Abbildung von StoCharts nach MoDeST

Christophe Boutter

Dependable systems and software
Universität des Saarlandes, Saarbrücken







Einleitung

Wir wollen einen StoChart

in MoDeST umwandeln.

```
process S2 {
    {= isInS2=1 =};
    stop
}
```







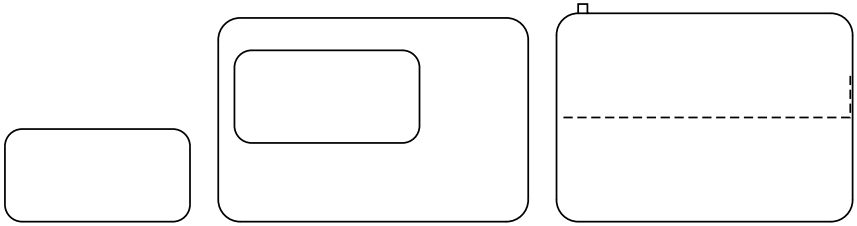
Ziele der Arbeit

- Generische Übersetzung schaffen
- Mit dem Ergebnis der Übersetzung den StoChart simulieren
- Durch Simulation nicht-offensichtliche Probleme erkennen

- Die Idee zu der Arbeit kam während der Erarbeitung des Papers: A *comparative reliability analyis of ETCS train radio communications* von Holger Hermanns, David N. Jansen und Yaroslav S. Usenko

StoCharts

StoCharts sind eine Erweiterung von StateCharts.
StoCharts beinhalten folgendes:

- States: 
- Transitionen:
- P-Pseudoknoten: 
- Actions, aufgeteilt in Trigger und Events
- `after`-Trigger: `after(EXP[2s])`

Warum in MoDeST übersetzen?

Die meisten StoChart-Konstrukte haben eine direkte Entsprechung in MoDeST:

- Der P-Pseudoknoten entspricht dem `palt`-Konstrukt
- Ein And-State entspricht dem `par`-Konstrukt
- Das `after`-Statement entspricht einer Clock-Variablen zusammen mit einer Wahrscheinlichkeitsverteilung
- Trigger und Events entsprechen den MoDeST-Actions







Die Übersetzung

- States werden in MoDeST-Prozesse übersetzt, da sie ein Verhalten repräsentieren.
- Transitionen werden in eine Abfolge übersetzt:
  - Der Guard wird zu einem `when`-Statement welches vor dem weiteren Ablauf steht
  - Der Trigger wird zu einer Action
  - Die Transition selbst wird zu einer Exception, die geworfen wird.
  - Das entsprechende `catch`-Statement enthält die Action, die für das Event steht, und den Prozessaufruf des nächsten States.

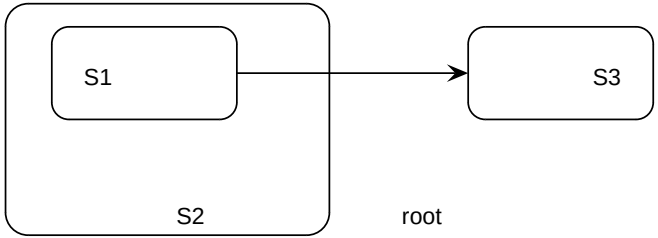






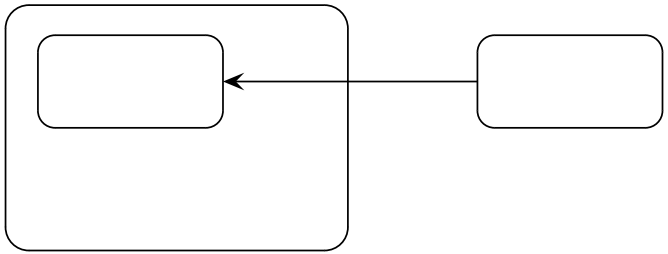
Challenge

Die Bordercrossing-Transitionen machen diese Arbeit erst interessant.



Wegen ihnen werden die Transitionen in Exceptions übersetzt.
Da Transitionen auch einen Scope haben, kann man herausfinden in welchem Scope die Exception gefangen werden muss, damit alle Prozesse die unter diesem Scope liegen durch die Exception auch beendet weren.

Challenge

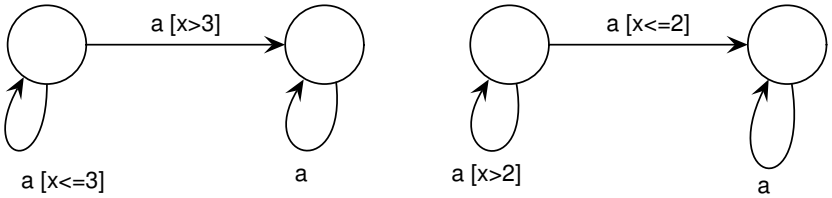


Die eingehenden Bordercrossing Transitionen stellen ein anderes Problem dar.
Jede dieser Transitionen generiert eine eigene Startkonfiguration für den State, in den sie hineinzeigen, so dass hier über Alternativen das richtige Verhalten gewählt werden muss.

```
process State(int alternative) {
   alt { :: when (alternative==0) default
         :: when (alternative==1) 1st incomming BC
         ...
```

Nachteile von MoDeST für diese Arbeit

- StoCharts: wenn eine Transition enabled ist, soll sie auch genommen werden können, unabhängig von anderen aktiven States.
- MoDeST: Synchronisation aller Prozesse über Aktionen im gemeinsamen Alphabet.
- Lösung: *Input Enabledness*: wir garantieren in Modest, dass in jedem Zustand ein *Nullschritt* moeglich ist, bezüglich der gleichen `action` wie der Trigger der betrachteten Transition.









Status

- Übersetzung ist in MosML realisiert.
- Parser für die TCM-Dateien ist in der Implementierung
- Teile des Abschlussberichts sind in der Überarbeitung
- Vorraussichliches Ende: Mitte-Ende November