# FM-Tools: Find, Use, and Conserve Tools for Formal Methods

git: https://gitlab.com/sosy-lab/benchmarking/fm-tools
web: https://fm-tools.sosy-lab.org

Dirk Beyer
LMU Munich, Germany

April 24, 2025, at Dagstuhl InfoEx Seminar

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

SoSy-Lab Software Systems

# Vision

- All tools for formal methods work together to solve hard verification problems and make our world safer and more secure.
- Model checkers and theorem provers can be integrated into the software-development process as seamless as unit testing today.
- Model checkers, theorem provers, SMT solvers, and testers use common interfaces for interaction and composition.

# Some Steps Towards the Vision

- **Find**: Which tools for software verification exist?
- ... for test-case generation?
- ... for SMT solving?
- ... for hardware verification?
- **Reuse**: How to get executables?
- Where to find documentation?
- Am I allowed to use it?
- How to use them?
- **Conserve**: Which operating system, libraries, environment?

# Requirements for Solution

- Support documentation and reuse
- Easy to query and generate knowledge base
- Long-term availability/executability of tools
- Must come with tool support
- Approach must be compatible with competitions

# Solution [1]

One central repository:
https://gitlab.com/sosy-lab/benchmarking/fm-tools which gives information about:

- Location of the tool (via DOI, just like other literature)
- License
- Contact (via ORCID)
- Project web site
- Options
- Requirements (certain Docker container / VM)
- Limits

Maintained by formal-methods community

# Example: Entry for KeY

```yaml
name: KeY
input_languages:
  - Java
project_url: https://www.key-project.org/
repository_url: https://github.com/KeYProject/key
spdx_license_identifier: GPL-2.0
benchexec_toolinfo_module: "benchexec.tools.key_cli"
fmtools_format_version: "2.0"
fmtools_entry_maintainers:
  - ricffb
```

# Example: KeY's Contacts

```yaml
maintainers:
  - orcid: 0000-0002-5671-2555
    name: Wolfgang Ahrendt
    institution: Chalmers University of Technology
    country: Sweden
    url: https://www.cse.chalmers.se/~ahrendt/
  - orcid: 0000-0002-9672-3291
    name: Bernhard Beckert
    institution: Karlsruhe Institute of Technology
    country: Germany
    url: https://formal.kastel.kit.edu/beckert/
  - orcid: 0000-0001-8000-7613
    name: Reiner Hähnle
    institution: TU Darmstadt
    country: Germany
    url:
       https://www.informatik.tu-darmstadt.de/se/gruppenmitglieder/groupmembers_detailseite_30784.en.jsp
  - orcid: 0000-0002-2350-1831
    name: Mattias Ulbrich
    institution: Karlsruhe Institute of Technology
    country: Germany
    url: https://formal.kastel.kit.edu/ulbrich/
  - orcid: 0000-0001-8446-4598
    name: Alexander Weigl
    institution: Karlsruhe Institute of Technology
    country: Germany
    url: https://formal.kastel.kit.edu/weigl/
```

# Example: KeY's Versions

```yaml
versions:
  - version: "2.13"
    doi: 10.5281/zenodo.12945286
    benchexec_toolinfo_options: []
    required_ubuntu_packages:
      - openjdk-21-jre-headless
    base_container_images:
      - ubuntu:22.04
```

# Example: KeY's Documentation

```
literature:
  - doi: 10.1007/978-3-030-64354-6
    title: "Deductive␣Software␣Verification:␣Future␣Perspectives␣
       -␣Reflections␣on␣the␣Occasion␣of␣20␣Years␣of␣KeY"
    year: 2020
  - doi: 10.1007/978-3-319-49812-6
    title: "Deductive␣Software␣Verification␣-␣The␣KeY␣Book␣-␣From␣
       Theory␣to␣Practice"
    year: 2016
  - doi: 10.1007/978-3-319-12154-3_4
    title: "The␣KeY␣Platform␣for␣Verification␣and␣Analysis␣of␣
       Java␣Programs"
    year: 2014
  - doi: 10.1007/s10270-004-0058-x
    title: "The␣KeY␣Tool"
    year: 2005
  - doi: 10.1007/3-540-40006-0_3
    title: "The␣KeY␣Approach:␣Integrating␣Object␣Oriented␣Design
```

# Example: KeY's Web-Page Entry

## Tools for Formal Methods: Tools

Tools Techniques Competitions Frameworks Input Languages Documentation of the YAML Schema ↗ Code on GitLab

FuSeBMC-AI
Gazer-Theta
GDart
GDart-LLVM
Goblint
Graves-CPA
Graves-Par
GWIT
Hornix
HybridTiger
Infer
Java-Ranger
JayHorn
JBMC
JCWIT
JDart
KeY
KLEE
KLEEF
Korn
Lazy-CSeq
Legion
Legion/SymCC
LF-checker
LIV
Locksmith

### KeY

KeY is a tool for deductive verification to prove the correctness of Java programs.

**Project URL:** https://www.key-project.org/

**Repository URL:** https://github.com/KeYProject/key

**Maintainers:** • Wolfgang Ahrendt • Bernhard Beckert • Richard Bubel • Reiner Hähnle • Mattias Ulbrich • Alexander Weigl

**Supported input languages:** • Java

**License:** • GPL-2.0

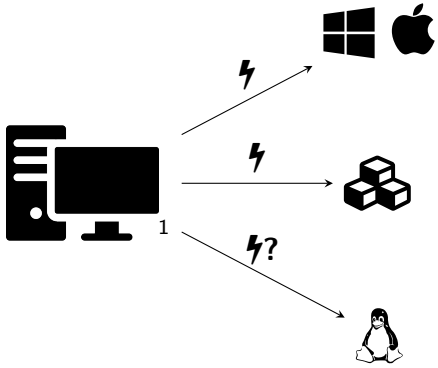
**Releases:** • 2.13

**Literature:**
- *Deductive Software Verification: Future Perspectives - Reflections on the Occasion of 20 Years of KeY. 2020.* DOI: 10.1007/978-3-030-64354-6
- *Deductive Software Verification - The KeY Book - From Theory to Practice. 2016.* DOI: 10.1007/978-3-319-49812-6
- *The KeY Platform for Verification and Analysis of Java Programs. 2014.* DOI: 10.1007/978-3-319-12154-3_4
- *The KeY Tool. 2005.* DOI: 10.1007/s10270-004-0058-x
- *The KeY Approach: Integrating Object Oriented Design and Formal Verification. 2000.* DOI: 10.1007/3-540-40006-0_3

# FM-Tools is FAIR

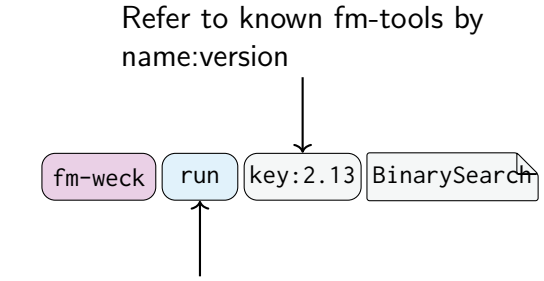
- **F**indable:
  overview is available on internet,
  generated knowledge base
- **A**ccessible:
  data retrievable via Git, format is YAML
- **I**nteroperable:
  Format is defined in schema,
  archives identified by DOIs, researchers by ORCIDs
- **R**eusable:
  Data are CC-BY, each tool comes with a license,
  format of tool archive standardized

# What about the Environment?



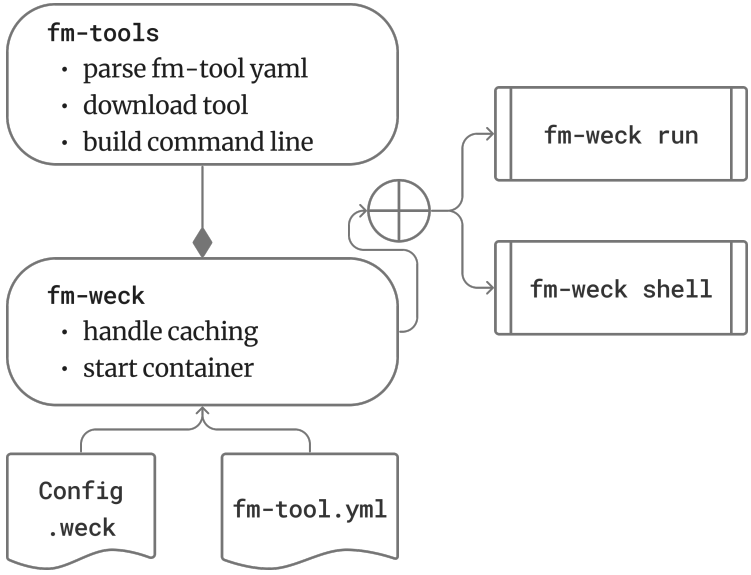

[1] Image: Flaticon.com

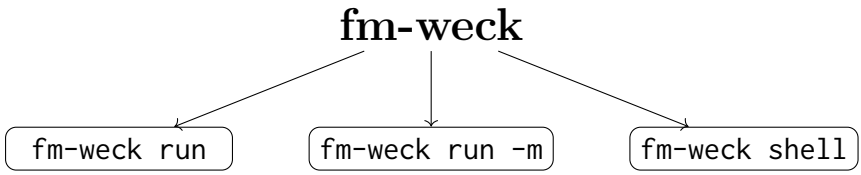
# FM-Weck: Run Tools in Conserved Environment [2, Proc. FM 2024]




- No knowledge of the tools CLI needed
- Tool runs in a container (no dependencies on host system)

# FM-Weck: Architecture






- Download and execute tool in container
- No knowledge of tool needed

- Download and execute tool in container
- Expert knowledge about tool required

- Spin up interactive shell in tool environment

# Conclusion

FM-Tools collects and stores essential information to:

- Generate a knowledge base about formal-methods tools [1]
  https://fm-tools.sosy-lab.org
- Conserve tool versions and their required environment
  (with help by Zenodo and Podman/Docker)
- Run a tool in conserved environment via FM-Weck [2]
- Please add your tool

https://gitlab.com/sosy-lab/benchmarking/fm-tools

# References I


[1] Beyer, D.: Find, use, and conserve tools for formal methods. In: Proc. Festschrift Podelski 65th Birthday. Springer (2024). https://www.sosy-lab.org/research/pub/2024-Podelski65.Find_Use_and_Conserve_Tools_for_Formal_Methods.pdf

[2] Beyer, D., Wachowitz, H.: FM-Weck: Containerized execution of formal-methods tools. In: Proc. FM. pp. 39–47. LNCS 14934, Springer (2024). doi:10.1007/978-3-031-71177-0_3