

Simulation Modeling using Protégé

Henrik Eriksson¹, Magnus Morin², Joakim Ekberg³,
Johan Jenvald², Toomas Timpka^{1,3}

¹Dept. of Computer and Information Science, Linköping University

²VSL Research Labs, Linköping, Sweden

³Dept. of Medicine and Health, Linköping University

Introduction

Running a computer-based simulation is often a matter of developing a simulation model and executing it in a simulation engine. A transparent separation of the model from the simulation code clarifies the problem and supports model modifications (e.g., for running alternative version of the model). It is possible to use ontology-based models to specify the vocabulary for the simulation as well as other relevant concepts and relationships in the model [1]. Such ontology-based models work well for discrete-event simulation, but can be less suitable for mathematically-oriented simulation based on equations. The ontology-based models, however, often require additional processing before it is possible to use them in a simulation engine.

It is possible to use Protégé as a platform for managing ontology-based models for simulation. We have used Protégé in combination with a custom-developed Protégé extension to support modeling for infectious-disease outbreak simulation. The extension assists users in navigating the model and in generating XML-based specifications for the simulation engine. An important advantage of this approach is that it is possible to modify the model at the Protégé level and that users can develop new models by bringing together model components, such as disease, community, and intervention submodels.

Background

Simulation of infectious-disease outbreaks can support preparedness, intervention planning, and response during ongoing events [2]. It is possible to use stochastic discrete-event simulation models to study outbreaks in simulated populations [3]. This type of simulation takes advantage of mixing networks that define the interaction patterns in the community. The model is based on *mixing groups* with defined transmission probabilities and contact rates. The simulator maintains the persons' disease states and their participation in mixing groups, and propagates the epidemic state according the transmission probabilities in the mixing groups.

Simulation Architecture

The main requirement on the simulation architecture is that it should combine modeling flexibility with run-time efficiency. Modeling flexibility is important because the domain requires simulation of multiple scenarios with varying models. Run-time efficiency is important because of the complexity of large simulated populations and time-consuming repetitions to produce an average of many stochastic simulation runs.

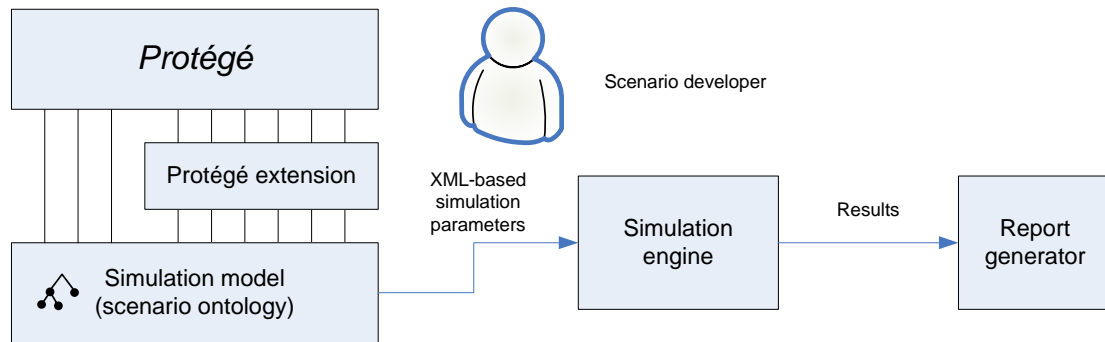


Figure 1. Simulation architecture based on an explicit simulation model.

Our approach is to use a simulation architecture that separates modeling and execution. Figure 1 shows the simulation architecture and its major components. Protégé with the custom-developed extension supports editing of the simulation model. The simulation model, which is OWL-based, is the basis for producing the simulation parameters. These parameters define the simulator settings, the initial state of the simulation, the random number seeds, as well as the disease, community, and intervention submodels. The simulation engine (which is implemented in C++ for efficiency) reads the simulation parameters at startup and initiates the simulation. Finally, the simulator engine outputs the simulation results in an XML-format. A report generator can then transform the results to alternative formats, such as HTML.

Simulation Model

Protégé is the basis for the simulation modeling part. We use the standard Protégé/OWL tabs for defining concepts and relationships, and two additional tab extensions for improved navigation and generation of simulation parameters.

Figure 2 shows the scenario tab extension to Protégé. This tab allows users to define new scenarios by specifying disease, community, and intervention submodels. Also, it is possible to compose new scenarios from reused scenario components. For example, users can employ the same community model in several types of scenarios. Alternatively, users can create new scenarios that vary the community model (e.g., the age profile and household sizes). The scenario descriptions include a list of assumptions that the scenario and its submodels make. This assumptions model documents the scenario and assists users in keeping track of the underlying assumptions for each simulation result [4].

In addition to the scenario tab, there is also a tab for simulation jobs. This tab allows users to configure and manage simulation runs, which consists of a number of scenarios to simulate. Users can specify job-specific parameters for the simulation and list the scenarios to run as part of the job. In the ontology, the scenarios and the simulation jobs are individuals of the Scenario and SimulationJob classes. The combination of the scenarios and the simulation jobs makes it possible to use the simulator for hypotheses testing by rapidly defining and simulating a series of scenarios.

The tab extension for simulation jobs includes a function for producing the XML-based simulation parameters (Figure 1) from the scenarios. This XML generator takes advantage of Jess and JessTab to analyze the OWL individuals and generate the XML output.

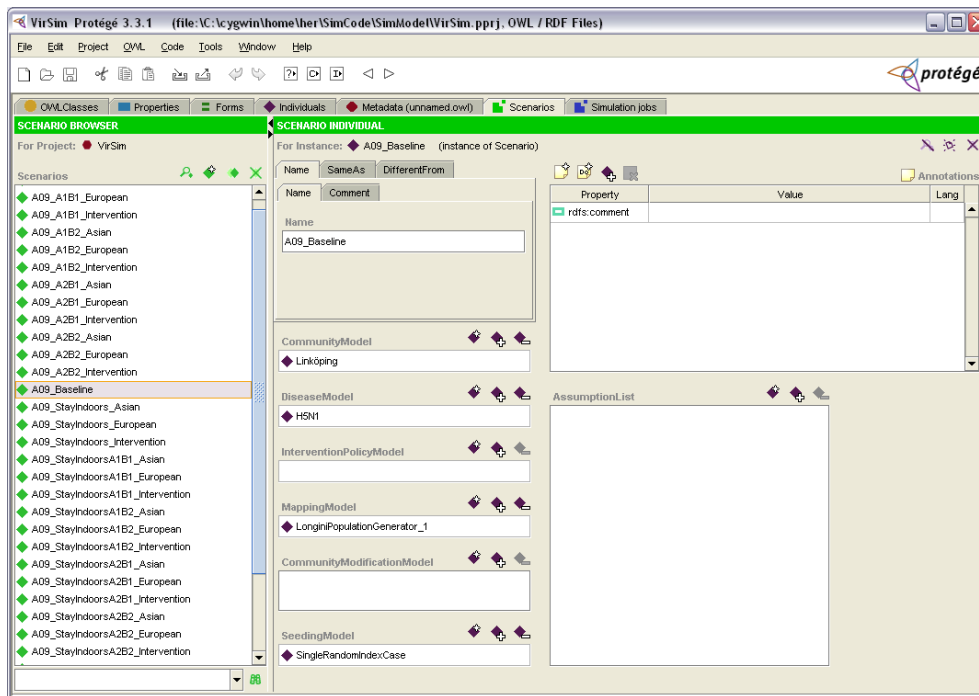


Figure 2. Protégé with the scenario tab extension. The left-hand side of the “Scenario” tab shows a list of the model scenario specifications. The right-hand side of the tab is the editor for the scenario individual.

Summary and Conclusions

We found that the separation of the simulation model and the simulation engine helped clarify the simulation model and the underlying model assumptions. Furthermore, the reuse submodels facilitated modeling efficiency. The use of ontologies and Protégé as the modeling platform made it easier to structure the models and to develop a library of models used for simulation.

Acknowledgements

This work was supported by the Swedish Research Council under contracts 2006-4433 and 2008-5252.

References

- [1] Eriksson H, Morin M, Jenvald J, Gursky E, Holm E, Timpka T. Ontology based modeling of pandemic simulation scenarios. *Stud Health Technol Inform.* 2007;129:755-9.
- [2] Jenvald, J., Morin., M., Timpka, T. & Eriksson, H. (2007). Simulation as Decision Support in Pandemic Influenza Preparedness and Response. In *Proc. of ISCRAM 2007*, pp. 295–304, May 13–16, Delft, The Netherlands.
- [3] Timpka T, Morin M, Jenvald J, Eriksson H, Gursky E. Towards a simulation environment for modeling of local influenza outbreaks. *AMIA Annu Symp Proc.* 2005:729-33.
- [4] Henrik Eriksson, Magnus Morin, Joakim Ekberg, Johan Jenvald, and Toomas Timpka. Assumptions Management in Simulation of Infectious Disease Outbreaks. Submitted for publication.