HAL
open science

A System Development Process with Event-B and the Rodin Platform

Jean-Raymond Abrial

To cite this version:

Jean-Raymond Abrial. A System Development Process with Event-B and the Rodin Platform. Embedded Real Time Software and Systems (ERTS2008), Jan 2008, Toulouse, France. hal-02270340

HAL Id: hal-02270340

https://hal.science/hal-02270340v1

Submitted on 24 Aug 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A System Development Process with Event-B and the Rodin Platform

Jean-Raymond Abrial

Swiss Federal Institute of Technology, Zurich (Switzerland)

jabrial@inf.ethz.ch

ABSTRACT

The aim of this presentation is to show that the Event-B technology can be put into practice. For this, we must follow a well defined development process. That process is precisely the one which has to be transferred to industry. The Rodin platform, in its final form, will be the supporting tool for achieving this. Before describing the Event-B development process however, we make precise what we mean by an industrial development process in general terms.

Keywords

Formal method, Event-B, Industrial process, Modelling tool

1. EVENT-B

Event-B [2] is the name of a mathematical (set-theoretic) approach used to develop *discrete system models*, be they computerized or not. It is the successor of the B Method [1], which has been used in various operational industrial projects [7] [6]. Event-B has been strongly influenced by the work of the Finnish School on Action System [5] [8].

It has been found over the years that a *unique paradigm*, that of discrete transition systems, was the common denominator to apparently very different computation mechanisms: sequential, distributed, concurrent, parallel, etc.

2. THE RODIN PLATFORM

The Rodin Platform [3] [4] [9] [11] is an open tool set implemented on top of Eclipse [10]. It is devoted to supporting the development of such systems. It has been developed within the framework of the European project Rodin [12].

It contains a modeling database surrounded by various plug-ins: static checker, proof obligation generator, provers, model-checkers, animators, UML transformers, requirement document handler, etc. The database itself contains the various modeling elements needed to construct discrete transition system models: essentially variables, invariants, and transitions.

3. FORMAL DEVELOPMENT

With the help of this palette, users can develop mathematical models and refine them. In doing so, they are able to reason, modify, and decompose their models before starting the effective implementation of the corresponding systems. Such an approach is well known and widely used in many mature engineering disciplines where reasoning on a abstract representation of the future system is routine. Just think of the usage of blueprints made by architects within a building construction process.

4. TECHNOLOGY TRANSFER

One of the main difficulties in transferring this technology is not that of its mastering by industry engineers (a common opinion shared by many analysts). It is rather, we think, the incorporation of this technology within the industrial *development process*. We believe that the above argument about the difficulty of mastering this technology is, in fact, a way of hiding (consciously or not) the one concerning the incorporation within the development process.

5. INDUSTRIAL PROCESS

It is now common practice among important critical system manufacturers (train signalling system companies, avionic and space companies, automotive manufacturers, power system designers, defense sector industries, etc.) to embark in important projects within the framework of a well-defined development process. Such a process contains the definition of the various milestones encountered in the system construction together with the precise definition of what is to be done between these milestones, by whom, and within which delays. It also contains different ways of re-iterating on these milestones in case the process encounters some difficulties.

Usually, industrial managers are very reluctant to modify their development processes because: (1) it is part of their company culture and image, (2) it is difficult to define and make precise, and (3) it is even more difficult to have them accepted and followed by working engineers.

In order to know how to modify the development process due to the introduction of some formal method technology (Event-B and Rodin) in the construction of complex systems, it is clearly very important to understand that this process is aimed at obtaining systems which can be considered to be *correct by construction*. This presentation does not pretend to solve all related problems nor to give the key to a successful incorporation of formal methods in industry: it aims at providing the beginning of a systematic way of envisaging these matters.

Let us now briefly present the Event-B development process and show how the Rodin platform supports it.

6. THE REQUIREMENT DOCUMENT

After the initial feasibility studies phase which is not subsequently modified, the second phase of the process is the

writing of the *requirement document*. It must be pointed out that most of the time such documents are very poor: quite often, they just contain the pseudo-solution of *a problem which, to begin with, is not stated*. Our opinion is that it is very risky to proceed further with such poor documents. More precisely, we think that it is necessary to rewrite them very carefully in a systematic fashion. Each requirement must be stated by means of a short English statement which is well recognizable and clearly labelled according to some taxonomy to be defined for each project.

The Rodin platform in its final form will provide a plug-in able to support the gradual construction of such structured requirement documents, to retrieve them, and to form the initial basis of the necessary traceability.

7. THE REFINEMENT STRATEGY

The next phase consists in defining a temporary *refinement strategy*. It contains the successive steps of the refined models construction. Clearly, it is out of the question to construct a unique model taking account of all requirements at once. Each such refinement step must give a reference to the precise requirements, stated in the previous phase, which are taken into account. A preliminary completion study can be performed in order to ensure that no requirements are forgotten. The refinement strategy in this phase is only temporary as it might be reshaped in further phases.

The Rodin platform in its final form will provide a plug-in able to support the writing of the refinement strategy and to check that it is correctly linked to the requirement document.

8. REFINEMENTS AND PROOFS

The next phase is divided up in many sub-steps according to the precise strategy defined in the previous phase. Each sub-step is made of the definition of the *formal refinement* which is performed, together with the corresponding *proofs*. It might be accompanied by some model-checking, model testing, as well as animations activities.

The three previous activities are very important to be performed at each refinement sub-step as they help figuring out that some requirements are impossible to achieve (or very costly), whereas some other had been simply completely forgotten. In other words, these activities help *validating the requirement document*. The outcome of these activities (checked or tested properties and model animations) can be seen and understood by the "client", who is then able to judge whether what has been formally modeled at a given stage indeed corresponds to what he had in mind. Notice that in each refinement sub-step, it might be found also that the previous refinement strategy was not adequate so that it has to be modified accordingly.

The Rodin platform provides the core elements able to support this central phase: modeling database, proof obligation generator, and provers. The surrounding plug-ins (model-checker, animator, UML translator) support the other requirement document validation activities

9. DECOMPOSITION

The next phase proceeds with the *decomposition* of the refined model obtained at the end of the previous one. In particular, this decomposition might separate that part of the model dealing with the external environment from that part of the system dealing with the hardware or software implementation. The latter part might be refined in the same way as it was done on the global model in the previous phase. This refinement/decomposition pair might be repeated a number of times until one reaches a satisfactory architecture.

The Rodin platform in its final form will provide plug-ins to support and prove that proposed decompositions are correct.

10. CODE GENERATION

The final phase consists in performing the various hardware or software *automatic code generation*. The Rodin platform in its final form will provide plug-ins to perform these translations.

11. CONCLUSION

As can be seen the incorporation of these phases within an existing development process is certainly not an easy task. An important point to take into account is the incorporation (and measurement) of the many proofs which have to be performed in order to be sure that the final system will be indeed "correct by construction". The Rodin project, as described here, will be extended with a new IP European FP7 Project named DEPLOY starting in February 2008.

12. REFERENCES

[1] J.-R. Abrial. *The B-Book: Assigning Programs to Meanings*. Cambridge University Press, 1996.

[2] J.-R. Abrial. *Modelling in Event-B: System and Software Engineering*. To be published by Cambridge University Press, 2008.

[3] J.-R. Abrial. *Tools for Constructing Large Systems (a proposal)*. In *Rigorous Development of Complex Fault-Tolerant Systems*. M. Butler, etc. (Eds). LNCS 4157 Springer, 2006

[4] J.-R. Abrial, M. Butler, S. Hallerstede, L. Voisin. *An Open Extensible Tool Environment for Event-B*. ICFEM 2006

[5] R. Back. *Decentralization of process nets with centralized control*. 2nd ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing, 1983.

[6] F. Badeau. *Using B as a high level programming language in an industrial project: Roissy val*. Proceedings of ZB'05, 2005.

[7] P. Behm. *Meteor: A successful application of B in a large project*. Proceedings of FM'99, 1999.

[8] M.J. Butler. *Stepwise Refinement of Communicating Systems*. Science of Computer Programming, 1996

[9] M.J. Butler and S. Hallerstede *The Rodin Formal Modelling tool*. BCS-FACS Christmas 2007 Meeting Formal methods in Industry London, 2007

[10] *Eclipse*. http://www.eclipse.org

[11] *Rodin Platform*. http://www.event-b.org

[12] *Rodin Project*. http://rodin.cs.ncl.ac.uk

Acknowledgements: I would like to thank L. Voisin, S. Hallerstede, Thai Son Hoang, F. Mehta, F. Terrier, M. Butler, D. Cansell, and Ch. Metayer who were all very active in the development of the Rodin Platform.