Translating the Feature Oriented Requirements Modelling Language (FORML) to Alloy

For the purpose of automated analysis

David Dietrich
CS846

Outline

- Problem and Motivation
- FORML
- The Translator
- The Analysis
- Demo
- Limitations
- Future Work
- Conclusions

Problem

- One kind of correctness of a state machine is a lack of conflicts when concurrent transitions are executed
- A conflict occurs when 2 transitions execute at the same time, but one of their post-conditions does not hold
  - e.g., Changing the same object with 2 transitions may lead to a non-deterministic post state
- The goal of this project is to create a translator to Alloy that can use the Alloy Analyzer to locate conflicts when 2 transitions execute concurrently

The Cost of Requirements Errors

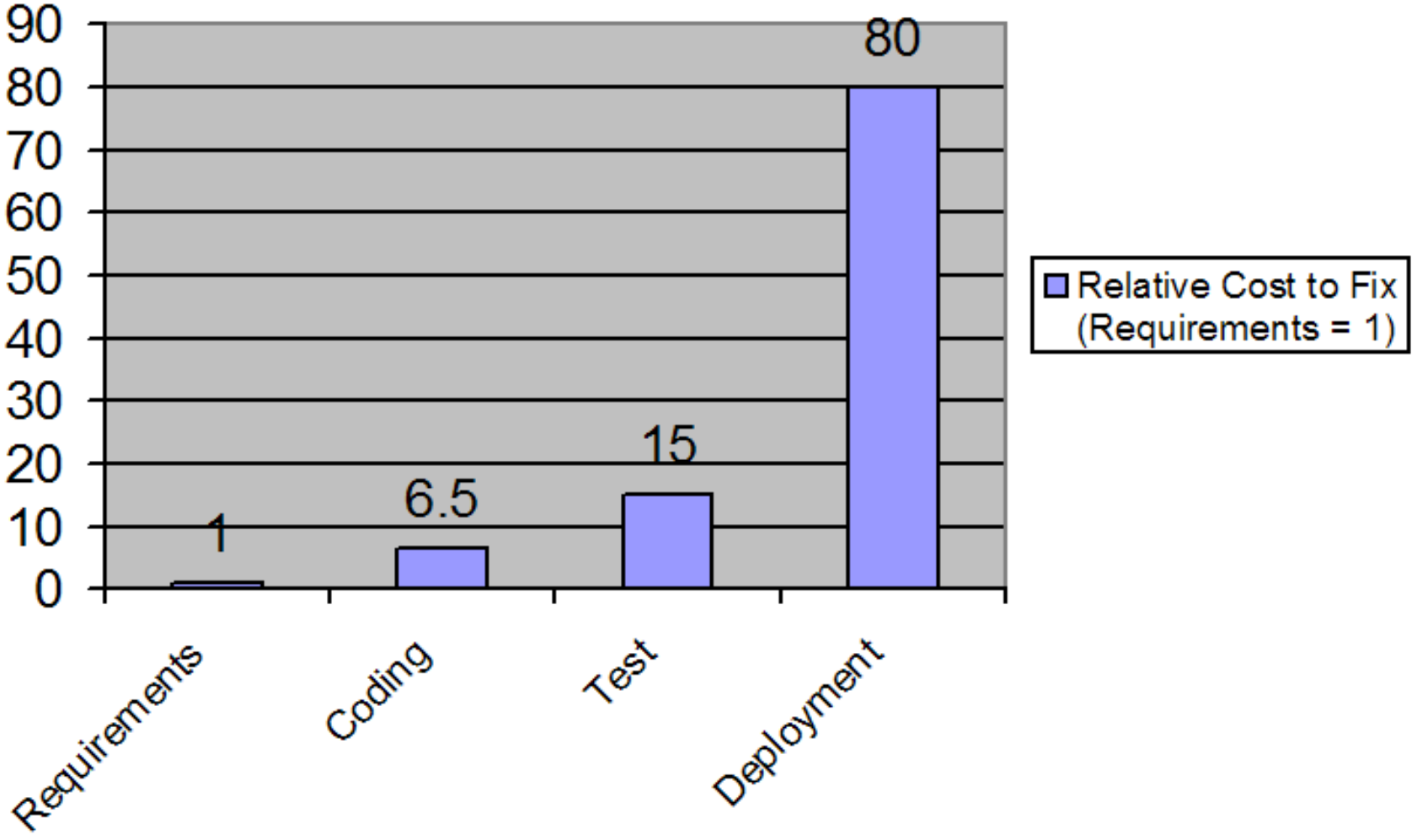
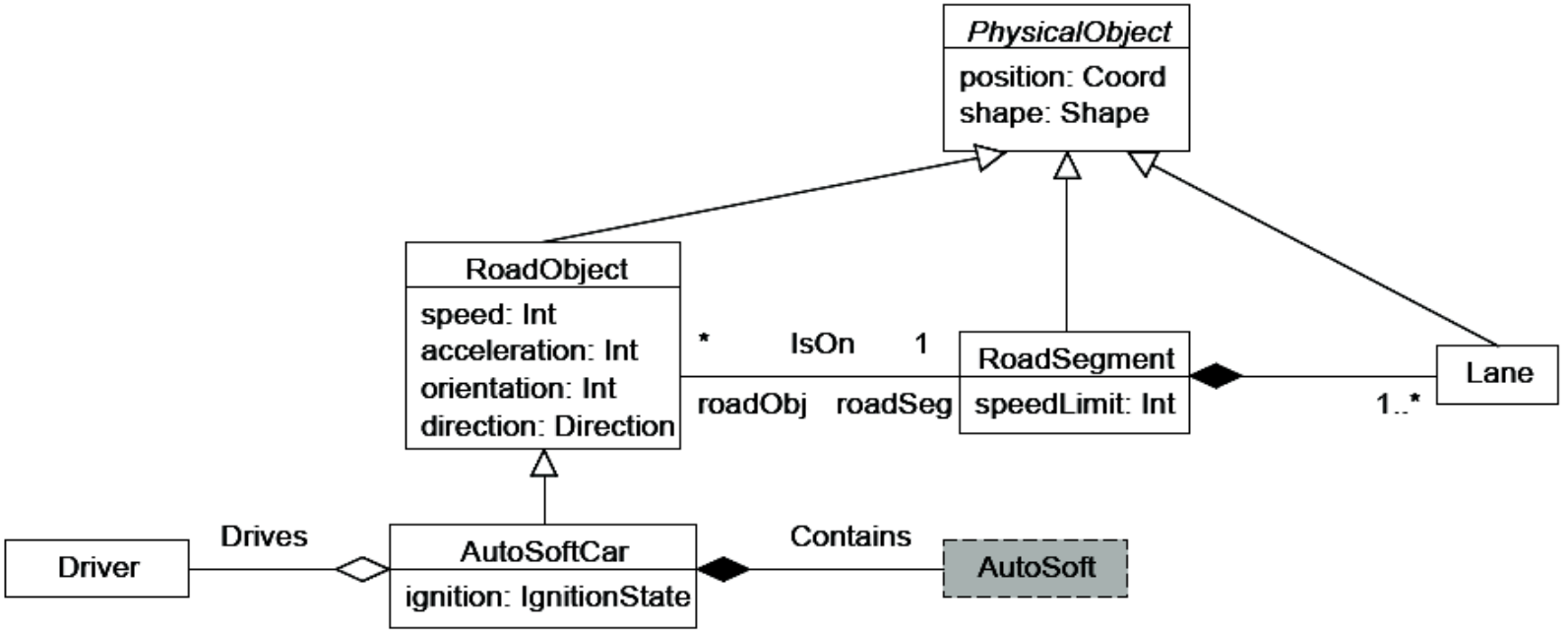


Image from http://www.stickyminds.com/sitewide.asp?ObjectId=12529&Function=edetail&ObjectType=ARTCOL

FORML

- Created by Pourya Shaker at the University of Waterloo
- Requirements modelling language that provides support for Software Product Lines and feature-oriented modelling
- FORML is designed as a graphical language, but a plain text grammar has also been created
- There are 2 models: the World Model, and the Behaviour Model

FORML – World Model

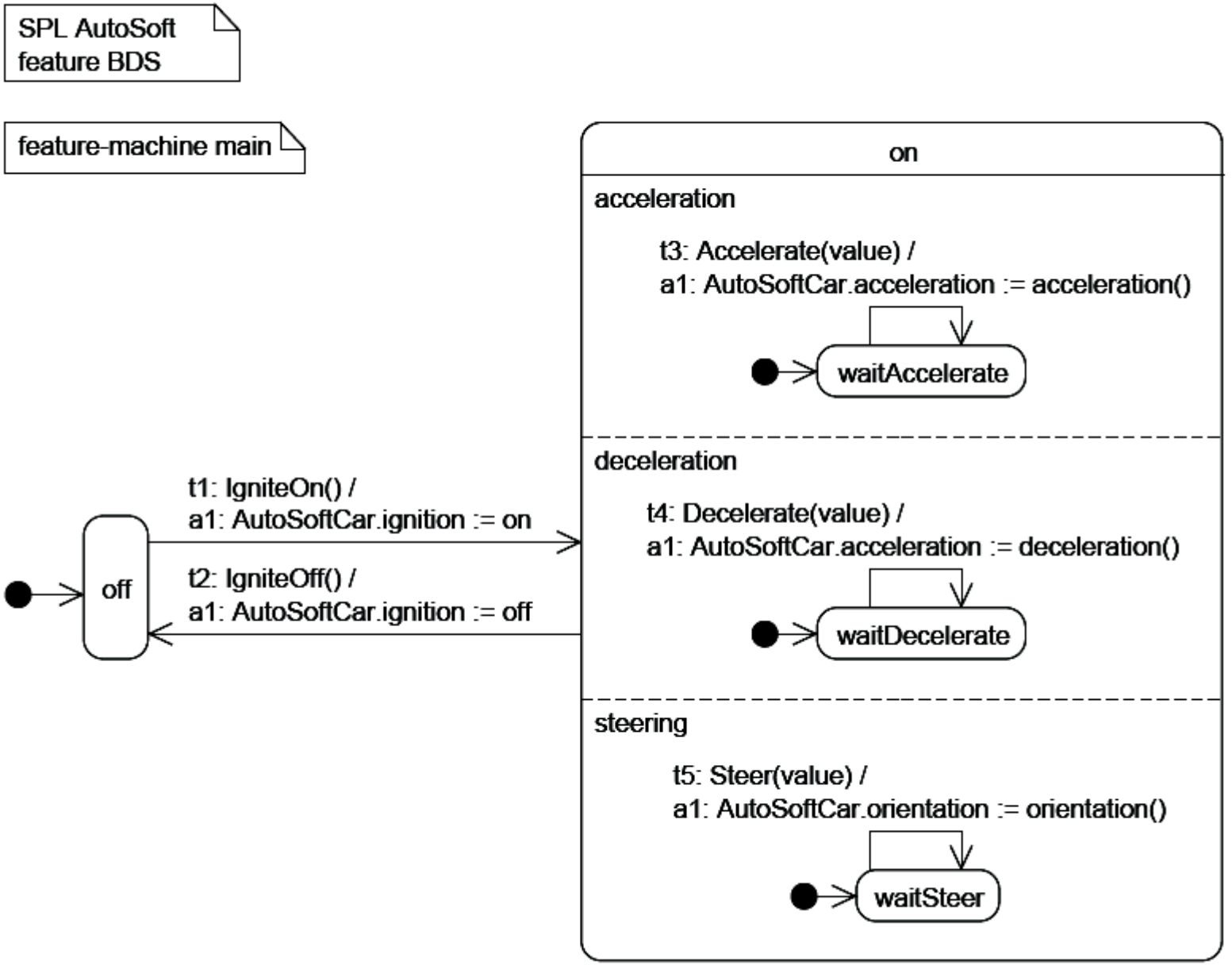


«AutoSoft»
BDS

«inputs»
IgniteOn()
IgniteOff()
Steer(value: Int)
Accelerate(value: Int)
Decelerate(value: Int)

enum IgnitionState = {on, off}

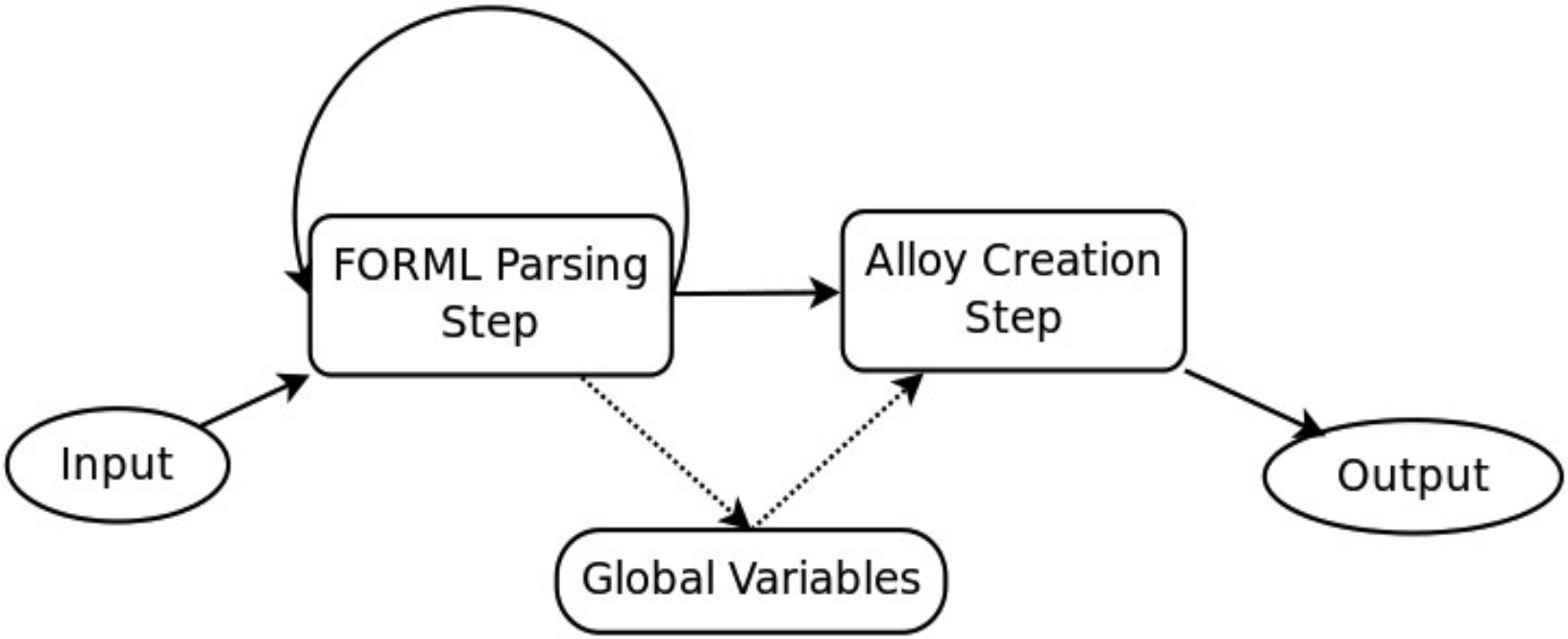
FORML – Behaviour Model



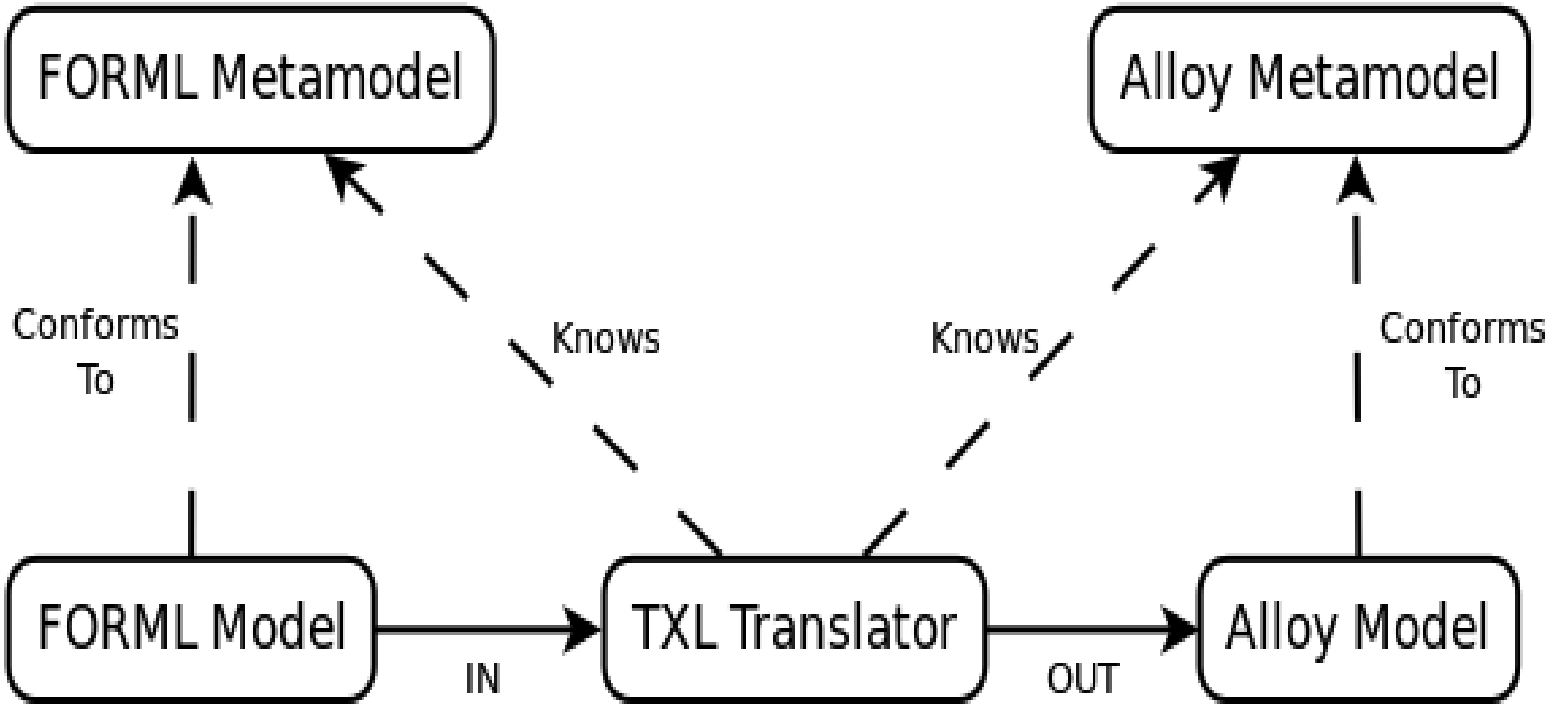
The Translator

- Takes a FORML model as input and produces an Alloy model
- Written in the Turing eXtender Language (TXL)
- I have only extended the translator, initial parts of it were written by Jan Gorzny (now at U of T)

Translator Architecture



Translation Overview

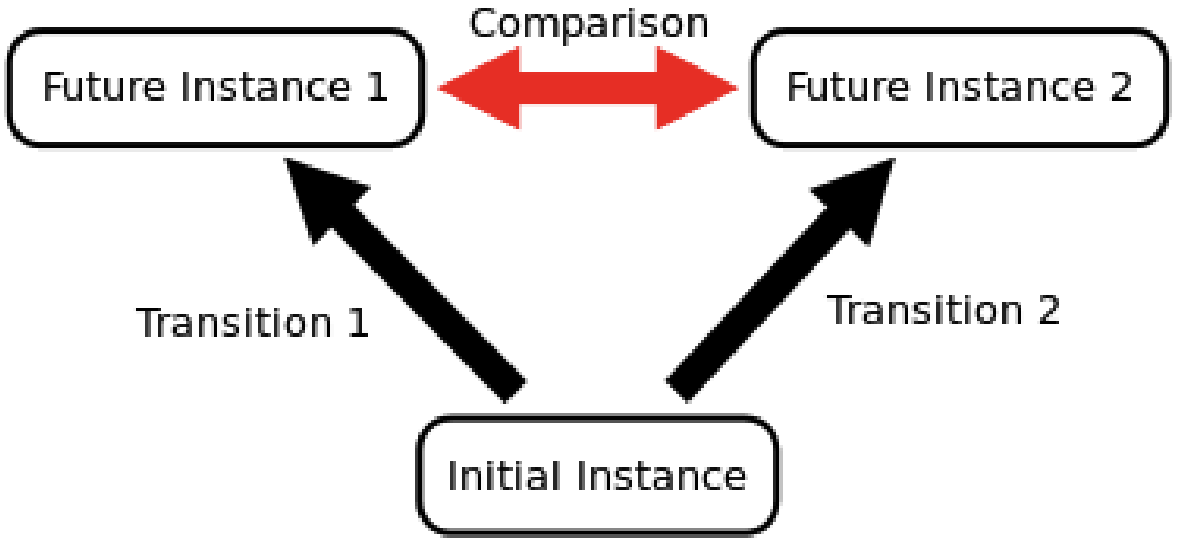


Actions on Transitions

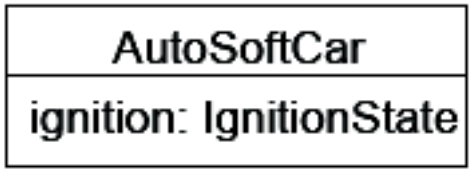
- A transition can have several World Change Actions (WCAs) on it
- The types of WCAs are:
  - Adding an object
  - Removing an object
  - Changing an attribute on an object
- Each object has a predicate created for each of the possible WCAs

How does the analysis take place?

(1) Two transitions are chosen
(2) Each transition has its action(s) executed, resulting in 2 new Future Instances
(3) The resulting futures are compared to find locations where the post conditions overlap



WCA Example




```
pred remove_AutoSoftCar (ws0, ws1 : WS, o1 : AutoSoftCar) {
    o1 in ws0.AutoSoftCars
    ws1.AutoSoftCars = ws0.AutoSoftCars - o1
}
pred add_AutoSoftCar (ws0, ws1 : WS, o1 : AutoSoftCar) {
    o1 not in ws0.AutoSoftCars
    ws1.AutoSoftCars = ws0.AutoSoftCars + o1
}
pred change_AutoSoftCar_ignition (ws0, ws1 : WS, o1 : AutoSoftCar, v1 : IgnitionState) {
    o1 in ws0.AutoSoftCars
    o1.(ws1.AutoSoftCar_ignition) = v1
}
```

Removing an object

Adding an object

Changing the ignition attribute

Transition Example

```
pred AutoSoft_BDS_main_t1 (ws0, ws1 : WS, a1_v1 : IgnitionState, a1_o1 : AutoSoftCar) {
    ws1.AutoSoftCars = ws0.AutoSoftCars
    ws1.AutoSoftCar_orientation = ws0.AutoSoftCar_orientation
    ws1.AutoSoftCar_acceleration = ws0.AutoSoftCar_acceleration
    ws1.Decelerate_value = ws0.Decelerate_value
    change_AutoSoftCar_ignition [ws0, ws1, a1_o1, a1_v1]
}
```

Frame Conditions

Change Ignition WCA

3 Kinds of Analysis

- Pairs of transitions which can remove and change the same object
- Pairs of transitions which can both change the same attribute on an object
- Single transitions which violate world state constraints
- Each method is encoded as an Alloy assertion

World State Constraints (WSC)

- `pred WSC(world_state){...}`
- Set of constraints over a single instance of the world model
- Implemented as a predicate in the Alloy model
- Contains:
  - Encodes cardinality constraints
  - Constraints specified by the user in their FORML model

Assertion Example

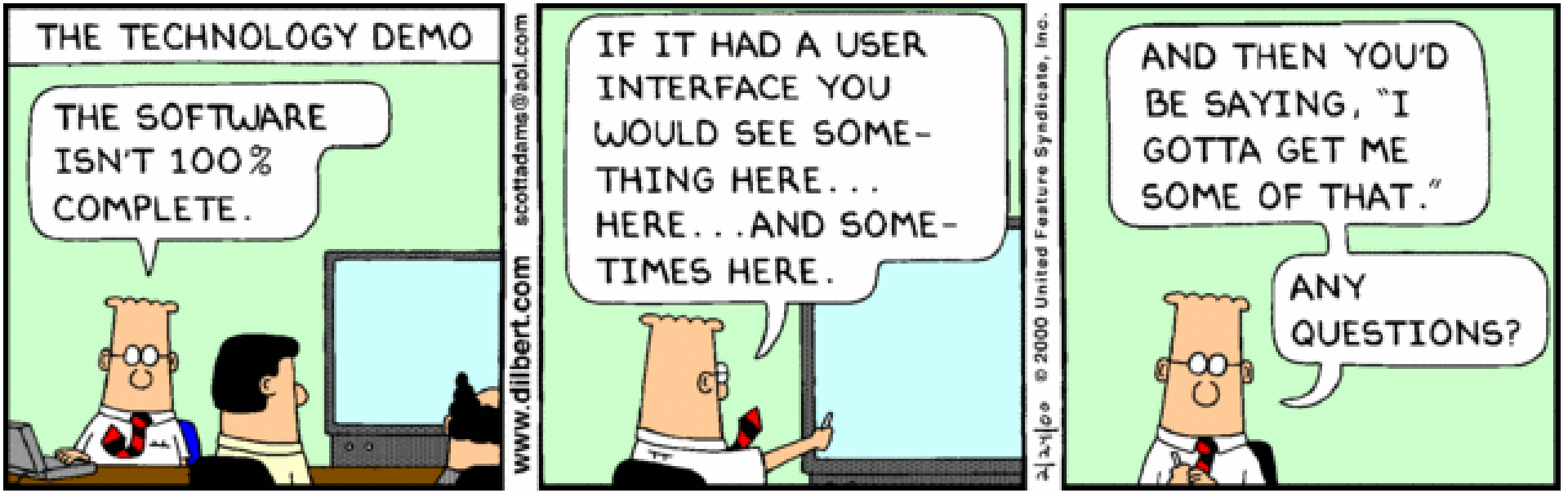
- An example of an assertion to check that a transition does not violate a World State Constraint

```
assert WSC_AutoSoft_BDS_main_t1 {
    all a1_v1 : IgnitionState, a1_o1 : AutoSoftCar |
        WSC [ws0] and AutoSoft_BDS_main_t1 [ws0, ws1, a1_v1, a1_o1] implies WSC [ws1]
}
```

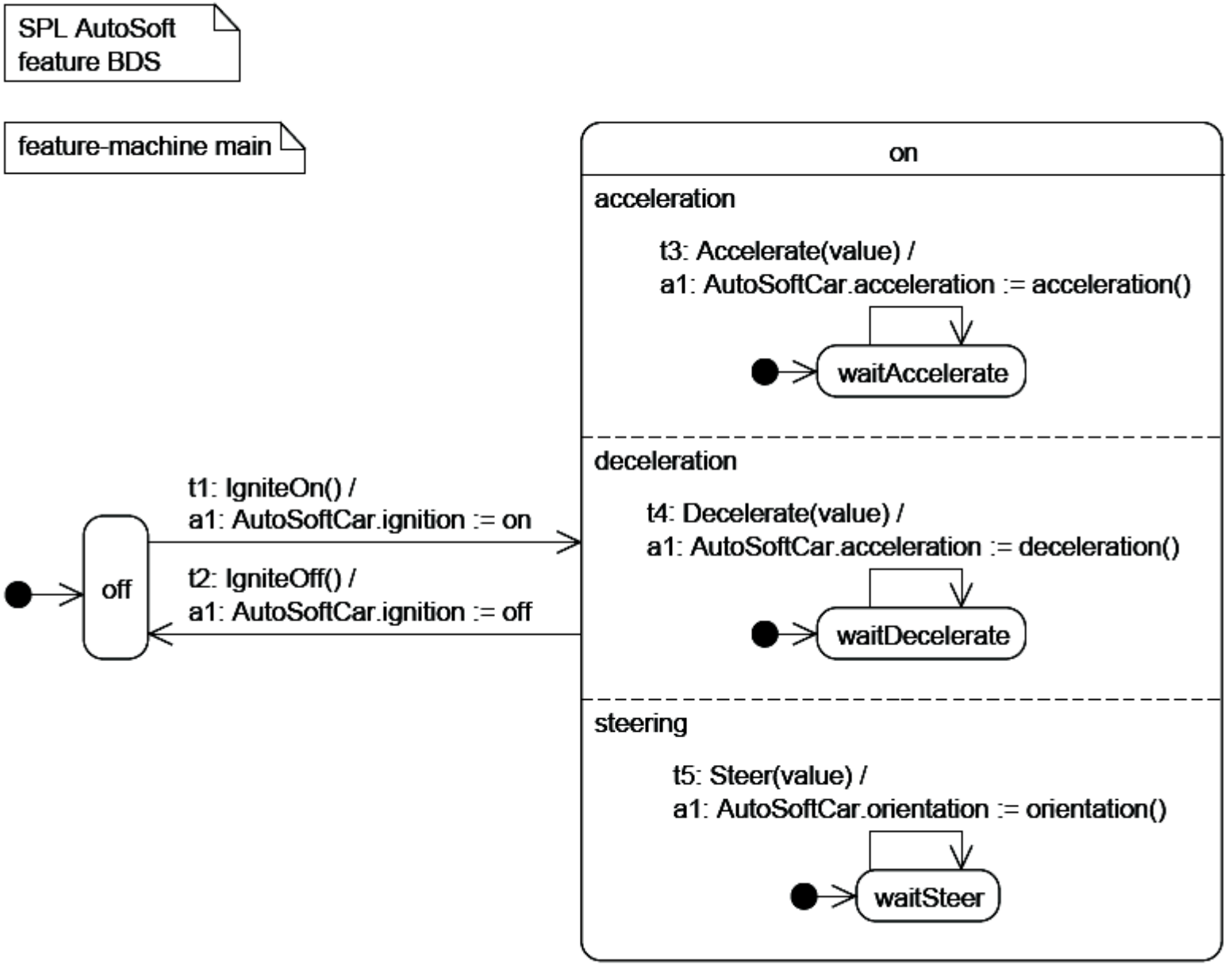
World State Transition Constraints

- `pred WSTC (world_state1, world_state2) {...}`
- Needed when a transition moves to a future state
- These encode constraints such as:
  - The parts of a composition must belong to the same whole over their lifetime
- Each transition must also encode its frame conditions

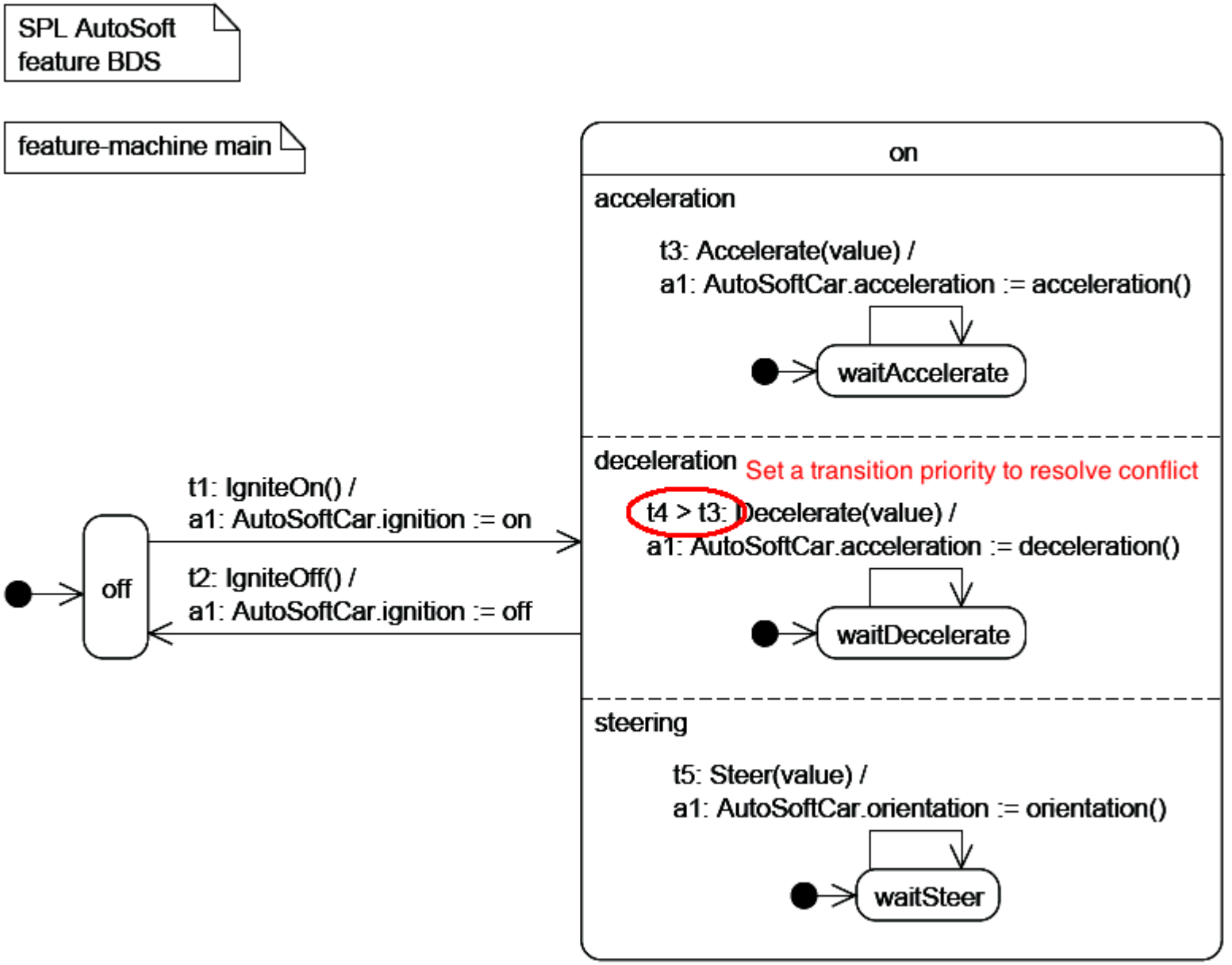
Demo


THE TECHNOLOGY DEMO
THE SOFTWARE ISN'T 100% COMPLETE.
IF IT HAD A USER INTERFACE YOU WOULD SEE SOMETHING HERE... HERE...AND SOMETIMES HERE.
AND THEN YOU'D BE SAYING, "I GOTTA GET ME SOME OF THAT."
ANY QUESTIONS?
www.dilbert.com scottadams@aol.com
© 2000 United Feature Syndicate, Inc.

FORML – Behaviour Model



Resulting Model



Results

- Translating the entire World Model and important parts of the Behaviour Model
- Implemented 3 methods of finding conflicts between pairs of transitions:
  - 2 transitions changing the same object
  - 1 transition removing, 1 transition changing the same object
  - 1 transition violating the World State Constraints

Editor Support

- Created a simple Emacs Major Mode for FORML
  - eLisp is just a DSL
- Provides syntax highlighting, code completion and (buggy) automatic indentation

Limitations

- Triggers and Guards on transitions are not used
  - The resulting model from the demo would still give a counterexample if re-translated and analyzed
  - This leads to false positives in an otherwise conflict free model
- Naming conflicts may occur
  - This is due to a limitation of TXL

Future Work

- Multiple actions per transition
- Simplified expressions (probably can't be done in TXL)
- Generalize the idea of an interaction to generalize the assertions
- (Refactoring)

TXL

- A fairly simple pattern matching Domain Specific Language
- Works with ambiguous grammars which is great
- Complex things need to be done in unintuitive ways
  - There is an extension called eTXL (however, it is no longer active and the only documentation is a Masters thesis)

Conclusion

- Using this method makes it possible to show correctness of a part of your model
  - This does not prove correctness of your entire model, only a facet of it
- By using a DSL like TXL to create the translator the initial learning overhead was lowered, but it complicated things later on