

Eclipse Plug-in for Spin and st2msc Tool

Tim Kovše, Boštjan Vlaovič, Aleksander Vreže, Zmago Brezočnik

Faculty of Electrical Engineering and Computer Science,
University of Maribor,
Maribor, Slovenia

{tim.kovse, bostjan.vlaovic, aleksander.vreze, brezocnik}@uni-mb.si

Abstract. In this article we present an Eclipse plug-in for Spin and Spin Trail to Message Sequence Chart conversion tool (st2msc). The plug-in can be used to edit a Promela model, run the formal verification of the model, and generate optimized MSC of the Spin trail with st2msc. It simplifies handling with extensive Promela models to a great extent.

1 Introduction

In an ideal development environment each change in the specification of the product would be immediately formally checked against requirements specification. This is not an easy task. To verify a real-life system it must be usually converted into a simpler “verifiable” format—the model of the (sub)system. Next, each requirement should be formally described in a way that developers fully understand its true meaning. If we use one of the temporal logics, this goal is not always easy to achieve. Finally, we check if the model satisfies the requirements. Several formal verification techniques and tools are available. We use the Simple Promela Interpreter (Spin) model checker [1], which suits our needs.

The final research goal is to build a framework for the systematic use of model checking in the software development process of our industry partners. We focus on Specification and Description Language (SDL) specifications that describe the implementation details and are used to build the production systems. Such specifications use SDL constructs and additional extensions that enable developers to include operators that are implemented in other programming languages. Our industry partners use C programming language for low-level or processor intensive operations. Therefore, the framework should support such SDL extensions.

A model can be generated manually or mechanically. Manual preparation of a model is error-prone and very time consuming task. Therefore, our research focuses on automating model generation from SDL specification and manageable presentation of the model and verification results.

In [2–4] we proposed a new approach to automated model generation from SDL to Promela. The generation of a model is based on formally specified algorithms. We validate these algorithms in practice with the use of an SDL to Promela (*sdl2pml*) tool, which implements most of our research results [5]. Applicability of our approach was tested on an implementation of an ISDN User

Adaptation (IUA) protocol, which is part of the SI3000 softswitch. Specification was developed by the largest Slovenian telecommunication equipment company, Iskratel d.o.o. It consists of 97,388 lines of SDL'93 code without comments and blank lines. An abstracted version of those blocks that were prepared for the automated generation of model consists of 28,563 lines of SDL code. The generated Promela model comprises of 79,281 lines of code. We have chosen IUA protocol because it includes all SDL constructs that are used in the specification of the softswitch. Additionally, it includes many external operators written in C with the use of extended Abstract Data Type (ADT) definitions. After semi-automatic generation of a model, we successfully ran the simulation and managed to discover an unknown invalid execution path.

When real-life telecommunication systems are studied, Spin trail can be very demanding. To ease the search for an error, we developed Spin Trail to Message Sequence Chart (*st2msc*) tool [6–8]. It provides automated generation of a Message Sequence Chart (MSC) diagram from Spin execution trail. Spin trail of the simple call with the use of IUA protocol consists of 55.371 lines of text. It contains 21 processes that interact with the use of 261 messages.

During the study of the IUA protocol, we learned that Promela editor with code folding capabilities and version control would improve an engineer's user experience and efficiency. Additionally, a need for a common development environment for the Spin, *sdl2pml*, and *st2msc* emerged. In this paper, we focus only on Eclipse plug-in for Spin and *st2msc*.

2 Eclipse plug-in for Spin and *st2msc* tool

Large Promela models are usually difficult to examine in common text editors. Therefore, we started the development of an Eclipse environment that would offer a better overview of a prepared model and simplify the use of *st2msc* and *sdl2pml*. We have chosen widely used Eclipse¹ platform for software development. This open source tool is capable of handling Integrated Development Environments (IDEs) for many different programming languages, including Java, C/C++, and PHP [9].

During our search for the existing solutions, we found Eclipse plug-in for Spin [10]. Unfortunately, it does not provide code folding that initiated our research. Therefore, we decided to start a new development project with the aim to simplify engineers work with extensive Promela models and provide integration for existing tools developed by our group. The developed plug-in includes:

- Promela file wizard,
- Spin perspective,
- Promela editor,
- preference pages,
- launch shortcuts, and
- update site.

¹ <http://www.eclipse.org/>

Preparation of a new Promela model in Eclipse IDE requires creation of a new file. Developed file wizard simplifies the creation of the file that is used as a container for a model. It automatically inserts the `init` function into the model. Existing models can be included with Eclipse’s import manager.

Eclipse perspective is a visual container for set of views and editors. Our plug-in implements Spin perspective. It contains the following views:

- St2msc—conversion of a Spin trail file to MSC diagram,
- Package Explorer—display of files associated with the model,
- Console—output of the Spin tool, and
- Problems—syntax problems in the model.

Spin perspective additionally includes Promela editor that can be used for viewing or editing a model. Current version includes several features, e.g., syntax highlighting, code folding, problem markers, and content assistant.

If syntax highlighting is enabled, all defined keyword groups are displayed in selected colour. Promela language reference defines the following keyword groups: Meta Terms, Declarators Rule, Control Flow, Basic Statements, Predefined, Embedded C Code, Omissions, and Comments. By default, all Promela keywords are included, but they can be extended by the user.

Code folding temporarily hides sections of the Promela code enclosed by brackets or between reserved pair of words: `do—od` and `if—fi`. This functionality is essential if real-life specifications are studied.

Promela editor utilizes Eclipse’s problem markers that provide a powerful error reporting mechanism. Syntax errors are also shown in the Problems view. Syntax error position is parsed from the output of a Spin’s syntax check. Additional help is provided by the Content assistant that can be used for the completion of the reserved keywords.

Settings for Promela editor, simulation, and verification are set in preference pages. Additionally, paths to external tools (Spin, *XSpin*, *C compiler*, and *st2msc*) are defined. Customizations of the editor include user-defined keywords for the syntax highlighting and color selection for keyword groups. Here, highlighting and code folding can be enabled or disabled.

At the moment, random and guided simulations are supported. Simulation parameters “skipped simulation steps” and “seed value” can be set. For guided simulation `pan.in.trail` file or user-selected trail file can be used. Simulation output is shown in the Console view. For interactive simulation XSpin tool can be launched.

In Fig. 1 Spin perspective is shown. Left side of the figure presents Package Explorer view with `iua` project. It includes Promela model file `iua_model.pm1`. At the forefront of the figure, options for the Spin verification are shown. The verification preference page is divided into two parts. In the upper part, user can import, export, or reload verification parameters. An engineer can load and store verification parameters in XML format. If loaded parameters are changed during the study of the model, primary settings can be easily reloaded. Verification options are included in the lower part of the verification preference page.

The Promela model of the system is shown behind the preferences page. An inline definition of `send_expire_timers(queue)` is collapsed, while `set(tmr, value)` is expanded. Promela editor's syntax highlighting is shown on comments and following keywords: `inline`, `typedef`, `hidden`, and `do`. Result of a syntax check is shown in Console view at the bottom of the Fig. 1.

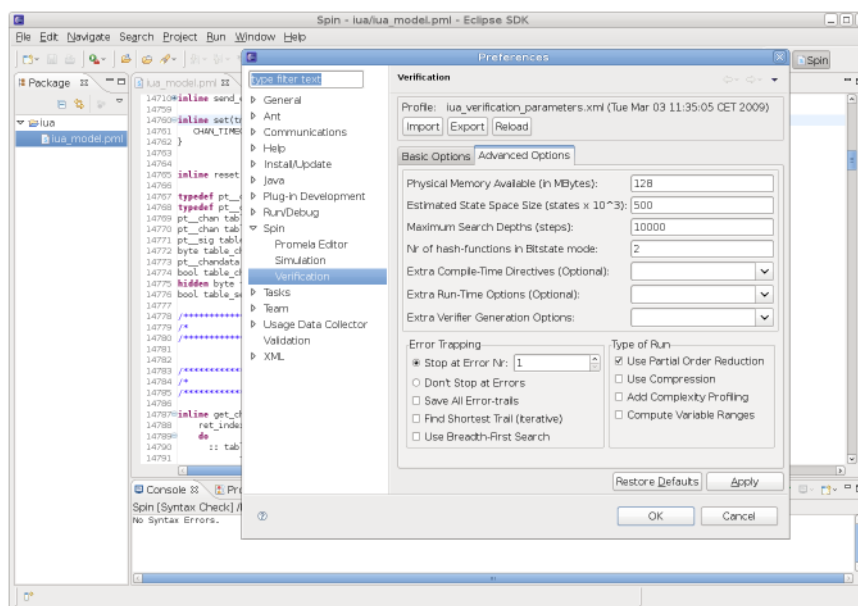


Fig. 1. Screen capture of Eclipse plug-in for Spin and *st2msc* tool

After successful study and preparation of the model, user can run the simulation or formal verification of the model. The Spin tool tracks the execution of the simulation in the trail file. Additionally, trail file describes the counterexample of the verification run. To ease the search for an error, we developed *st2msc* tool for automated generation of ITU-T Z.106[11] and Z.120[12] standardized MSC diagrams. Its graphical interface is implemented as a view in Eclipse plug-in. Engineer can focus on specific parts of the trail with selection of messages that should be included in the MSC diagram. Additionally, user can merge processes into virtual processes [8]. Generated diagrams can be studied with the use of any MSC tool, e.g., ObjectGEODE. The *st2msc* tool is implemented in Java and consist of 1.216 lines of code.

The plug-in is in constant development. Therefore, we have implemented feature for automatic installation and future updates from the development server. For the installation of the plug-in, Eclipse update manager can be used.

3 Conclusion

Using the implementation of the IUA protocol as a test case, we have shown how Eclipse plug-in for Spin and st2msc tool can be very helpful during development of Promela models. Code folding and syntax highlighting functionalities in Promela editor give a user a better overview of a modelled system.

One of our major goals for further work is to implement a guided simulation in our plug-in. Another important goal is to integrate an MSC editor into Eclipse IDE. If the latter is realized, users will not need to install third-party tools for graphical presentation of MSC diagrams. We also plan to include our sdl2pml tool into the Eclipse plug-in. If this is done, all steps of the real-life verification system starting from the SDL specification of a system down to the verification of its model by model checking will be covered by a single user-friendly framework.

References

1. Holzmann, G. J.: The Spin Model Checker, Primer and Reference Manual. Addison-Wesley (2004)
2. Vlaovič, B.: Automatic Generation of Models with Probes from the SDL System Specification: Ph.D dissertation (in Slovene). University of Maribor, Faculty of EE&CS, Maribor, Slovenia (2004)
3. Vreže, A.: Extending automatic modeling of SDL specifications in Promela with embedded C code and a new model of discrete time: Ph.D dissertation (in Slovene). University of Maribor, Faculty of EE&CS, Maribor, Slovenia (2006)
4. Vlaovič, B., Vreže, A., Brezočnik, Z., Kapus, T.: Automated Generation of Promela Model from SDL Specification. *Computer Standards & Interfaces* **29**(4) (2007) 449–461
5. Vreže, A., Vlaovič, B., Brezočnik, Z.: Sdl2pml — Tool for automated generation of Promela model from SDL specification. In: *Computer Standards & Interfaces*, in press
6. Kovše, T., Vlaovič, B., Vreže, A., Brezočnik, Z.: Automatic Generation of MSC Diagram from Spin Execution Trail (in Slovene). *Electrotechnical Review*, submitted for publication
7. Kovše, T.: Integration of Spin tool into Development Environment Eclipse (in Slovene). University of Maribor, Faculty of EE&CS, Maribor, Slovenia (2008)
8. Kovše, T., Vlaovič, B., Vreže, A., Brezočnik, Z.: Spin Trail to Message Sequence Chart Conversion Tool. In: *The 10th International Conference on Telecommunications*, Zagreb, Croatia
9. Clayberg, E., Rubel, D.: Eclipse: Building Commercial-Quality Plug-ins. Addison-Wesley Professional (2006)
10. Rothmaier, G., Kneiphoff, T., Krumm, H.: Using SPIN and Eclipse for Optimized High-Level Modeling and Analysis of Computer Network Attack Models. *SPIN 2005, LNCS* (2005) 236–250
11. International Telecommunication Union: Common interchange format for SDL. Recommendation Z.106, Telecommunication Standardization Sector of ITU, Geneva, Switzerland (1996)
12. International Telecommunication Union: Message sequence chart (MSC). Recommendation Z.120, Telecommunication Standardization Sector of ITU, Geneva, Switzerland (1999)