Version 4.5.0

Release date: March 2019

Atelier B 4.5.0 is a ***Community Edition*** version, freely downloadable on Atelier B website[1].

New functionalities:

Atelier B 4.5.0 has been released on March 2019.

This version fixes, cumulatively[2] since version 4.2.1, 146 bugs and several improvements are included:

- Simplified installation with Linux [4.5]
- Proof Obligation Generation
  - For Event-B, specific proof obligations may be generated by the new proof obligation generator: deadlock freeness, non-divergence, feasibility, coverage and exclusivity [4.5]
- Automatic proof
  - Interfacing tool « *IAPA* » (Linux only) to connect with provers from *Why3*[3] platform [4.5]
- Interactive proof
  - New feature to use SMT solvers to define proof rules applying to the current goal [4.5]
  - New feature to add its own groups of proof commands [4.5]
  - Improvement in typing of proof commands [4.5]
  - Configurable timeout added to proof commands belonging to pp family [4.4.2]
  - A new proof command added to Apply Tactic [4.4.2]
  - New functionalities in the proof tool to prove rules [4.4.2]
  - Integration of ProB model checker in the interactive prover [4.3]

[1] https://www.atelierb.eu/en/download/

[2] Through commercial releases 4.3.0 and 4.4.2

[3] http://why3.lri.fr/

- C4B code generator now supports arrays indexed by enumerated type variables [4.5]
- Editor now displays proof obligations [4.4.2]
- Coding rules are now verified in Atelier B [4.3]

New functionalities of the versions 4.4.2 and 4.3 are described in their respective version notes (release notes 4.4.2, release note 4.3).

Linux Universal installation

The Linux distribution is provided as a zip file. In order to ease installation, the file can be unzipped at any place on the hard drive.

Once unzipped, manual modification of the 3 following files is required: ***AtelierB***, ***startAB*** and ***startBB***.

They contain default paths that need to be adapted to match the targeted installation directory.

In ***AtelierB*** changes are required on the following lines:

```
ATB*ATB*AtelierB_Directory: /<installation_directory>/atelierb-free-4.5.0/
ATB*ATB*Atelier_Database_Directory: /<installation_directory>/atelierb-free-4.5.0/press/bdb
ATB*ATB*Print_Command: /<installation_directory>/atelierb-free-4.5.0/bbin/bprint
ATB*BART*RefinerFile: /<installation_directory>/atelierb-free-4.5.0/press/include/PatchRaffiner.rmf
```

In ***startAB*** changes are required on the following lines:

```
LD_LIBRARY_PATH=/<installation_directory>/atelierb-free-4.5.0/bbin/linux_x64:$LD_LIBRARY_PATH
export LD_LIBRARY_PATH
/<installation_directory>/atelierb-free-4.5.0/bbin/linux_x64/AtelierB $*
```

In ***startBB*** changes are required on the following lines:

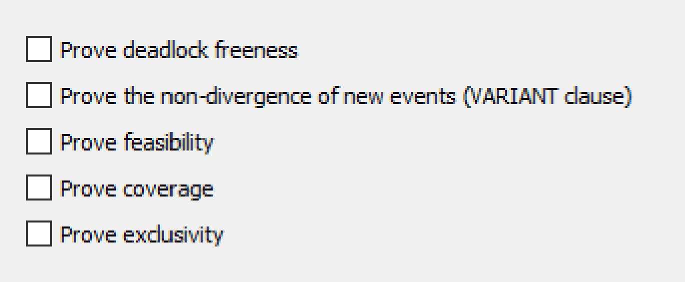
```
LD_LIBRARY_PATH=/<installation_directory>/atelierb-free-4.5.0/bbin/linux_x64:$LD_ LIBRARY_PATH
export LD_LIBRARY_PATH
/<installation_directory>/atelierb-free-4.5.0/bbin/linux_x64/bbatch -r=/<installation_directory>/atelierb-free-4.5.0/AtelierB $*
```

Proof obligation generation in Event-B

For Event-B the PO configuration file has be enriched with the following proof obligations:

- Deadlock freeness (DLF),
- non-divergence (DIV),
- feasibility (FIS),
- coverage (COV),
- exclusivity (EXC)

Their generation can be activated by ticking the project options in *preferences/system model* of the project.



□ Prove deadlock freeness

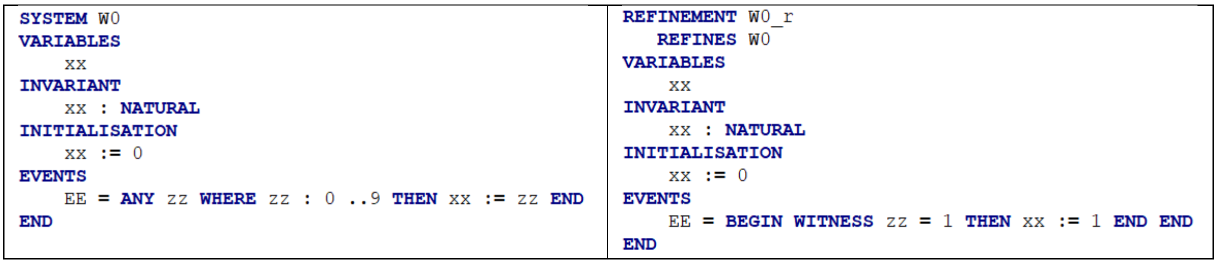
□ Prove the non-divergence of new events (VARIANT clause)

□ Prove feasibility

□ Prove coverage

□ Prove exclusivity

Moreover, the principle of witnesses (WITNESS) has been added to GOP NG. These witnesses allow to simplify non-deterministic substitution feasibility proofs by providing a value for each quantified variable (as shown in the following example).

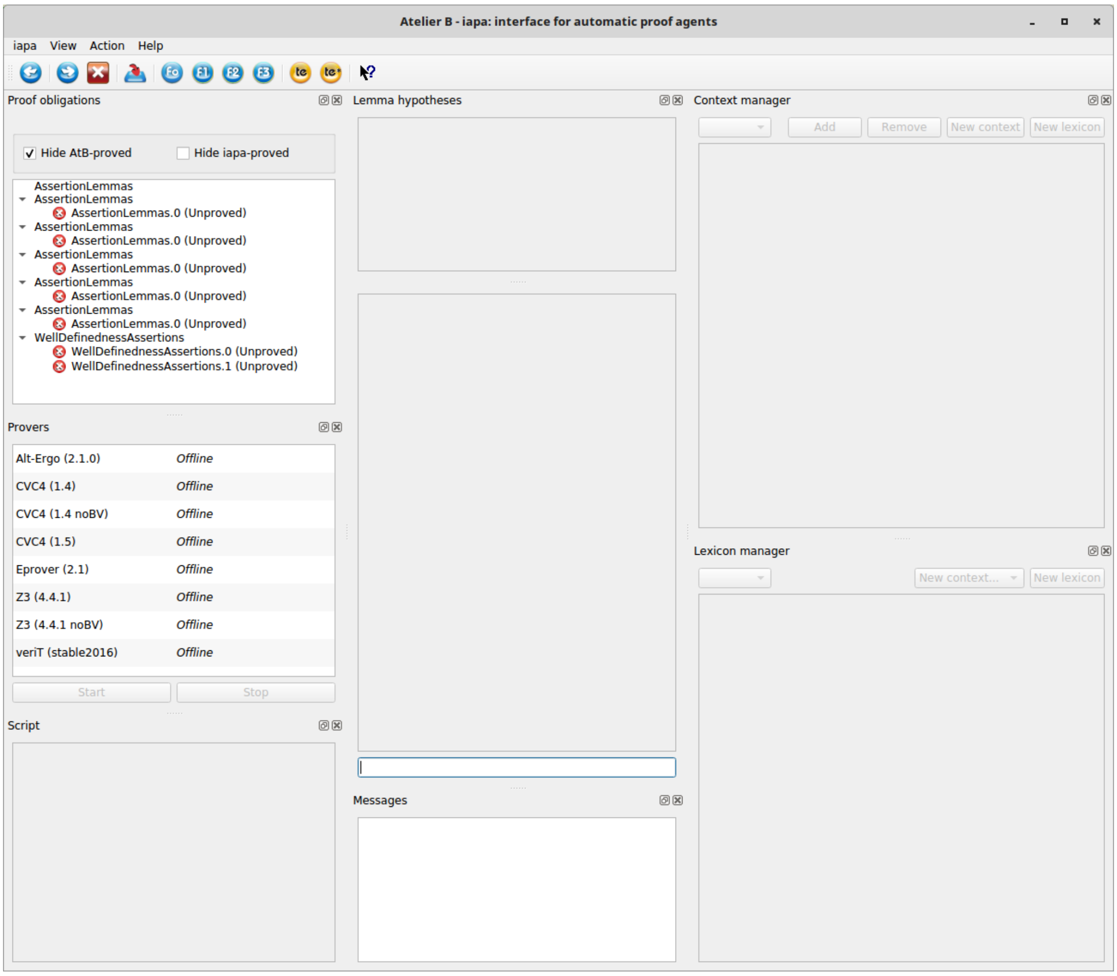


```
SYSTEM W0
VARIABLES
     xx
INVARIANT
     xx : NATURAL
INITIALISATION
     xx := 0
EVENTS
     EE = ANY zz WHERE zz : 0 ..9 THEN xx := zz END
END
```

```
REFINEMENT W0_r
    REFINES W0
VARIABLES
     xx
INVARIANT
     xx : NATURAL
INITIALISATION
     xx := 0
EVENTS
     EE = BEGIN WITNESS zz = 1 THEN xx := 1 END END
END
```

IAPA tool (Linux)

The new Atelier B module IAPA (Interface for Automatic Proof Agents), is a new interface for “almost” automatic proof based on the solvers provided by the platform Why3 (developed by the BWare project).



☑ Hide AtB-proved ☐ Hide iapa-proved

```
AssertionLemmas
▾ AssertionLemmas
    AssertionLemmas.0 (Unproved)
▾ AssertionLemmas
    AssertionLemmas.0 (Unproved)
▾ AssertionLemmas
    AssertionLemmas.0 (Unproved)
▾ AssertionLemmas
    AssertionLemmas.0 (Unproved)
▾ AssertionLemmas
    AssertionLemmas.0 (Unproved)
▾ WellDefinednessAssertions
    WellDefinednessAssertions.0 (Unproved)
    WellDefinednessAssertions.1 (Unproved)
```

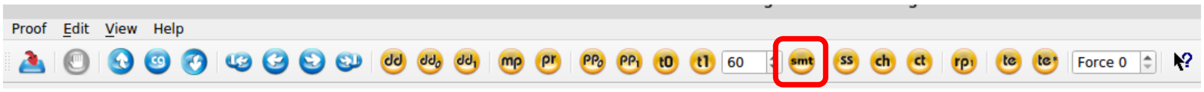
Alt-Ergo (2.1.0)	*Offline*
CVC4 (1.4)	*Offline*
CVC4 (1.4 noBV)	*Offline*
CVC4 (1.5)	*Offline*
Eprover (2.1)	*Offline*
Z3 (4.4.1)	*Offline*
Z3 (4.4.1 noBV)	*Offline*
veriT (stable2016)	*Offline*

It is « automatic » in the sense that proof obligations (PO) are automatically transmitted to the provers. However, manual selection of needed hypotheses is required. Indeed, proof obligations of real industrial project may contain many hypotheses, not all in relation with a particular PO. The large number of formulas may slow down provers.

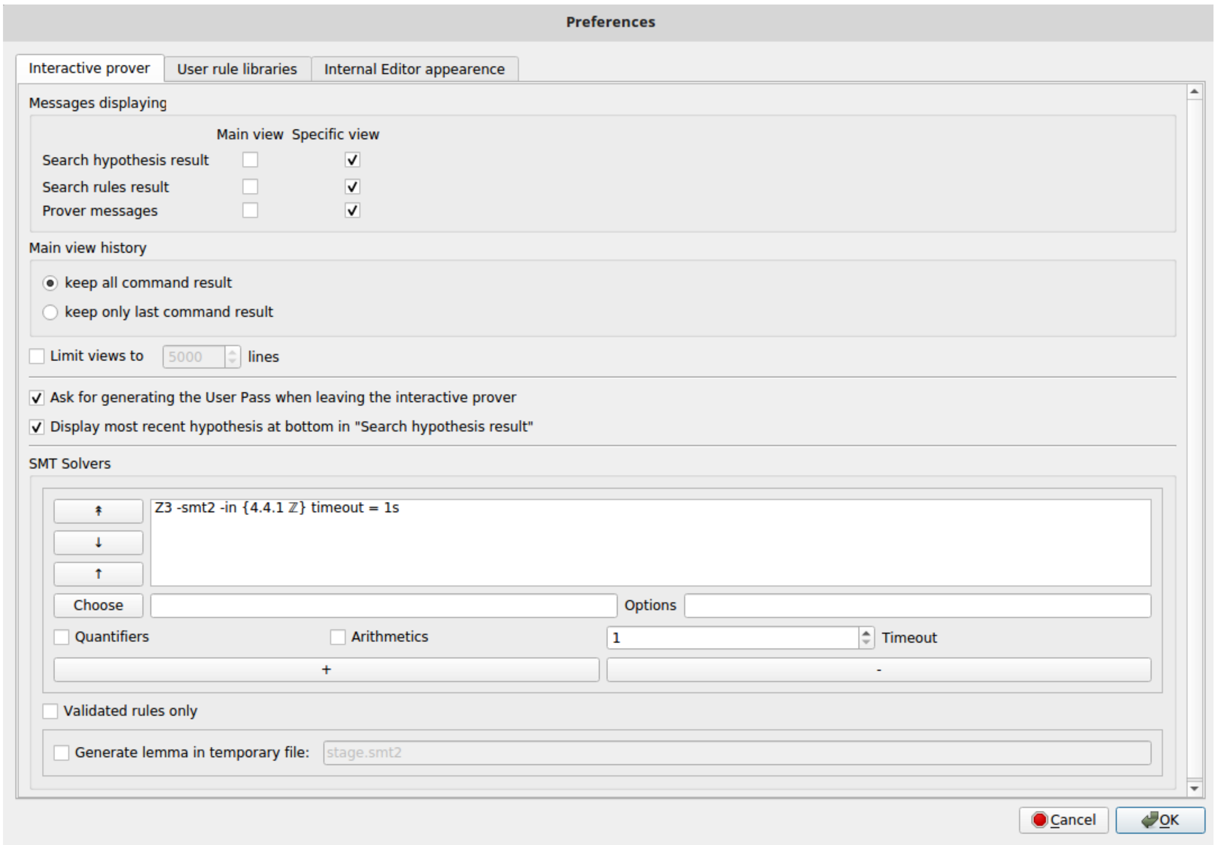
This tool is a new interactive tool (distinct from the usual tools). It presents a PO goal and hypotheses, and permits launching several provers on a selection of hypotheses related to the goal. Several mechanisms are provided in order to be able to perform this selection in a reproducible and efficient way. One can see IAPA as a tradeoff between the provided automatic tools (F0, F1, …) and the interactive prover.

The Drudges tools

The drudge functionality is a new mechanism which completes the already existing interactive prover, by automating proofs of intermediate goals



This function uses a new command that tries to prove the current goal using external (SMT) solvers. If a solver achieves to prove the goal, then a proof rule, in the form of a theorem corresponding to the current goal, is produced. This rule is saved with the other proof rules and may be used whenever necessary even on system which do not provide external SMT solvers.



Messages displaying

	Main view	Specific view
Search hypothesis result	□	☑
Search rules result	□	☑
Prover messages	□	☑

Main view history

● keep all command result

○ keep only last command result

□ Limit views to 5000 lines

☑ Ask for generating the User Pass when leaving the interactive prover

☑ Display most recent hypothesis at bottom in "Search hypothesis result"

SMT Solvers

Z3 -smt2 -in {4.4.1 ℤ} timeout = 1s

Choose | Options

□ Quantifiers □ Arithmetics 1 Timeout

+ | -

□ Validated rules only

□ Generate lemma in temporary file: stage.smt2

Cancel | OK

The main advantage of the drudges tool is that it uses the power of SMT solvers while remaining compatible with the usual Atelier B proof process. It is also possible to prove the rules produced by the drudges tool with the usual Atelier B proof tools.

In order to use external SMT solvers[4] one first needs to install a SMT solver and to configure the preferences.

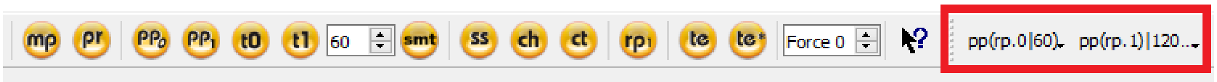
By default, propositional logic and equality reasoning are used. First order logic reasoning and integer arithmetic reasoning are optional, and may be enabled by checking “Quantifiers” and “Arithmetics” respectively.

[4] For instance, CVC4 (https://github.com/CVC4/CVC4), verit (http://www.verit-solver.org), and Z3 (https://github.com/Z3Prover/z3).

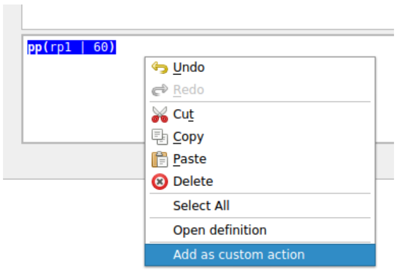
Communication with external SMT solvers is based on the standard SMT-LIB, version 2.0, through a command line call. If a solver needs parameters for this interaction, they should be filled in the "Options" text box. For instance, Z3 requires "Options" to be filled with "-smt2 -in".

Proof command configuration

A configurable tool bar has been added to the interactive proof interfaces. It is a place where users can add icons triggering arbitrary proof scripts. As an example, in the following picture, instances of PP have been added.



Such commands are added through the command editor contextual menu.


pp(rp1 | 60)

New functionalities

In addition to the major additions this Atelier B version also provides:

- A new proof command « ***fw(n)*** », to skip several commands during interactive play of a proof script. This command is also available in the proof tree contextual menu “Jump To”.
- An improvement of command typing which is more efficient due to the use of the B compiler (BComp).
- The translator “C4B” (C code generation) now handles arrays indexed by an enumerated type.