

On the Monitorability of Session Types, in Theory and Practice (Artifact)

Christian Bartolo Burlò  

Gran Sasso Science Institute, L'Aquila, Italy

Adrian Francalanza  

Department of Computer Science, University of Malta, Msida, Malta

Alceste Scalas  

DTU Compute, Technical University of Denmark, Kongens Lyngby, Denmark

Abstract

In the paper “On the Monitorability of Session Types, in Theory and Practice” we study the *monitorability* of message-passing black-box processes against protocol specifications expressed as session types; we formalise a monitor synthesis procedure, prove its correctness, and discuss its implementation — as a tool that synthesises an executable monitor

(in the Scala programming language) from a given session type. This artifact contains the aforementioned monitor synthesis tool, called *STMonitor*; it includes the tool source code, and documentation to reproduce the examples and benchmarks described in the paper.

2012 ACM Subject Classification Software and its engineering → Development frameworks and environments; Software and its engineering → Software verification and validation; Theory of computation → Concurrency

Keywords and phrases Session types, monitorability, monitor correctness, Scala

Digital Object Identifier 10.4230/DARTS.7.2.2

Acknowledgements This work has been partly supported by: the project MoVeMnt (No: 217987-051) under the Icelandic Research Fund; the BehAPI project funded by the EU H2020 RISE under the Marie Skłodowska-Curie action (No: 778233); the EU Horizon 2020 project 830929 *CyberSec4Europe*; the Danish Industriens Fonds Cyberprogram 2020-0489 *Security-by-Design in Digital Denmark*. The authors would like to thank the anonymous reviewers and Mario Alfonso Prado-Romero for testing the artifact and for providing useful feedback.

Related Article Christian Bartolo Burlò, Adrian Francalanza, and Alceste Scalas, “On the Monitorability of Session Types, in Theory and Practice”, in 35th European Conference on Object-Oriented Programming (ECOOP 2021), LIPIcs, Vol. 194, pp. 20:1–20:30, 2021.

<https://doi.org/10.4230/LIPIcs.ECOOP.2021.20>

Related Conference 35th European Conference on Object-Oriented Programming (ECOOP 2021), July 12–16, 2021, Aarhus, Denmark (Virtual Conference)

1 Scope

This artifact includes the monitor synthesis tool described in Section 5 of the companion paper. Its purpose is to allow the reproduction of the example discussed in Section 5.2 of the companion paper, and the benchmarks in Section 6.

2 Content

The artifact package includes:

- the source code of *STMonitor* (as a compressed archive);
- a ready-to-use VirtualBox image (with Ubuntu 20.04) including *STMonitor* and all the required dependencies. If asked to log in, the credentials are:



© Christian Bartolo Burlò, Adrian Francalanza, and Alceste Scalas; licensed under Creative Commons License CC-BY 4.0

Dagstuhl Artifacts Series, Vol. 7, Issue 2, Artifact No. 2, pp. 2:1–2:3



DAGSTUHL
ARTIFACTS SERIES

Dagstuhl Artifacts Series

Schloss Dagstuhl – Leibniz-Zentrum für Informatik,
Dagstuhl Publishing, Germany



2:2 On the Monitorability of Session Types. . . (Artifact)

- username: `stmonitor`
- password: `stmonitor`

After logging in, `STMonitor` is available in the directory: `/home/stmonitor/artifact`

The artifact instructions are available in the file `README.md`, in the main directory of `STMonitor`. For better readability, you may use a Markdown preview tool; for example, when running the VirtualBox image above you can execute:

```
grip README.md
```

One limitation of `grip` is that it does not fully support links to directories and non-Markdown files. Alternatively, you can read `README.md` directly on GitHub, on:

<https://github.com/chrisbartoloburlo/stmonitor> (release v0.0.1).

3 Getting the artifact

The artifact endorsed by the Artifact Evaluation Committee is available free of charge on the Dagstuhl Research Online Publication Server (DROPS). In addition, the artifact is also available at: <https://github.com/chrisbartoloburlo/stmonitor> (release v0.0.1).

4 Tested platforms

- `STMonitor` has been compiled and tested under Ubuntu 20.04 and 20.10, and MacOS 11.2.3.
- The VirtualBox image has been tested with VirtualBox 6.1 under Ubuntu 20.10, Windows 10, and macOS 11.2.3.
- The benchmarking scripts *require* a Unix-like operating system providing an `/usr/bin/time` utility compatible with GNU Time¹ — e.g., Ubuntu 20.04: see `README.md` for details.

5 License

The artifact is released under the MIT License [1].

6 MD5 sum of the artifact

d95472f57ddf8852dd7edabf5697e6ae

7 Size of the artifact

4.34 GiB

A Instructions

A.1 Kick-the-tires

- If you are using the provided VirtualBox image, you can simply execute the following command from the main directory of `STMonitor`:

¹ <https://www.gnu.org/software/time/>

```
sh scripts/benchmarks.sh kickthetires
```

The command should complete in around 5-10 minutes on a modern computer, and print the directories containing generated plots (in PDF format). If the plots are generated, then the artifact works correctly, and it is possible to continue the evaluation and execute the full benchmarks (see below). Note: the kick-the-tires plots are not very informative, and are only generated as a test.

- If you are **not** using the provided **VirtualBox image**, you will first need to read `README.md` and install the required dependencies.

A.2 Reproducing the benchmarks in Section 6

- If you are using the provided **VirtualBox image**, you can simply execute the following command from the main directory of `STMonitor` (see `README.md` for more details about the benchmarking options):

```
sh scripts/benchmarks.sh 5 smtp-python smtp-postfix pingpong http
```

The command completes in around 3 hours on `VirtualBox` running on a dual-core Intel Core i5, 8 GB RAM, macOS 11.2.3. When it completes, the command prints the directories containing generated plots (in PDF format). For more accurate (and longer) benchmarks, you can replace the argument '5' (which is the number of repetitions) with a higher number.

- If you are **not** using the provided **VirtualBox image**, you will first need to read `README.md` to install the required dependencies.

A.3 Reproducing the example in Section 5.2, and more

Please see `README.md`: it describes more examples, with pointers to the relevant parts of the artifact source code. It also provides more details about the benchmark implementation and options.

References

- 1 The MIT license. URL: <https://opensource.org/licenses/MIT>.