
  - analyze all possible scenarios for a given set of servers
- Search found missed hard deadline
- Extensive "Monte-Carlo simulation" did not find flaw

Density Control in Wireless Sensor Networks



- Wireless sensor network:
  - network of battery-powered nodes with sensing capabilities
  - (low-power) radio communication
  - monitor an area
- Density control:
  - extend lifetime: add redundancy
  - nodes periodically turned off to save power
    - sensing area covered by active nodes

OGDC Wireless Sensor Network Algorithm

OGDC: state-of-the-art density control algorithm

- Idea: select "best placed" node to become active next
  - minimize total overlap
- Simulated by developers using ns-2 with wireless extension
  - OGDC outperforms other algorithms

The Algorithm

Periodically:

- Some nodes become active
  - broadcast
- "Best placed" node should become active next
  - the better the location, the smaller the back-off timer
- Back-off timer expires: node becomes active and broadcasts
- Sensing area covered by active nodes: sleep!

Formal Methods Challenges

- New form of communication: radio transmission
  - broadcast to nodes within transmission range
  - OGDC definition uses transmission delay
- Time (power, timers, delays, ...)
- Geometric entities (areas, angles, distances, ...)
- Probabilistic behaviors
- Analyze performance and correctness

TTBOMK: First formal analysis of advanced WSN algorithm

Real-Time Maude Specification of OGDC (I)

- Easy to define communication model (5 equations)
- Probabilistic behaviors not supported by
  - simulation: "sample" values using pseudo-random function

Real-Time Maude Specification of OGDC (II)

Locations in a two-dimensional plane: $x \,.\, y$

```
sort Location .
op _._ : Rat Rat -> Location [ctor] .
```

Sensor nodes in OGDC:

```
class WSNode | backoffTimer : TimeInf,
               coverageArea : Bitmap,
               remainingPower : Nat,
               roundTimer : TimeInf,  ...
```

Node identified by its location:

```
subsort Location < Oid .
```

Modeling Communication in WSN

Broadcast to all nodes within transmission range

- Idea: node "sends" a `broadCast` message
- A `broadCast` message is "dissolved" into a single message to each node within range
- Message collisions not modeled

Modeling Broadcast

```
sort MsgCont .    --- Message content
msg broadcast_from_ : MsgCont Location -> Msg .
msg msg_from_to_ : MsgCont Location Location -> Msg .
```

The following equation defines the desired equivalence:

```
var C : Configuration .  var MC : MsgCont .
eq {< L : WSNode | >  (broadcast MC from L)  C} =
   {< L : WSNode | >  distributeMsg(L, MC, C)} .
```

Modeling Broadcast (cont.)

```
op distributeMsg : Location MsgCont Configuration
                   -> Configuration [frozen (3)] .

var MSG : Msg .  var O : Oid .
eq distributeMsg(L, MC, none) = none .

eq distributeMsg(L, MC, MSG C) =
    MSG distributeMsg(L, MC, C) .

eq distributeMsg(L, MC, < L' : WSNode | >  C) =
     < L' : WSNode | >  distributeMsg(L, MC, C)
     if L withinTransRangeOf L'
     then dly(msg MC from L to L', f(L, L')) else none fi .
```

Rewrite Rules

OGDC has 11 rewrite rules. One is

```
rl [startingNodePowerOn] :
   < L : WSNode | remainingPower : P, backoffTimer : 0,
                  hasVolunteered : true >
   < Random : RandomNGen | seed : M >
  =>
   < L : WSNode | remainingPower : P monus transPower,
                  backoffTimer : INF, status : on >
   < Random : RandomNGen | seed : random(M) >
   broadcast (powerOnWithDirection randomDirection(M)) from L .
```

Rewrite Rules (cont.)

Another rule is:

```
crl [recPowerOn1] :
    (msg (powerOnWithDirection D) from L' to L)
    < L : WSNode | status : undecided, neighbors : NBS,
                   bitmap : BM >
  =>
    < L : WSNode | status : off,
                   neighbors : NBS (L' starting (D >= 0)),
                   bitmap : updateBitmap(L, BM, L'),
                   backoffTimer : INF >
    if (L withinTwiceTheSensingRangeOf L')
       /\ coverageAreaCovered(updateBitmap(L, BM, L')) .
```

Real-Time Maude Simulation of OGDC

- Pseudo-random placements of nodes
  - different initial seeds give different node placements
- Measure # active nodes and coverage percentage throughout network lifetime
- Parameter values as in OGDC paper
- Simulation results:
  - ns-2 simulations: 17 to 19 active nodes for 200 – 600 nodes
  - Real-Time Maude simulations:

Total number of nodes	200	400	600
# active nodes for seed=1	32	38	45
# active nodes for seed=5	38	47	51
# active nodes for seed=97	26	46	58
# active nodes for seed=313	38	54	60
# active nodes for seed=341	37	39	61

Real-Time Maude Simulations of OGDC (II)

- But . . . ns-2 simulations ignored transmission delays
- Delays important in definition of OGDC
- Real-Time Maude simulations without delays:

Total number of nodes	200	400	600
# active nodes for seed=1	21	25	22
# active nodes for seed=5	19	20	24
# active nodes for seed=97	19	23	25
# active nodes for seed=313	21	23	21
# active nodes for seed=341	23	20	18

Flaw in OGDC

- Two nodes close/on top of each other:
  - "tie-breaker" difference in back-off timers
  - difference less than transmission delay!
- Simulation without transmission delay:
  - newly active node broadcasts and twin turns off!
- Simulation with transmission delay:
  - newly active node broadcasts, but twin turns on before message arrives!

OGDC Conclusion

Real-Time Maude simulations provide realistic performance estimates for advanced WSN algorithm

Some Other "Concrete" Applications

- LMST topology control protocol for wireless sensor networks (Katelman, Meseguer, and Hou, UIUC)
  - includes MAC contention
  - VeStA statistical model checking
- Found several bugs in embedded car software used by major car makers (Japan)
  - bugs not found by model-checking tools used in industry
- Power management algorithms for multimedia systems
- Parts of NORM multicast protocol developed by IETF

Semantic Framework for Real-Time Systems

- Real-Time Maude suitable semantic framework for real-time formalisms and languages
  - expressiveness
  - natural timed model
- Formal semantics for languages and notations
  - timed extension of the actor model
  - timed model transformations (e-Motions, MOMENT-2)
  - MDE languages for embedded systems

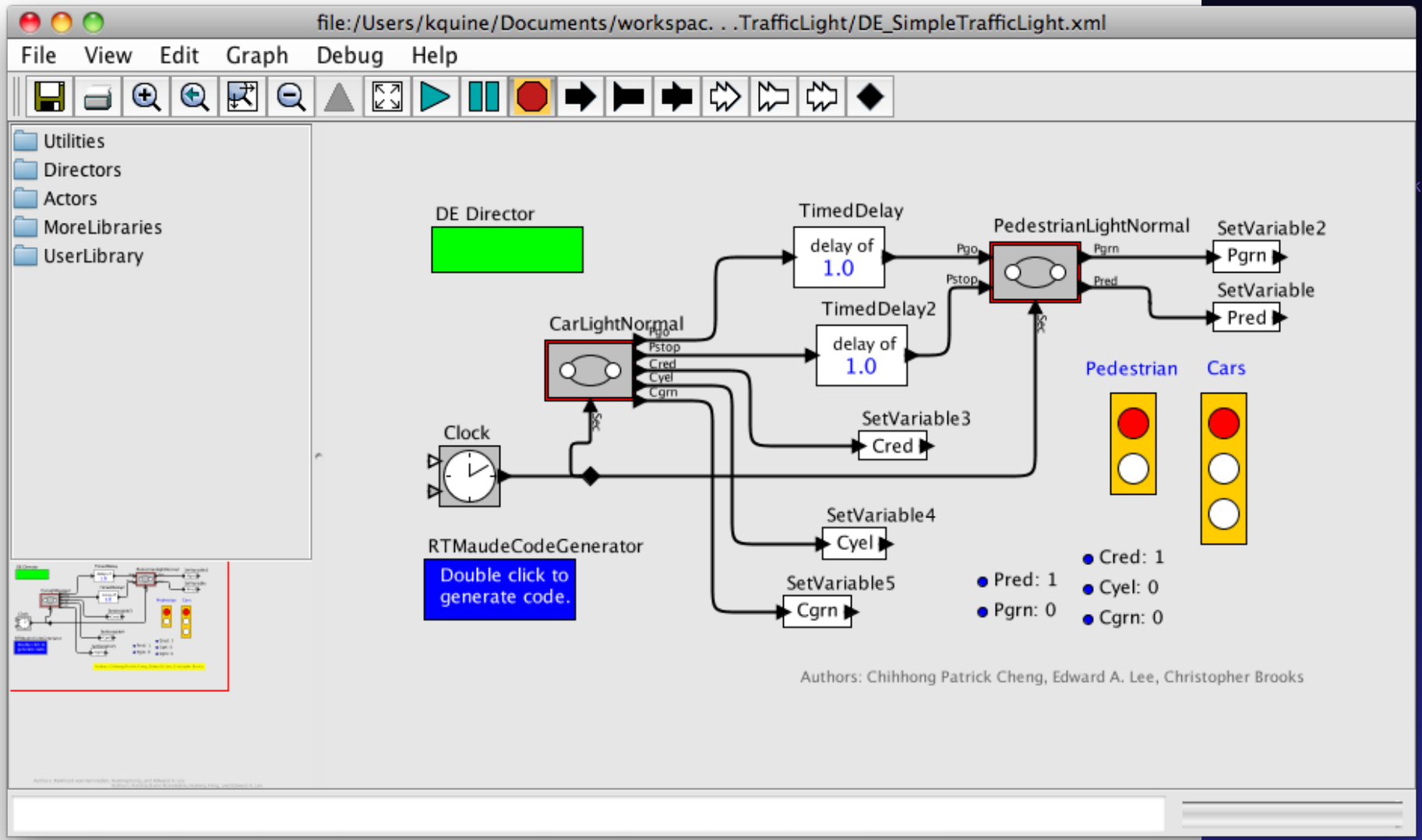
Model-driven Development

- MDE langauges for embedded systems:
  - AADL avionics standard
  - Ptolemy II
  - UML extensions
- Intuitive graphical modeling language
- Support for code generation
- Need for precise semantics and formal analysis

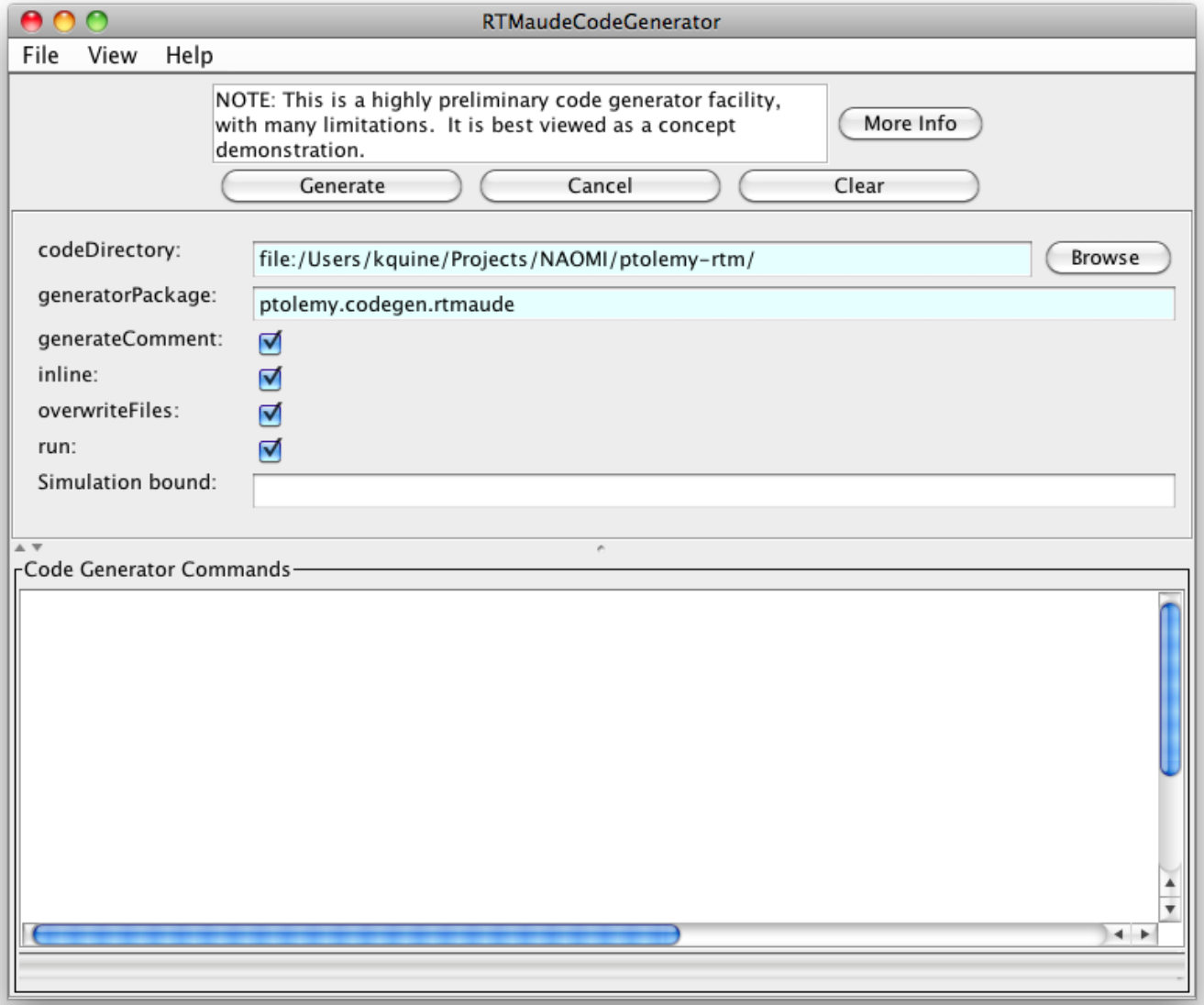
Enriching MDE Tools with Formal Analysis

- Define Real-Time Maude semantics of MDE language
  - Real-Time Maude's expressiveness crucial
- Use code gen infrastructure to synthesize Real-Time Maude spec from MDE model
- Define atomic propositions so that user can specify LTL properties without understanding Real-Time Maude

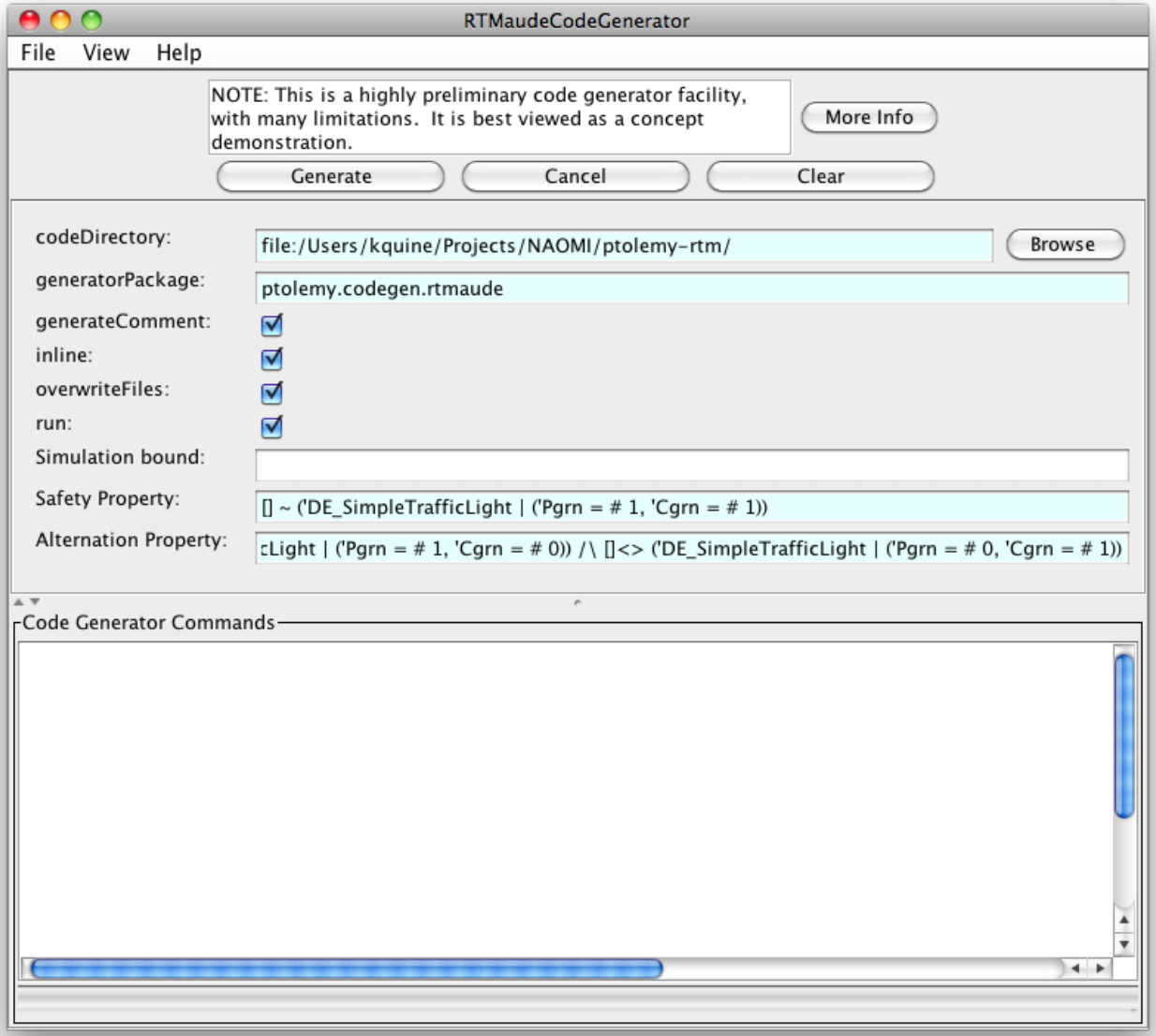
Analyzing Ptolemy II Models from within Ptolemy



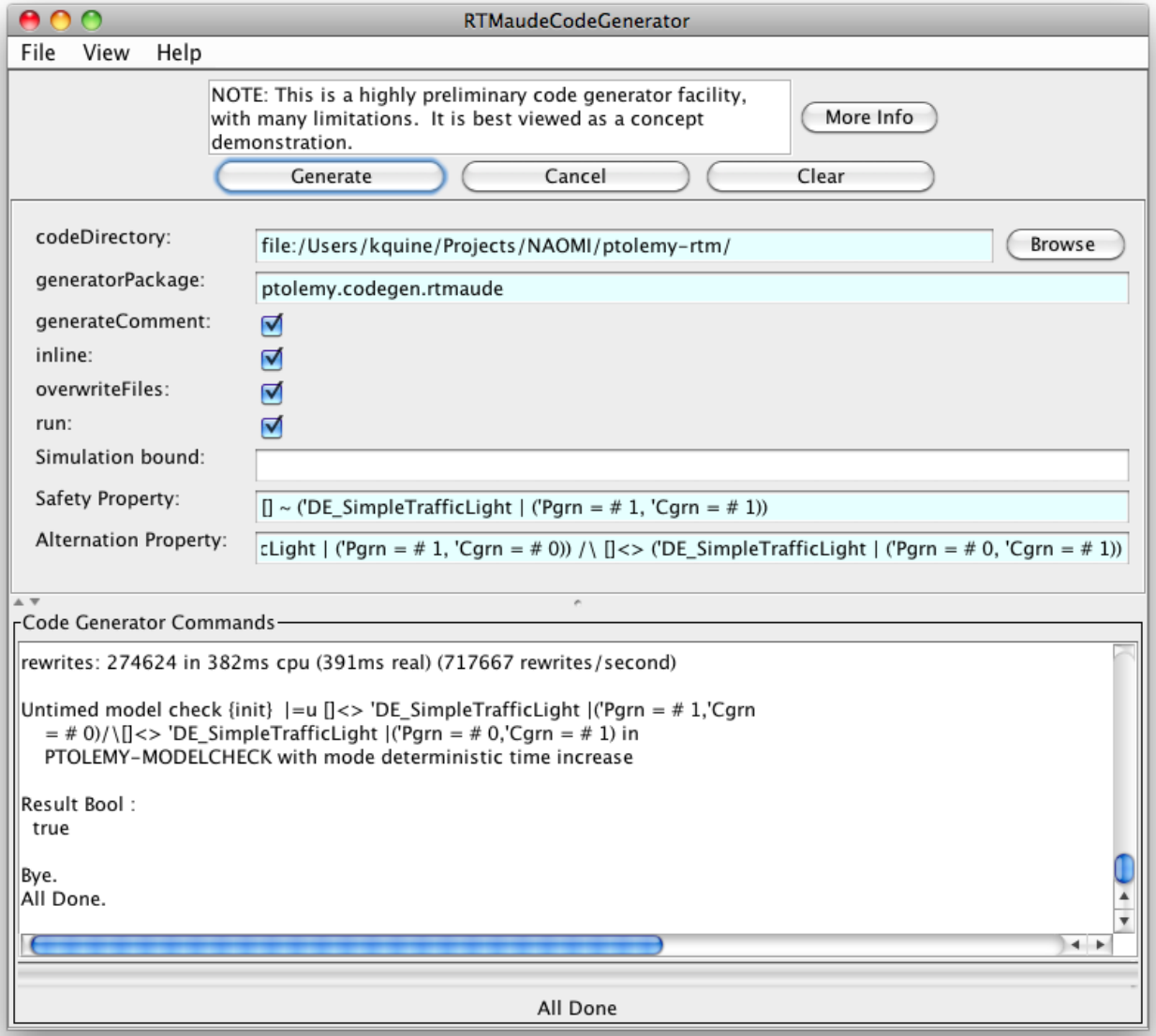
Analyzing Ptolemy II Models from within Ptolemy



Analyzing Ptolemy II Models from within Ptolemy



Analyzing Ptolemy II Models from within Ptolemy



Context
Applications
"Concrete" Systems
Semantic Framework for Real-Time Systems
Formal Analysis for MDE Languages

Some Other Formal Analysis Applications

- Part of AADL avionics standard
- Handset software analysis (DoCoMo Labs)
- Real-time model transformations in MOMENT-2 and e-Motions
- Patterns for real-time systems:
  - safe operation of medical devices
  - PALS: physically asynchronous, logically synchronous

Summary

- Real-Time Maude flexible, expressive, and intuitive
- Used on state-of-the-art systems in different domains
  - value added to domain-specific analysis
- Useful both as simulation tool and model checker
  - search found error in CASH
  - simulation gives better performance estimates for OGDC than simulation tool used by WSN community
- Semantics and formal analysis capabilities for real-time MDE languages and tools
- Completeness for new classes of dense time systems