Responses to Reviewers' Comments

for paper titled
Formal verification and validation of run-to-completion style state-charts using Event-B
by
K. Morris, C. Snook, T.S.Hoang, G. Hulette, R. Armstrong, and M. Butler

Thanks We would like to thank the anonymous reviewers for their constructive comments and providing us with the chance to improve our paper. Below we summarise our major changes and address the problems and questions raised by the referees.

Reviewer #1:

```
Presentation
This paper presents an embedding of statecharts in Event-B. As such, it can
be considered as a shallow embedding of statecharts in Event-B. A strong
point of the paper is to adress through refinements the derivation of a
concrete application. The starting point being the basic semantics of the
execution model of statecharts and the last refinement being the considered
application at a given level of detail. Moreover, the paper goes beyond the
usual safety properties usually addressed by such approaches : liveness
properties are here considered.
From my point of view, the presented work is valuable. Actually, it
can be considered as emerging from the synthesis of three conference papers
[12,13,14].

Discussion
I would appreciate if the authors could elaborate their work along the
following points:
- Semantics. Statecharts have a long history. By now, they can be
considered as belonging to the family of synchronous languages. It
would be interesting if the authors could give some comments on the
choice of statecharts and especially on the relevance of the considered
semantics of run-to-completion. Otherwise stated could the authors
highlight the benefits of such a choice in general and of stachecharts
especially.
```

Response: This is a good point related to the motivation hinted at in the abstract. We added the following para in the SCXML background section.

> " State-charts, with 'run to completion' semantics, are considered to be a synchronous language in the sense that the external triggering event waits for the behaviour that it enables to complete before making any further progress. In contrast, Event-B has an asynchronous semantics due to the non-deterministic selection of events to fire. Of course, synchronous behaviour can be explicitly modelled by the addition of control variables that define the enabledness of events (i.e. remove the non-determinism). This is how we can define the translation suggested in this paper. UML-B state-machines constrain the firing of transitions to some extent but, like Event-B, do not have an underlying fully synchronous semantics. The advantage of an asynchronous semantics is its flexibility. However, when we wish to model processes that are essentially synchronous in nature, the need to explicitly add the synchronous semantics to each model becomes a burden, obscuring the particular problem being modelled. Since many components (e.g. controllers) used in a system are based on synchronous behaviour, we

are interested in adapting a modelling language with run to completion semantics to support Event-B style refinement. ”

Reviewer #1:

```
- Proof of temporal properties. Different temporal properties are con-
sidered. Since the proof of such properties are outside (currently) the
Event-B method they have considered other tools and methods.
It would have been interesting to outline the frontier. As a matter of
fact, it is not clear for me if the Event-B properties can be tackled
in temporal logic, e.g., how sequences are embedded? Could you be
precise, comment your embedding?
```

Response: The trace semantics for Event-B models have been presented in existing work [1, 2, 3]. Based on the semantics, proof obligations are generated from the Event-B models accordingly. For example, the invariant preversevation proof obligations are generated per event to ensure that the invariant holds always, i.e., $\mathsf{G}(\mathsf{I})$, where I is the invariant. Here, we consider the trace semantics with fairness assumption and reason about progress properties using the combination of deadlock-freeness and convergence event arguments (generated as proof obligations). In other words, the trace semantics of Event-B is implicit and represented by the set of proof obligations. We added the following paragraph related to various proof oblgations that we used to Section 3.2 (Event-B).

> “Proof obligations are generated to ensure the consistency of Event-B models. An important proof obligation in Event-B is invariant preservation to prove that safety properties (encoded as invariants of the models) will not be violated for any reachable states. In this paper, we also make use of other proof obligations in Event-B such as (relative) deadlock-freeness and (conditional) event convergence to construct our proof of liveness properties under some fairness assumptions.”

Reviewer #1:

```
- Proof of fairness properties. In order to deal with fairness properties
you advocate a strong fairness assumption. From my point a view, this
is a strong operating assumption. May be you could comment on that?
Last, but may be I am wrong, I have the intuition that weak fairness
for the events handling dequeuing external triggers would be enough?
Please could you comment?
```

Response: You are right, we only need to have weakness assumption on the dequeue external trigger event. We have changed the Assumption 1 to the following.

> “We assume that all internal system event `e` are strongly fair, i.e., $\mathsf{SF}(\mathsf{e})$; and the dequeue external trigger event is weakly fair, i.e., $\mathsf{WF}(\mathsf{dequeueExternalTriggered})$.”

Reviewer #1:

```
 p.2 The three rules are not at the same level. The first ones are expressed
explicitely in terms of Event-B refinement features, while the third one
addresses statecharts.
```

Response: To address other concerns we have added a ”Statechart Refinement” section to the manuscript. Please refer to Section 5 in the revised manuscript. This section now states, and illustrates the refinement rules.

Reviewer #1:

p. 10 typo? listing 3 l. 10 \emptyset

Response: We could not find the typo in Listing 3. However, we now consistently use ∅ for Listing 1.

Reviewer #1:

p.13 Could you illustrate the sentence all possible combinations of each set of transitions that can fire together are calculated and corresponding events are generated, at appropriate refinement levels.

Response: The paragraph has been revised to include an example based on the drone model.

> " Secondly, all possible combinations of each set of transitions that can fire together are calculated and corresponding events are generated, at appropriate refinement levels (given by the refinement annotations embedded in the SCXML model), that refine the abstract basis events. The transitions that can fire together are those that are triggered by the same trigger (or are both untriggered) and are in different parallel (and) sub-states. For example, the untriggered transitions shown in the parallel states FLYOP and BATTERYOP of figure 3 are combined into an event in the Event-B representation of the model, through a conjunction of the guards and actions of each of the transitions. "

Reviewer #1:

p.15 Fig. 5 is too small. One cannot read its text.

Response: The figure has been enlarged to improve readability.

Reviewer #1:

p.18 It would have been interesting to state the discussion of the first paragraph of section 7: Verification of Safety Properties within the context of the proof obligation generator you have at hand.

Response: We added the following paragraph related to various proof oblgations that we used to Section 3.2 (Event-B), in particular stating that safety properties are modelled as invariants in Event-B.

> "Proof obligations are generated to ensure the consistency of Event-B models. An important proof obligation in Event-B is invariant preservation to prove that safety properties (encoded as invariants of the models) will not be violated for any reachable states. In this paper, we also make use of other proof obligations in Event-B such as (relative) deadlock-freeness and (conditional) event convergence to construct our proof of liveness properties under some fairness assumptions."

Reviewer #1:

p.19 Could you precise your notion of run.

Response: Run here refers to different simulation paths that can be generated to explore systems behaviors. We have revised the text as shown below

> "System events: Events other than external events are called system events. They are the events by which the system responds to the external triggers (by creating different runs or simulation paths). "

Reviewer #1:

`p. 20 typo. We are now present`

Response: The text has been revised as follow:

> "We now present a theorem and a corollary related to (relative) deadlock-freeness properties for a different set of events."

Reviewer #1:

`p. 21 Could you give a formal definition or at least a reference of your strong fairness.`

Response: We added the following paragraph to Section 3.2 (Event-B).

> " For the trace semantics corresponding to Event-B machines and the intepreation of LTL properties over traces, we refer the readers to [2]. Here, we recall the notation for fairness assumptions underlying event-based formalisms such as Event-B [5, 3]. Given an event e , a weak-fairness assumption $\mathsf{WF}(\mathsf{e})$ states that if e is enabled continually, then it must occur infinitely often. Similarly, a strong-fairness assumption $\mathsf{SF}(\mathsf{e})$ states that if e is enabled infinitely often, then it must occur infinitely often. Formally,
>
> $$\mathsf{WF}(\mathsf{a}) \Leftrightarrow (\mathsf{FG}\,\mathsf{enabled}(\mathsf{e}) \Rightarrow \mathsf{GF}\,[\mathsf{e}]) \text{ , and}$$
> $$\mathsf{SF}(\mathsf{a}) \Leftrightarrow (\mathsf{GF}\,\mathsf{enabled}(\mathsf{e}) \Rightarrow \mathsf{GF}\,[\mathsf{e}]) \text{ ,}$$
>
> where G and F are the temporal operators denoting *globally* and *finally*, respectively; and $\mathsf{enabled}(\mathsf{e})$ denotes that event e is enabled and $[\mathsf{e}]$ denotes an occurance of event e . "

Reviewer #1:

`p.22 Could you comment on your definition of anticipated events. Why the set of convergent events is necessary to recall just before?`

Response: We added the following sentence after the definition for anticipated events.

> "Note that the anticipated events augment the set of convergent events and respect the variant that are used to prove the convergence property."

Reviewer #1:

`p.22 Proof of Convergence an Anticipation I wonder if this paragraph should not be before because you use such arguments before just after stating Theorem 2.`

Response: We highlight the fact that we will prove the convergence and anticipation property later by adding the following sentence after Theorem 2.

> "Note that Theorem 2 relies on convergence of internal events and anticipation of external events, which we will prove later. "

Reviewer #1:

`p.23 typo. it will be dequeued.`

Response: The text has been revised as follow:

" If an external trigger is raised, then eventually, it will be dequeued."

Reviewer #1:

`p. 23 could you explain the square bracket notation, e.g. [externalTrigger.t]`

Response: We explain the [] notation as part of the explanation about fairness assumptions (newly added). The explanation of the "dot" notation is in Theorem 3, that is [e.t] indicates an occurance of event e with parameter value t.

> "where G and F are the temporal operators denoting *globally* and *finally*, respectively; and enabled(e) denotes that event e is enabled and [e] denotes an occurance of event e . "

Reviewer #1:

`typo. they do no hold a priori.`

Response: The text has been revised as follow:

> "These proof will need to be done for each individual SCXML state-chart as they do not hold a priori."

Reviewer #1:

`typo. relying on lexicographic order . . .`

Response: The text has been revised as follow:

> "We present a generic approach to reason about the proof of convergence and anticipation relying on lexicographic order as follow."

Reviewer #1:

```
 The paragraph Proof of Convergence and Anticipation needs to
be written again. There are many typos: this event removes, discards,
decreases, accroding, . . .
Moreover the sentence
The external events are anticipated accroding to the above variants triv-
ially since they only modify the external queue eQ. Note that we do not
attempt to prove the convergence of any future events here. Instead, we
assume that these future events will be prove to be convergence later.
seems to me problematic. In your definition of anticipated you did not
say that these events should be proven convergent later?
```

Response:

Karla: →**Son** Could you please take a look just to make sure I did not introduced mistakes with my changes

The entire Proof of Convergence and Anticipation subsection has been revised. Please refer to the provided manuscript.

Reviewer #1:

p.25 could you state explicitly your strong fairness property and the interplay with the temporal properties you are concerned with.

Response: The definition of fairness assumptions is now added in the new paragraph in Section 3.2 (Event-B). The specific (strong) fairness assumption that we made is in Assumption 1 (Strongly Fair System Events). This fairness assumption is used in the proof for Theorem 2 and Theorem 3 to prove the termination of repsonses for external triggers.

Reviewer #1:

p.25 A reference to the seminal Unless of Unity could be in order.

Response: We have added a reference to "Chandy, M., Misra, J.: Parallel program design - a foundation. Addison-Wesley (1989)" when discussing Unless property.

Reviewer #1:

p. 26 (Theorem 5) I think that the indexes in eQ should be first stated as legal in both quantifications.

Response: We added the condition that both indexes i, j must be in the domain of eQ, i.e., $\mathsf{i} \in \mathsf{dom}(\mathsf{eQ})$ and $\mathsf{j} \in \mathsf{dom}(\mathsf{eQ})$.

Reviewer #1:

p. 27 All the temporal proofs have been done in an adhoc way without any tool support. It would be interesting to have a feedback about this? To be provocative, if you are interested in temporal proofs why did you choose this tool? Have you considered TLA which does support temporal proofs (as well as refinements in a certain way)?

Response: As explained earlier, the Event-B approaches to proving temporal properties is implicit via proof obligations. In TLA, the proving of temporal properties is more explicit compared to Event-B. The last time we check the TLA+ Proof System, it does not support reasoning about many temporal operators (http://tla.msr-inria.inria.fr/tlaps/content/Documentation/Unsupported_features.html). We added the following paragraph to the related work section.

> "A method that is closely related to Event-B and also supports reasoning about safety and liveness properties is TLA+ [6]. TLA+ is supported by the TLA+ Toolbox [4]. On the one hand, temporal properties (both safety and liveness) are *explicitly* stated as properties of the TLA+ models and reasoning about them often requires applying proof rules related to properties of traces. On the other hand, Event-B defines proof obligations based on the underlying trace semantics [1, 2, 3], hence reasoning about *implicitly* temporal properties in Event-B simply involves discharging the relevant proof obligations. Furthermore, at the time of writing, the TLA+ Proof System (part of the TLA+ Toolbox) does not fully support the reasoning with many temporal operators.[1]"

Reviewer #1:

It would be interesting to analyze if the proofs are specific to the example or to the underlying semantics?

Response: We added the following sentence to explain which variants are generic.

[1] http://tla.msr-inria.inria.fr/tlaps/content/Documentation/Unsupported_features.html (accessed June 2021)

"Note that except the variant related to the internally triggered events and untriggered events, i.e., (2), all other variants, i.e., $V_{externalTrigger}$, $V_{dequeueInternalTriggered}$, $V_{noTriggeredTransitionsEnabled}$, and $V_{noUntriggeredTranstitionsEnabled}$ are generic according to the underlying run-to-completion semantics."

Reviewer #1:

```
PS Could you put another zip version on the repository: I have had some
strange problems (missing characters) with some machine files? For such
files, I was not able to play again the proofs.
```

Response:

Son: To put the Rodin configuration as README file. Probably use the Soton Bundle 2012.

Reviewer #2:

```
The paper introduces a technique for the refinement of 'run to completion'
statechart modelling notation (using SCXML language) while preserving safety
properties. The statechart specification is translated to event-B formalism,
allowing for formal verification using a theorem prover. The proposed approach
is demonstrated using a statechart specification of a drone.

Positive points:
+ Interesting topic
+ Technique well motivated
+ The paper is well written and easy to read.

Negative points:
- One single case study is not enough to validate the proposed approach. The
statechart specification of the drone is rather small. More elaborated models
are required to validate the proposed approach.
```

Response: Although we agree with the reviewers point of view that a single case study is not enough for a full evaluation of the approach. We want to emphasize the fact we have describe other model in previous publication. We have clearly added references to this other models in the example section. The case studies have grown in complexity and the drone in particular makes use of all the features for model construction and refinement.

Reviewer #2:

```
General comments:
  - The three refinement rules listed in the introduction have not been
  described explicitly in the rest of the paper. Please describe them (using
  minimal examples) in section 3.
```

Response: A new section has been added to state and discuss the refinement rules. Please refer to section 5 in the revised manuscript.

Reviewer #2:

```
  - In the introduction, the paragraph before last "Page 3: lines 7 to 17" that
  compares the proposed approach to the work presented in [4] may be pushed to
  Section 3 or 4 since such comparison is meaningless before presenting the
  details of the approach and the example.
```

Response: As suggested by the reviewer the paragraph has been moved to section 4.

Reviewer #2:

```
 - The paper lacks a related work section. Please add one.
```

Response: A related work section has been added to the manuscript. Please see section 2

Reviewer #2:

```
Minor points:
  - After an introductory word or phrase, use a comma (this is a recurrent in
  the paper). For example, e.g. --> e.g., i.e. --> i.e., "To verify liveness
  we outline" --> "To verify liveness,  we outline", etc.
```

Response: The requested changes have been completed.

Reviewer #2:

```
- Abstract: "We introduce" --> "In this paper, we introduce".
```

Response: The text has been revised as follow:

> " In this paper, we introduce a notion of refinement into a run to completion state-chart modelling notation, and leverage Event-Bs tool support for theorem proving. "

Reviewer #2:

```
- Add "Even-B" to the list of keywords.
```

Response: The requested change has been completed.

Reviewer #2:

```
- Page 2: line 5: "Particularly attractive is providing" --> "Particularly
attractive in providing"
```

Response: The text has been revised as follow:

> "It is particularly attractive, to provide accessibility to abstraction/refinement via Rodin/Event-B which has an intuitive metaphor in the Statechart semantics [12,14,13]. "

Reviewer #2:

```
- Page 2: line 10: "safety preservation" --> "safety properties preservation"
```

Response: The requested change has been completed.

Reviewer #2:

```
- Page 2: line 20: "Preservation of safety" --> "Preservation of safety
properties"
```

Response: The requested change has been completed.

Reviewer #2:

```
- Page 2: line 45: "in the sense of [9]" --> "in the sense adopted by Lamport [9]"
```

Response: The requested change has been completed.

References

[1] J-R. Abrial. *Modeling in Event-B: System and Software Engineering*. Cambridge University Press, 2010.

[2] Thai Son Hoang, Steve Schneider, Helen Treharne, and David Williams. Foundations for using linear temporal logic in event-b refinement. *Formal Aspects of Computing*, 28, 04 2016.

[3] Simon Hudon, Thai Son Hoang, and Jonathan S. Ostroff. The Unit-B method — refinement guided by progress concerns. *Software and System Modeling*, 15(4):1091–1116, October 2016.

[4] Markus Alexander Kuppe, Leslie Lamport, and Daniel Ricketts. The TLA+ toolbox. In Rosemary Monahan, Virgile Prevosto, and José Proença, editors, *Proceedings Fifth Workshop on Formal Integrated Development Environment, F-IDE@FM 2019, Porto, Portugal, 7th October 2019*, volume 310 of *EPTCS*, pages 50–62, 2019.

[5] Leslie Lamport. Proving the correctness of multiprocess programs. *IEEE Transactions on Software Engineering SE-3*, 2:125–143, March 1977.

[6] Leslie Lamport. *Specifying Systems, The TLA+ Language and Tools for Hardware and Software Engineers*. Addison-Wesley, 2002.