Simulationszeit

Dario Hornsteiner, Sabine Daniela Hasenleithner, Sofia Bonini

5. Mai, 2021

Simulationszeit: Allgemein

- Eine fiktive Modellzeit, unabhängig von der realen Zeit und der Ausführungsdauer
- Simulationszeit $\longrightarrow$ monoton steigend
- Ereignisse verbrauchen Rechenzeit, aber keine Simulationszeit
- Intervalle zwischen Ereignissen verbrauchen Simulationszeit, aber keine Rechenzeit
- Simulationszeit springt von Ereigniszeitpunkt zu Ereigniszeitpunkt $\longrightarrow$ Simulationsuhr wird weiter gesetzt

TimeSpan

$\longrightarrow$ java.lang.Object

$\longrightarrow$ desmoj.core.simulator.TimeSpan

- Zeitspannen der Simulationszeit
- Constructors =
  - TimeSpan(double duration)
  - TimeSpan(long duration)
  - TimeSpan(double duration, java.util.concurrent.TimeUnit unit)
  - TimeSpan(long duration, java.util.concurrent.TimeUnit unit)
- *duration* $\longrightarrow$ Dauer der Zeitspanne
- Default: Einheit der Referenzzeit

TimeInstant

$\longrightarrow$ java.lang.Object

$\longrightarrow$ desmoj.core.simulator.TimeInstant

- Zeitpunkt der Simulationszeit
- Constructor = TimeInstant(long time,
  java.util.concurrent.TimeUnit unit)
- Parameter: Wert des Zeitpunkts mit gewünschter Einheit
- *time* kann auch *Double*, *Date oder Calender* sein

SimClock

$\longrightarrow$ java.lang.Object

$\longrightarrow$ java.util.Observable

$\longrightarrow$ desmoj.core.simulator.SimClock

- Kapselung der Simulationszeit
- Simulationszeit kann von jedem Objekt abgefragt werden
- Observer können sich bei der Simulationuhr registrieren
- automatischer Statistikzähler $\longrightarrow$ Jedes mal, wenn die Simulationszeit sich ändert, wird ein Zähler benachrichtigt und kann nun den Wert abfragen, den er überwacht
- Könnte Performance schwächen
- Constructor = SimClock(java.lang.String name)

Scheduling

$\longrightarrow$ java.lang.Object

$\longrightarrow$ desmoj.core.simulator.NamedObject

$\longrightarrow$ desmoj.core.simulator.Scheduler

- Das Scheduling soll die Simulation steuern
- Zwei Modellierungsstile:
  1. Ereignisorientierter Modellierungsstil
  2. Prozessorientierter Modellierungsstil (hold, passivate)
- Constructor = Scheduler(Experiment exp, java.lang.String name, EventList eventList)

ShowProgressBar

$\longrightarrow$ java.lang.Object

$\longrightarrow$ desmoj.core.simulator.NamedObject

$\longrightarrow$ desmoj.core.simulator.Experiment

Methoden:

- *public boolean isShowProgressBar()*
- *public void setShowProgressBar(boolean newShowProgressBar)*
- *public void setShowProgressBarAutoclose(boolean autoclose)*

tracePeriod, start, stop

$\longrightarrow$ java.lang.Object

$\longrightarrow$ desmoj.core.simulator.NamedObject

$\longrightarrow$ desmoj.core.simulator.Experiment

- *public void tracePeriod(TimeInstant startTime, TimeInstant stopTime)*
- *public void start()*
- *public void start(TimeInstant initTime)*
- *public void stop(ModelCondition stopCond)*
- *public void stop(TimeInstant stopTime)*

ExecutionSpeedRate und get-Methoden

$\longrightarrow$ java.lang.Object

$\longrightarrow$ desmoj.core.simulator.NamedObject

$\longrightarrow$ desmoj.core.simulator.Experiment

- *public void setExecutionSpeedRate(double rate)*
- Folgende Gleichung gilt hier für eine Geschwindigkeitsrate > 0:
  rate * simulation-time = wall-clock-time (reale Uhrzeit).
  Falls die Rate $= 0$ oder < 0 ist: die Simulation wird so schnell wie möglich durchgeführt.
  Der default Wert beträgt $= 0$ (also so schnell wie möglich).
- get-Methoden zu: *SimClock*, *Model*, *Scheduler*, *RealTimeStartTime*, *StopTime*, *ExecutionSpeedRate*

Code-Beispiel

```
public static void main(String[] args) {

  ExampleModel model = new ExampleModel();
  Experiment exp = new Experiment("");
  model.connectToExperiment(exp);

  exp.setShowProgressBar(true);
  exp.setShowProgressBarAutoclose(true); // close when finished

  exp.setExecutionSpeedRate(60); // couple simulation time to real time
  // rate = 60 -> 1 min simulation time = 1 sec real time

  TimeInstant stopTime = new TimeInstant(5, TimeUnit.MINUTES);
  exp.tracePeriod(new TimeInstant(0), stopTime);
  exp.stop(stopTime);

  exp.start();
}
```