Universität Rostock

Traditio et Innovatio

A DSL for Continuous-Time Agent-Based Modeling and Simulation

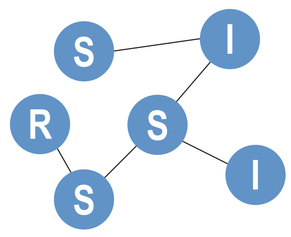
Tom Warnke

Motivation

Most state-of-the-art agent-based modeling and simulation frameworks offer a way to describe agent behavior in a programming language. Whereas these frameworks support easy development of time-stepped models, continuous-time models can only be implemented by manually scheduling and retracting events. To facilitate a separation of concerns into model- and simulation-specific code for continuous-time ABMS, we propose an embedded domain-specific language, which allows describing agent behavior concisely, and corresponding simulation algorithms, which allow executing continuous-time models.

Example SIR Model

- Agents are connected in a network
- Agents are either **S**usceptible, **I**nfectious, or **R**ecovered
- Initially, agents are susceptible or infectious
- Susceptible agents get infected after a random waiting time that depends on the number of infectious network neighbors
- Infectious agents recover after a random waiting time

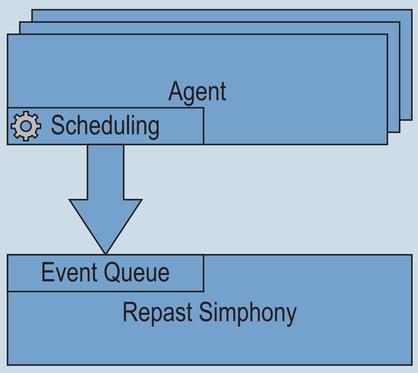


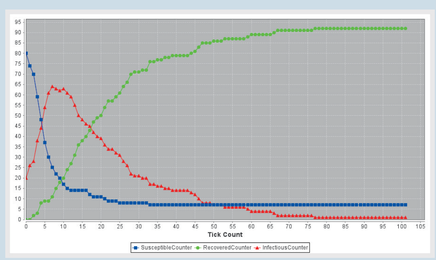
Implementing the SIR model without and with the DSL

The SIR model in vanilla Repast Simphony

138 lines of Java code in the agent class, of which 53 are behavior definition

```
private ISchedulableAction scheduledEvent;

public void getInfected() {
   this.infectionState = InfectionState.INFECTIOUS;
   scheduleRecovery();
   informNeighbours();
}

private void informNeighbours() {
   for (Agent agent : network.getAdjacent(this)) {
      agent.rescheduleInfectionEventIfPresent();
   }
}

public void rescheduleInfectionEventIfPresent() {
   if (infectionState == InfectionState.SUSCEPTIBLE) {
      if(scheduledEvent != null) {
         schedule.removeAction(scheduledEvent);
      }
      scheduleInfection();
   }
}

private void scheduleInfection() {
   // ...
}
```





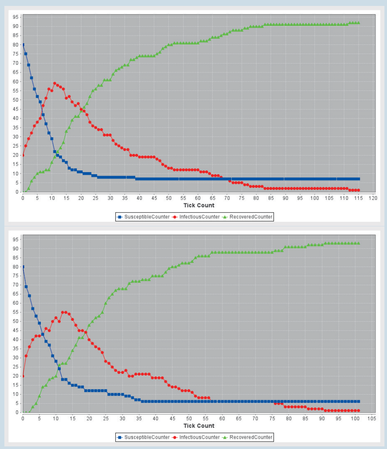
Simulation results from the model with manual scheduling

The SIR model in Repast Simphony with the DSL

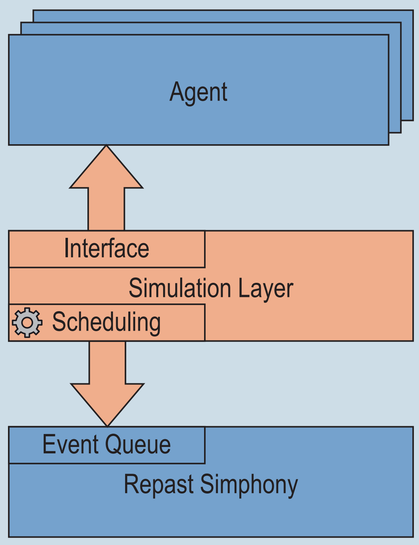
42 lines of Java code in the agent class, of which 10 are behavior definition

```
addRule(() -> this.isInfectious(),
        () -> exp(recoverRate),
        () -> this.infectionState = InfectionState.RECOVERED);

addRule(() -> this.isSusceptible(),
        () -> exp(infectionRate
                * neighbours(SIRAgent.class).filter(
                (SIRAgent agent) -> agent.isInfectious())
                .size()),
        () -> this.infectionState = InfectionState.INFECTIOUS);
```

- Reusable simulation layer separates model- and simulation-specific code
  - Provides interface for specifying agent behavior in an embedded domain-specific language without accessing the event queue
  - Rule-based syntax with guards, a waiting time expression, and an effect
  - Uses Java 8 lambda expressions for succinct anonymous function definitions
- Model is executed by CTMC-style (discrete event) stochastic simulation algorithms
  - First Reaction Method - recalculates all events after each executed event
  - Next Reaction Method - exploiting locality to keep unaffected events in the queue



Simulation results from the model executed with the First Reaction Method and the Next Reaction Method



Results

- Compact description of continuous-time agent-based models
- Syntax inspired by rule-based modeling languages
- CTMC semantics with semantically sound simulation algorithms (two so far)
- Arbitrarily complex functions can be used inside the rules
- Event queue, observation, visualization etc. from Repast Simphony is available

References

North, M. J., N. T. Collier, J. Ozik, E. R. Tatara, C. M. Macal, M. Bragen, and P. Sydelko. 2013. “Complex Adaptive Systems Modeling with Repast Simphony”. Complex Adaptive Systems Modeling 1 (1): 1–26.

Warnke, T., T. Helms, and A. M. Uhrmacher. 2015. "Syntax and Semantics of a Multi-Level Modeling Language". In Proceedings of the SIGSIM PADS ’15, 133–144: ACM.

Warnke T., A. Steiniger, A. M. Uhrmacher, A. Klabunde, and F. Willekens. 2015. "ML3: a language for compact modeling of linked lives in computational demography". In Proceedings of the 2015 Winter Simulation Conference.

Warnke, T., O. Reinhardt, and A. M. Uhrmacher. 2016. "Population-Based CTMCs and Agent-Based Models". In Proceedings of the 2016 Winter Simulation Conference.

Tom Warnke | tom.warnke@uni-rostock.de

Modeling and Simulation Group

University of Rostock

mosi.informatik.uni-rostock.de