The CamilleX Framework for the Rodin Platform

Thai Son Hoang[0000−0003−4095−0732], Colin Snook[0000−0002−0210−0983], Dana Dghaym[0000−0002−2196−2749], Asieh Salehi Fathabadi[0000−0002−0508−3066], and Michael Butler[0000−0003−4642−5373]

ECS, University of Southampton, Southampton, U.K.
{t.s.hoang, cfs, d.dghaym, a.salehi-fathabadi, m.j.butler}@soton.ac.uk

Abstract. We present the CamilleX framework for the Rodin platform in this paper. The framework provides a textual representation and persistence for the Event-B modelling constructs. It supports direct extensions to the Event-B syntax such as machine inclusion and record structures, as well as indirect extensions provided by other plugins such as UML-B diagrams. We discusses CamilleX's design and in particular, its extension mechanisms and examples of their use.
Keywords: Event-B, Rodin Platform, XText, CamilleX

1 Introduction and Motivation

The Event-B modelling method [1] is a discrete state-transition formal modelling language. The main supporting tool for Event-B is the Rodin Platform (Rodin) [2], which facilitates the editing of Event-B models and reasoning about them. Rodin is based on Eclipse and provides an extensible platform via Eclipse's plug-in mechanism. This is very important for the openness approach to both Event-B as the modelling method and to the supporting Rodin [11].

Rodin [2] is a supporting platform for Event-B and is developed in Eclipse. One of the main components of the Rodin Core is the Rodin repository. In particular, the Rodin repository stores the elements in a tree-shaped structured database and does not make any assumptions about the elements stored in the repository. The internal structure of the model repository (called the 'Rodin repository') was influenced by the choice of using Eclipse as the underlying basis for Rodin [11]. Essentially, an Event-B model in Rodin is a collection of modelling elements. The 'syntax' using keywords that users see in the GUI is provided by the corresponding editors and does not exist in the persisted model. As the structure is independent of the Event-B modelling language, it makes extending Event-B straight-forward. Another important component of the Rodin Core is the Rodin builder which runs the Event-B core tools automatically. While the design of the Rodin repository aids extensibility, it has other consequences. The models are persisted in XML files which makes it difficult for humans to read and understand the model. It is difficult, for example, to compare two versions of a model when using version control tools for collaboration. Moreover, it is

challenging to develop a functional modelling user interface, in particular, the editor for the XML files.

Our motivation is to have a true, human readable, text-based persistence of Event-B models which overcomes the limitations of the current modelling user interface.

2 Background

This section provides background information about the CamilleX-relevant technology, namely, the Eclipse Modelling Framework (EMF) and XText.

EMF [10] is an Eclipse-based framework for implementing modelling languages. An abstract syntax is defined by a meta-model and code is then generated to provide a repository for instances of the model. In previous work [9] we have implemented EMF tooling for Event-B models with persistence into the Rodin repository. Many of our plug-ins, including UML-B, are based on this Event-B EMF framework and utilise the extension mechanism that we built into it. The basis of this inheritance structure is the generic meta-class, *EventBElement* which provides facilities for extending the meta-model with new features. The most important of these is the *extensions* containment of *AbstractExtension*. Since this is inherited by all other model element classes, an extension containment can be defined for any kind of concrete model element by sub-classing *AbstractExtension* and providing support for persistence, processing and translation as required. The CamilleX tools described herein are based on this EMF meta-model and make use of its extension mechanism, both for syntactic extensions to the modelling language as well as to support model contributions provided by other plugins.

XText [3] is a powerful framework for developing programming languages and domain-specific languages. The input to the framework is a grammar describing the input language and the result of the framework tooling is "a full infrastructure, including parser, linker, typechecker, compiler as well as editing support for Eclipse" [7]. In particular, the editing support generated from XText includes features such as content assist and customisable framework for validation and code generation. Internally XText relies on EMF, e.g., for loading the in-memory representation of any parsed text files. This enables XText models to be used by any other EMF-based tools since the XText grammar can be seen as 'just' an alternative persistence for EMF models.

3 CamilleX

The main aim of the CamilleX framework is to provide text-based serialisation of Event-B models. Furthermore given the existing facilities for Event-B in Rodin, we have the following design principles for CamilleX.

– Reuse the existing Event-B tools of Rodin as much as possible.

– Support direct extension of the Event-B syntax to provide additional features.
– Provide compatibility with other kinds of 'higher-level' models that contribute to the overall model, e.g., UML-B diagrams [8].

Section 3.1 gives an overview of the basic design for the CamilleX framework. We will discuss direct extensions to the Event-B syntax in Section 3.2 and indirect extension by plug-ins to contain other kinds of models in Section 3.3.

3.1 The Basic Design

CamilleX supports two types of textual files XMachine and XContext, which in turn will be automatically translated to the corresponding Rodin Event-B components (machine and context). The reverse transformation from Event-B to CamilleX is also supported and can be invoked manually as shown in Figure 1. Note that the representation of CamilleX constructs (XMachines and

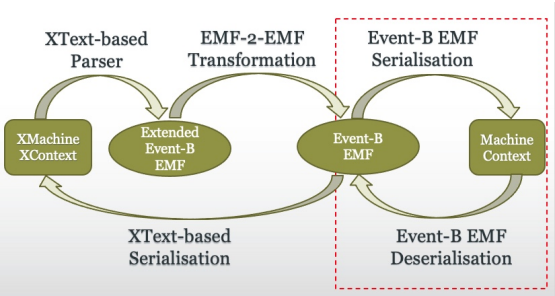

Fig. 1: Overview of CamilleX and Rodin Event-B Constructs

XContexts), uses an extended Event-B EMF to accommodates Event-B syntax extensions (e.g., machine inclusion and records structure) which are 'flattened' into the (core) Event-B EMF during the automatic translation.

Essentially, CamilleX provides the "outer" syntax to Event-B models while relying on the Event-B static checker to check the "inner" syntax of Event-B (i.e., Event-B mathematical formulae).

An important difference between the syntax of CamilleX and that of Camille is that CamilleX supports comments "everywhere". Since Camille relies directly on the structure of the underlying XML serialisation, it can only accept comments attached to the individual modelling elements. For CamilleX, comments can appear anywhere in the textual representation of the Event-B models and are ignored (i.e. omitted) during the translation to Rodin Event-B constructs.

As we rely on the Event-B static checker for checking the inner syntax of the Event-B models, we implemented a callback mechanism to report any errors and warnings raised by the Rodin static checker back to the CamilleX constructs.

3.2 Direct Extensions to the Event-B Syntax

In this section, we present two extensions of the CamilleX constructs to support machine inclusion [6] and records structure [5]. The steps for extending CamilleX are as follows.

1. Extend the Event-B EMF with the new modelling element(s).
2. Extend the grammar of the CamilleX construct and regenerate the supporting tools.
3. Extend the CamilleX validator to ensure the consistency of the added modelling elements.
4. Extend the CamilleX generator to translate the newly added modelling elements into standard Event-B in the model output to Rodin.

Machine Inclusion The machine inclusion extension provides the concepts for a machine to include other machines and for an event to synchronise with one or more events from the (different) included machines. The details of the mechanism are described in [6]. We summarise the main ideas of for a machine A that includes a machine B below:

- A inherits all variables and invariants of B.
- B's variables can only be modified from A via *synchronising* with an event of B.
- Multiple instances of B can be included via *prefixing* and in that case, B's variables and events are renamed accordingly.

The CamilleX generator is extended to "flatten" the machine inclusion and event synchronisation into standard Event-B EMF before serialisation into Rodin Event-B constructs.

- For each **includes** clause of a machine, the translator copies the variables and invariants from the included machine. If the included machine is prefixed, multiple copies of the variables and invariants are generated and renamed accordingly.
- For each **synchronises** clause of an event, the translator copies the content of the included events (i.e., the parameters, guards and actions) and renames them appropriately if the included machine is prefixed.

Record Structures The records extension provides the ability to use record structures within Event-B machines and contexts. The record structures and their translation to Event-B are described in [5]. Records can be declared in both contexts and machines and will generate different Event-B modelling elements in each. The CamilleX generator is extended to "flatten" the records into standard Event-B EMF before serialisation into Rodin Event-B constructs. A record in a context will generate either a carrier set or (if it is an inherited record) a constant. The fields of a record in a context will be generated as constants with the appropriate type, depending on the multiplicities, **one** (total functions), **many** (binary relations), or **opt** (partial functions). A record in a machine must inherit

another record, and is generates a variable of the machine. The fields of a record in a machine will also be generated as variables with the appropriate type in the machine (depending on the record's multiplicity).

3.3 Indirect Extensions by Plug-ins

In the previous section, we showed how to directly extend the syntax of the CamilleX constructs, i.e. XContexts and XMachines to support mechanisms such as machine inclusion and record structures. In this section, we describe a generic extensible mechanism for integration with other plug-ins such as UML-B.

We introduce the notion of *containment*, to enable XContexts and XMachine to include external components such as UML-B diagrams. We introduce the contains clauses which references a DiagramOwners. Each DiagramOwner contains zero or more Diagrams which will contribute to the containing Machines or Contexts. The abstract meta-class, Diagram, can then be sub-classed to contribute the specific desired model syntax. For example, the UML-B diagram types, statemachine and classdiagram, both extend Diagram.

An extension point is created for the CamilleX generator, which allows plug-ins to contribute an implementation of how the contained components are translated in order to contribute to the Event-B models. The CamilleX generator will then defer to this translation for the specific type of contained components declared in the extension, e.g., UML-B state-machines or class diagrams.

4 Conclusion and Future Work

This paper presents the CamilleX framework which provides textual serialisation of Event-B models. In particular, we reuse the existing Event-B tool-chain of Rodin, by providing only the "outer" syntax for the Event-B models. The design of CamilleX supports both direct extension to the Event-B syntax and indirect extensions by plug-ins to contain other types of components such as UML-B diagrams. Our experience shows that CamilleX improves the usability of Rodin and assists users in developing Event-B models.

Future work on machine inclusion will suppress the generation of unnecessary proof obligations (e.g., those that are related to included invariants are correct-by-construction), add support for importing a refinement-chain (instead of individual machines), and integrate with context instantiation.

Currently CamilleX does not fully support the extension refinement of record structures as described in [5]. At the moment, properties of the record fields are translated as axioms and invariants after other "normal" axioms and invariants. Sometimes we need to rearrange the order. For example, the generated elements need to go before or in between the other normal elements. In order to provide more flexible ordering of elements, the Event-B EMF will be restructured to have a single collection of child elements.

Support for reasoning about availability properties with the notion of rigid events and parameters [4] can be also added to CamilleX.

Although the CamilleX containment extension allows for integration with UML-B, the UML-B diagrams are currently persisted in EMF XMI format. For similar reasons to CamilleX, it would be advantageous to have a human-readable text persistence for UML-B diagrams. We are therefore developing XUML-B, which will provide an XText persistence for UML-B.

Acknowledgement

This work is supported by the following projects:

- HiClass project (113213), which is part of the ATI Programme, a joint Government and industry investment to maintain and grow the UK's competitive position in civil aerospace design and manufacture.
- HD-Sec project, which was funded by the Digital Security by Design (DSbD) Programme delivered by UKRI to support the DSbD ecosystem.

References

1. J-R. Abrial. *Modeling in Event-B: System and Software Engineering*. Cambridge University Press, 2010.
2. J-R Abrial, M. Butler, S. Hallerstede, T.S. Hoang, F. Mehta, and L. Voisin. Rodin: An open toolset for modelling and reasoning in Event-B. *Software Tools for Technology Transfer*, 12(6):447–466, 2010.
3. Lorenzo Bettini. *Implementing Domain-Specific Languages with Xtext and Xtend*. Packt Publishing, second edition edition, 2016.
4. Dana Dghaym, Thai Son Hoang, Michael Butler, Runshan Hu, Leonardo Aniello, and Vladimiro Sassone. Verifying system-level security of a smart ballot box. In *ABZ 2021*, 2021.
5. Asieh Salehi Fathabadi, Colin Snook, Thai Son Hoang, Dana Dghaym, and Michael Butler. Extensible record structures in Event-B. In *ABZ 2021*, 2021.
6. Thai Son Hoang, Dana Dghaym, Colin F. Snook, and Michael J. Butler. A composition mechanism for refinement-based methods. In *22nd International Conference on Engineering of Complex Computer Systems, ICECCS 2017, Fukuoka, Japan, November 5-8, 2017*, pages 100–109. IEEE Computer Society, 2017.
7. The XText Project. XText website, 2020. `https://www.eclipse.org/Xtext/`.
8. Mar Yah Said, Michael J. Butler, and Colin F. Snook. Language and tool support for class and state machine refinement in UML-B. In Ana Cavalcanti and Dennis Dams, editors, *FM 2009*, volume 5850 of *Lecture Notes in Computer Science*, pages 579–595. Springer, 2009.
9. Colin Snook, Fabian Fritz, and Alexei Iliasov. Event-B and Rodin Documentation Wiki: EMF Framework for Event-B. `http://wiki.event-b.org/index.php/EMF_framework_for_Event-B`, 2009. Accessed May 2020.
10. D. Steinberg, F. Budinsky, M. Paternostro, and Ed Merks. *Eclipse Modeling Framework*. The Eclipse Series. Addison-Wesley Professional, 2nd edition, 2008.
11. Laurent Voisin and Jean-Raymond Abrial. The Rodin platform has turned ten. In Yamine Aït Ameur and Klaus-Dieter Schewe, editors, *Abstract State Machines, Alloy, B, TLA, VDM, and Z - 4th International Conference, ABZ 2014, Toulouse, France, June 2-6, 2014. Proceedings*, volume 8477 of *Lecture Notes in Computer Science*, pages 1–8. Springer, 2014.