Applying a Formal Method in Industry: a 15-year trajectory

Thierry Lecomte[1]

[1] ClearSy,
Aix en Provence, France
Thierry.lecomte@clearsy.com

Abstract. This article presents industrial experience of applying the B formal method in the industry, on diverse application fields (railways, automotive, smartcard, etc.). If the added value of such an approach has been demonstrated over the year, using a formal method is not the panacea and requires some precautions when introduced in an industrial development cycle.

Keywords: B formal method, deployment, industry.

1 Introduction

Historically, the B Method [1] was introduced in the late 80's to design correctly safe software. Promoted and supported by RATP[1], B and Atelier B, the tool implementing it, have been successfully applied to the industry of transportation. Figure 1 depicts the worldwide implementations of the B technology for safety critical software, mainly as automatic pilots for metros. Today, Alstom Transportation Systems and Siemens Transportation Systems (representing 80% of the worldwide metro market) are the two main actors in the development of B safety-critical software development. Both have a product based strategy and reuse as much as possible existing B models to develop future metros.

A more widely scope use of B appeared in the mid '90s, called *Event-B* [2], to analyze, study and specify not only software, but also whole systems. *Event-B* has been influenced by the work done earlier on Action Systems by the Finnish School (Action System however remained an academic project). *Event-B* is the synthesis between B and Action System. It extends the usage of B to systems that might contain software but also hardware and pieces of equipment. In that respect, one of the outcome of *Event-B* is the proved definition of systems architectures and, more generally, the proved development of, so called, "system studies" [7][10], which are performed before the specification and design of the software. This enlargement allows one to perform failure studies right from the beginning in a large system development. *Event-B* has been applied in many cases to various fields: certification of smartcard security policies (level EAL5+, Common Criteria), verification of

[1] Régie Autonome des Transports Parisiens : operates bus and metro public transport in Paris

Ariane 5 launcher embedded flight software, generation of proven hardware specification [6], etc. *Event-B* has now its own modelling and proof platforms: Atelier B[2] and Rodin[3].

In this article, we try to make clear what the different usages of B are in industry, and to report on experienced added-value, in order to provide more arguments for and against formal methods.

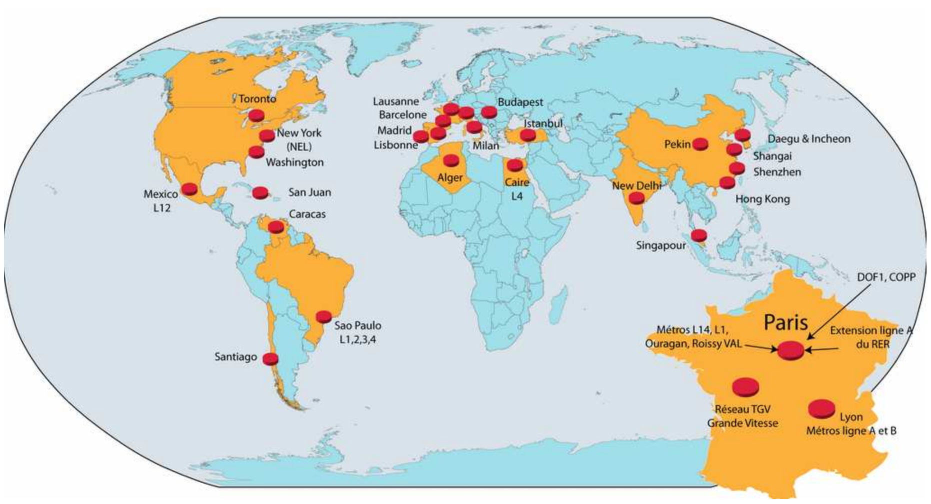


Fig. 1. Worldwide implementations of systems embedding software generated from B models.

2 The Genesis

First real success was Meteor line 14 driverless metro in Paris: Over 110 000 lines of B models were written, generating 86 000 lines of Ada. No bugs were detected after the proofs, neither at the functional validation, at the integration validation, at on-site test, nor since the metro lines operate (October 1998). The safety-critical software is still in version 1.0 in year 2009, without any bug detected so far.

At that time, because of demographic explosion in Paris and its suburb, it was decided to reduce the interval between trains (by using fully automated trains) in order to transport more passengers, as it is not possible or is very costly to modify existing infrastructures or to create bigger, specific trains and cars. No technique/technology/method was seen as mature enough to back the development off the embedded software. RATP spent several millions € for transforming a tool developed internally at Alsthom Transportation Systems into a CASE tool able to generate SIL4 compliant code, leading to the creation of Atelier B.

[2] *http://www.atelierb.eu/*

[3] *http://sourceforge.net/projects/rodin-b_sharp/*

It was initially a one-man decision to fund the development of a prototype tool that would be used for building a software responsible for transporting safely millions passengers a year. This decision had many consequences on the organization of RATP, leading to have almost more people involved in the verification of the development process and documentation than engineers producing software on contracting company side. Even if Atelier B was not developed formally, this tool was subject to extensive verification and validation. In particular:

- the theorem prover was subject to external expertise,
- a dedicated tableau-based prover was build to validate most of the theorem prover mathematical rules,
- a committee was set up to demonstrate by hand unprocessed rules
- a mini automated prover was developed to verify the correctness of the dedicated tableau-based prover

The overall process was time and resource consuming but at the end it was accepted as a sound process. Several qualifications were operated by RATP on B software, that are today subcontracted to any V&V engineer. Thanks to the huge contribution on V&V methodology, B development cycle is perceived as a standard and doesn't require any more specific resources.

Siemens and Alstom claim today to develop most safety critical software with B. Siemens has also developed a useful technology, a tool able to generate semi-automatically refinements and implementations, leading to have safety-critical software developed for a cost similar to any other not-safety related software [3].

However, the "magic" of a 100% proven software also requires verifying that its formal specification complies with requirements written in natural language. As everyone knows, natural language is imprecise, ambiguous and requires frequently interpretation. Demonstrating that requirements and specification match is not an easy game, and is not only a question of traceability/coverage but more related to understanding. Except for trivial examples, most requirements are expressed with jargon/technical language which needs to be made explicit, transforming a 2 or 3 line sentence into a full page mathematical predicate. Compliancy in this case requires specialists and is highly subject to human error, requiring extensive validation/simulation at system level.

3 System Level Modeling

B was initially invented for modeling software. However it has appeared at several occasions that a 100% proven software is not a guaranty against failure, as the proof is related to the compliancy between specification and implementation, not to the correctness of the specification of the software regarding the system where it is plugged into. For example, one metro automatic pilot was not able to stop a train at a platform because of a specification error

The idea of using B for modeling system level specification and to identify formally correct software specification then emerged.

3.1 Embedded Software

The invention of Event-B in the 90's coincided with the massive introduction of electronics onboard cars, leading to a cultural shock and serious difficulties to specify and maintain the electronic architecture of recent cars. Lack of methods and tools for validating distributed specification lead us to enter a 5 year close collaboration with the maintenance department of a car manufacturer. At that time, a sub-contracted diagnosis system was able to identity 40% of the faults, theoretically leading to change all removable electronic components to solve the problems. It was too late to operate on the design, so it was decided to set up a formal model of the 52 embedded functions of a car, covering comfort and safety-related functions, (software based, electronics, mecatronics, etc.). Models were developed from different sources of information such as driver manual, technical documents, diagrams, etc. Calling subcontractors providing components was also part of the modeling phase, in order to get a better understanding of the car behavior, being expected or not, especially in case of failure. This aspect is of paramount importance as the real specification of a car appears to be distributed over a large number of persons. Sometimes this specification is not reachable, for example in the case of a manufacturer not willing to share the internal of the devices he is producing. This lack of knowledge leads sometimes to misbehaviors or "self-emerging specification", resulting from the contribution of several devices put together on a car by independent teams. It was for example possible, in a special case, to lock a driver inside a car, even if he had the key to release and open the door.
In the case of our modeling, some of the 52 functions were abandoned because of the lack of information. The related model was in fact full of question marks and we were not able to answer these questions, even by testing the equipments. Hopefully these functions are all not related to safety.

The modeling was constituted of a flat 30 000 lines B model, principally used to make explicit the behavior of the car. No proof was conducted on it. Cause/consequences matrices were extracted from this model and lead to the construction of a excel file, providing hints on the equipment at fault on a diagnosed car. This approach was repeated on 4 different cars but sharing architecture and some equipment. Part of the modeling was reused from one car to the other, leading to cut modeling time by 3 between the first one and the last one.

The resulting documentation was then provided to the design department while explaining that this kind of information is required to maintain modern cars.

3.2 Platform Screen Doors

In France, RATP has used for years platform screen doors (PSD) that prevent customers to enter or to fall on tracks. Such a system was adopted by the METEOR driverless metro, as it dramatically improves trains availability. In order to offer higher quality services and more safety to its customers, RATP was trying to introduce this kind of protection system in several lines, automated or not. For practical reasons, trains and cars could not be modified with the introduction of PSD. Before starting to deploy a new PSD system in an entire line, RATP initiated a project aimed at developing a prototype PSD system for three stations of line 13 [5][8][9].

Once the train is at standstill, the controller should be able to detect train doors opening and closing, and then issue PSD opening and closing orders. These orders have to be securely issued (failure by elaborating a wrong opening order may lead to customers injury or death), and controller have to be designed, tested and validated in accordance with railway regulations (IEC 50126, 50128, 50129 in particular).

In order to reach the required safety level during project timescale, we decided to set up a development method aimed at reaching targeted reliability, and also ensuring traceability between the different stages of the projects in order to reduce the validation effort. This method was heavily based on the B formal method, and applied during most phases of the project.

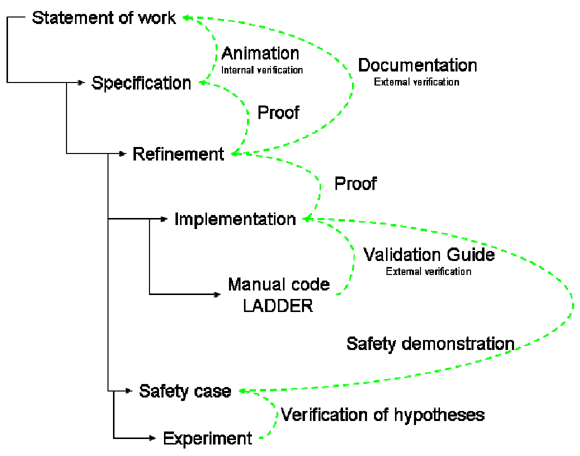


Fig. 2. Development and verification process.

Before any development activity, a formal functional analysis of the system was performed, to evaluate “completeness” and ambiguity freeness of the statement of work. The B method was used to:

- Verify on the overall system (PSD + controller) that functional constraints and safety properties were verified (no possibility to establish forbidden connections between train and platform or between train and tracks).
- Lead to the observation of dangerous system behaviour.

System and software specification were then formalized in B by the development team, taking into account only nominal behaviour for the sensors (in absence of perturbation). Models obtained from previous functional analysis (independent from any PSD controller architecture) were directly reused. The proposed architecture was modelled and inserted in these previous models. New architecture was successfully checked by proof to comply with functional specification of the system, including parts of the French underground regulations. Controller functions were then precisely modelled (train arrival, train detection, train departure, train door opening, train door closing, etc). In the meantime, an independent safety case[4] was developed in parallel by the security team, in order to precisely define how external perturbations may influence the behaviour of the PSD controller. Perturbations were given a priori or a posteriori frequencies, depending on availability of such data at RATP, and a mathematical model, independent from the B model, was set up in order to determine quantitatively the security level of the system. A priori frequencies were verified during the eight month experiment. In case these frequencies were not verified and lower system security below SIL3 level, the PSD controllers would have to be redesigned considering this new information.

Specification documentation was partly elaborated from the system level models developed during this project. The composys[5] tool helps the modeller to add contextual information (comments, description, component name, etc) in B models that are used to generate in natural language the specification documentation describing the complete system. As events are associated to components and as variables are used within events (read/write), Composys computes relationships among components constituting the system being modelled, depending on how variables are read or modified. This document was used to check models with experts of the domain, unable to read and understand formal models.

The development of the software was based on the formal models, as B enables the production of source code, proven to comply with its specification. Siemens automaton can be programmed in the LADDER language but, unfortunately, requires entering program source code via its graphical interface (according to its certificate) to keep its SIL3 accreditation. A dedicated translation schema (from B to LADDER) was elaborated. B to LADDER state diagrams translation is straightforward and some optimisations were introduced in order to verify temporal constraints (cycle time in particular). During validation phase, one can determine which event of the B model corresponds to the path of the LADDER program for a cycle (a LADDER program is defined by logical equations and is analyzed in term of execution path). In case the source code is automatically generated by a qualified translator (as for automatic pilots, by Siemens and Alstom), no unit test is required, this testing phase being covered by the proof of the model.

In this project, as the source code was not generated automatically by such a translator, test was required and test specification was elaborated by usual means. Some months after the beginning of the project, we obtained a fully functional, tested and validated application. The process described above has enabled us to produce a 100% tested, error free (against its specification) software when running validation test bench for the first time. A dedicated test bench was designed to simulate major perturbations (sensors were emulated) and run during days, but no faulty behaviour was observed.

[4] Safety oriented study that provides a convincing and valid argument that a system is adequately safe for a given application in a given environment.

[5] http://www.composys.fr

3.3 SmartCard

On the contrary of safety-critical systems where standards do not require to use formal methods to reach highest safety level, the microelectronics security standards, especially in the smart card domain, oblige circuits to be checked against formal methods for EAL5+ levels and higher. For EAL5+ devices, the security policy needs to be formally verified. Design modeling could make use of semi-formal methods and a table based traceability between specification and design documentation is sufficient. EAL6 and higher levels require (almost) fully use of formal methods at every phase. EAL constraints also propagate to subcontractors involved in the development as well as the technical/confidentiality organization (what is the use of inviolable smartcard if information on the PIN code generation process is easily reachable?).

We were involved in the first certifications of EAL5+ smartcard microcircuits in France, for different companies and in collaboration with different evaluation centers. The main reason for reaching this level was initially due to marketing. Our first evaluations were performed by independent experts that were not aware of B, its restrictions (well known in the railway, due to experience) and French government made some remarks on the resulting evaluation report, leading to the writing of a methodological guidelines for conducting evaluation on Event-B models.

Since then, smartcard microcircuits are regularly evaluated at EAL5+ level, based on an Event-B model of the security policy, with a far better confidence on the results. For a recent product, the evaluation was also performed in Germany by a TÜV, leading to acceptance but also contributing to improve the process (the remarks emitted by the evaluation center have been transmitted to the French government).

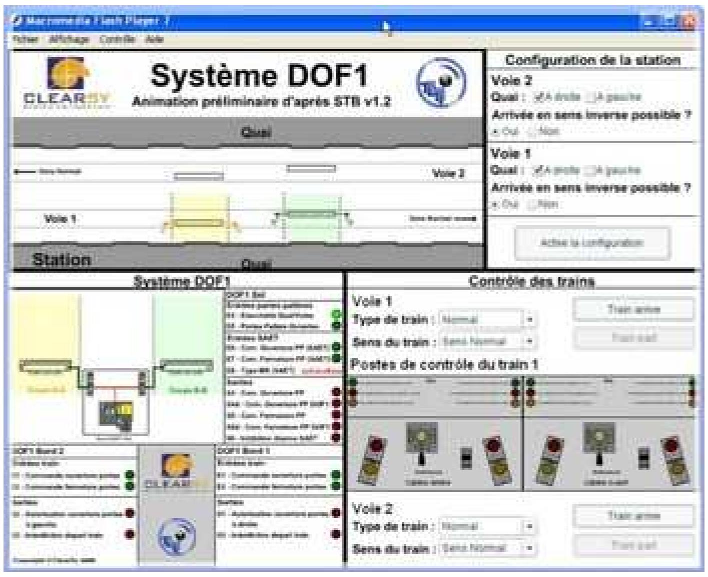
4 Animation and Documentation

Everyone has experienced difficulty to have third party person understanding a state of the art formal model. The mathematical language is at fault, as well as the text-based representation. To counter this, we have experimented two different approaches:

- Generation of documentation, based on the B model and on a dictionary provided by the user, in order to have sentences in natural language describing entities and behavior, as well as a static graphical representation of the relationships among the different entities. This approach is not error prone as you can make mistakes when writing the dictionary, but its main advantage is to make your modeling more understandable that could be evaluated and studied by a third party expert.
- Generation of a graphical animation of the system would help to understand and validate the dynamic part of the modeling, that could be tricky to assess

when dealing with large models and complex/complicated enabling conditions for events. For example, a USB device was once modeled and proved correct until the model was animated and demonstrated not being compliant with the USB protocol (one guard was made too restrictive but it was not detected by proof).

In the case of the development of PSD, specification documentation was partly elaborated from the system level models developed during this project and documented with composys. This tool has no proof capabilities but, as an engineering tool, helps the modeller to add contextual information (comments, description, component name, etc) in B models that are used to generate in natural language the specification documentation describing the complete system. As events are associated to components and as variables are used within events (read/write), Composys computes relationships among components constituting the system being modelled, depending on how variables are read or modified.


Macromedia Flash Player 7
Fichier Affichage Contrôle Aide
CLEARSY
Système DOF1
Animation préliminaire d'après STB v1.2
Configuration de la station
Voie 2
Quai :
Arrivée en sens inverse possible ?
Voie 1
Quai :
Arrivée en sens inverse possible ?
Activer la configuration
Quai
Voie 2
Voie 1
Station
Quai
Système DOF1
Contrôle des trains
Voie 1
Type de train :
Sens du train :
Postes de contrôle du train 1
Voie 2
Type de train :
Sens du train :
Fig. 3. An animated model submitted for a call for tender.

Animation was used at several occasions, including during the writing of proposals where B models were developed and associated with a flash-based animation The resulting B model was animated [11] with the Brama animator[6], in order to verify on given scenarios that the model produced was corresponding to the real system we were modelling. This model animator was not part of the validation process, as this would require it to be qualified as a SIL3 software, but it helped us to check models against reality and to internally verify their suitability.

The animations were also used to make top managers happy and "intelligent", because usually they have to read thick documents (call for tender, proposals) and

[6] *http://www.brama.fr*

most of the time they mainly read the commercial part. Standalone animations allow for easily understanding the functional specification of a system and to play with the system, keeping in mind that the resulting software will be generated from the B model being animated. We finally discovered that this kind of deliverable was widely distributed among target service and could be more or less considered as gentle virus (we saw our animation used as screen-saver on many computers).

5 Conclusion

Formal methods have been considered for years as hobbies for PhD and academic people, by industry. Being in charge of disseminating the B method, we have experimented its introduction into several development processes, including ours. If B is well introduced in the railways domain, it has started to conquer microelectronics, due to the fact that this method has acquired maturity over the years. However it is important to determine how much and where to use a formal method within an existing organization. New tools and new practices are available to ease acceptance in industry.

References

1. Abrial, J.R. (1996) , *The B-book: Assigning programs to meanings,* Cambridge University Press
2. Abrial, J.R (2005)., *Rigorous Open Development Environment for Complex Systems: event B language*
3. Burdy, L.(1996) , *Automatic Refinement. In Proceedings of BUGM* at FM'99
4. Casset, L.,*(1999) A formal specification of the Java byte code verifier using the B method,* Lisbonne 99
5. Sabatier, D. & al (2006), *Use of the Formal B Method for a SIL3 System Landing Door Commands for line 13 of the Paris subway*, Lambda Mu 15
6. Benveniste, M. & al (2009), *A Proved "Correct by Construction" Realistic Digital Circuit*, RIAB, FMWeek 2009
7. Sabatier, D. & al (2008), *FDIR Strategy Validation with the B method*, DASIA 2008
8. Lecomte, T. (2008), *Safe and Reliable Metro Platform Screen Doors Control/Command Systems*, FM 2008
9. Lecomte, T. & al (2007), *Formal Methods in Safety Critical Railway Systems*, SBMF 2007
10. Hoffmann, S. & al (2007), *The B Method for the Construction of Micro-Kernel Based Systems*, ZB 2007
11. Lecomte, T. & al (2007), *BRAMA: a New Graphic Animation Tool for B Models*, ZB 2007