Automated transformation of Event-B machines into EB4EB deep instances*

Rémy Sangoï[1], Yamine Aït-Ameur[1], Guillaume Dupont[1], Peter Rivière[2], and Neeraj K. Singh[1]

[1] INPT-ENSEEIHT/IRIT, University of Toulouse, France
[2] Japan Advanced Institute of Science and Technology, Nomi, Japan
{yamine,gdupont,nsingh}@enseeiht.fr, remy.sangoi@toulouse-inp.fr, priviere@jaist.ac.jp

Recent advances in Event-B and its uses has led to the definition of the EB4EB framework [1], a reflexive meta-modelling of Event-B within Event-B. This framework offers the possibility to express an Event-B machine as a set of *first-order logic and set theory formulae*, and makes it possible to quantify over the constituents of an Event-B model. In turns, this enables the definition, among other things, of new custom proof obligations, or so-called *analyses*.

At its core, this framework is associated to two instantiation mechanisms, which rely on the definition of the various parts of an Event-B model in a context, as a set of algebraic equations. Although the link between said equations and the source machine is not hard to grasp, the transformation from the source Event-B model to the corresponding Event-B context is done manually, which can be tedious and error-prone.

In this proposition, we aim at mitigating this issue by providing a plug-in for the Rodin platform that effectively automates this transformation. The input of the transformation is a standard Event-B machine defined in Rodin, and the output is a corresponding context, that contains the "deep" instance (a context) of EB4EB that represents that machine.

The EB4EB Framework. The EB4EB framework [1] takes the form of a set of theories formalising Event-B models using constructive data-types and associated operators. Figure 1a presents the main data-type of the theory. Said data-type defines fields for every component of a machine. It is generic on *STATE*, the state-space of the machine, and on *EVENT*, the set of labels used for the events of the machine.

A so-called "deep" instance of the theory is created by following the template presented in Figure 1b. It essentially consists in defining each field of the `Machine` data-type, usually using set comprehensions.

The transformation plug-in. As stated previously, the input of the transformation is an Event-B machine as present in Rodin (i.e., a `.bum` file). Forming the deep instance from the existing machine requires the following main steps:

1. type each variable and aggregate the types to provide an instance to the *STATE* type parameter;

* This work was funed by the ANR-19-CE25-0010 *EBRP:EventB-Rodin-Plus* project.

```
THEORY EvtBStruc
  TYPE PARAMETERS EVENT, STATE
  DATA TYPES
    Machine(STATE, EVENT)
    constructors
      Cons_machine(
        Event : ℙ(EVENT),
        State : ℙ(STATE),
        Init : EVENT,
        Progress : ℙ(EVENT),
        AP : ℙ(STATE),
        Grd : ℙ(EVENT × STATE),
        BAP : ℙ(EVENT × (STATE × STATE)),
        Inv : ℙ(STATE),
        Thm : ℙ(STATE),
        Variant : ℙ(STATE × ℤ),
        Ordinary : ℙ(EVENT),
        Convergent : ℙ(EVENT)
      )
```

(a) Machine data-type definition

```
CONTEXT Deep
SETS Ev, ...
CONSTANTS mch, ...
AXIOMS
  axm1: partition(Ev, ...)
  axm2: mch ∈ Machine(..., Ev)
  axm3: Event(mch) = Ev
  axm4: State(mch) = ...
  axm5: Init(mch) = ...
  axm6: Progress(mch) = {...}
  axm7: AP(mch) = {...}
  axm8: Grd(mch) = {...}
  axm9: BAP(mch) = {...}
  axm10: Inv(mch) = {...}
  axm11: Thm(mch) = {...}
  axm12: Variant(mch) = {...}
  axm13: Ordinary(mch) = {...}
  axm14: Convergent(mch) = {...}
  ...
```

(b) Deep instantiation template

Fig. 1: The `Machine` data-type and its use

2. associate a label to each event and gather them to provide an instance to the *EVENT* type parameter;
3. generate the context and the equations for each field following the template of Figure 1b;
4. generate each set comprehension by aggregating the predicates located in each associated field of the machine, with the variables as the quantified variables of the comprehension (+ their primed version for the BAP).

These steps have been mechanised in the form of a plug-in for the Rodin platform. The plug-in adds an entry in the context menu associated to a machine. Clicking the entry starts the transformation and effectively generates a context (a `.buc` file) with the deep instance. It also sets up a theory path to be able to reference EB4EB's core theories. Note that the EB4EB core theories must be available in the current workspace.

Conclusion The provided plug-in enables the generation of an EB4EB deep instance from an existing Event-B machine, allowing the user to use the various tools associated to the framework on said machine.

It is to be noted that the plug-in's architecture makes it easy to "customise" it in order to generate additional elements. It also provides the capability to define and process custom *annotations* on a model, although this feature is still in early stage, and not fully functional.

The plug-in is open source and available at `https://github.com/r-sangoi/custom_po_eb4eb_rodin`.

References

1. Rivière, P., Singh, N.K., Aït-Ameur, Y.: Reflexive Event-B: Semantics and correctness, the EB4EB framework. IEEE Trans. Reliab. 73(2), 835–850 (2024), `https://doi.org/10.1109/TR.2022.3219649`