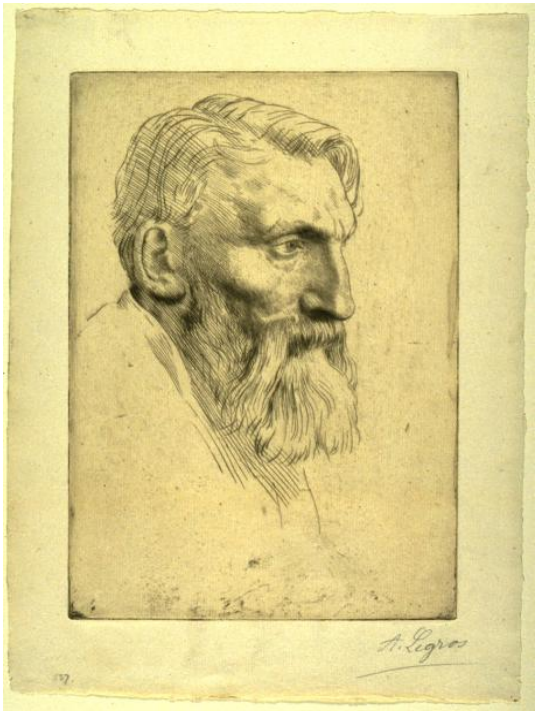
Project IST-511599
RODIN
"Rigorous Open Development Environment for Complex Systems"



RODIN Deliverable D28

Report on Assessment of Tools and Methods

Editor: *Elena Troubitsyna (Aabo Akademi University, Finland)*

Public Document

30th October 2007

http://rodin.cs.ncl.ac.uk/

Contributors:

Budi Arief (University of Newcastle upon Tyne, UK),
Michael Butler(University of Southampton, UK),
Alex Iliasov (University of Newcastle upon Tyne, UK),
Dubravka Ilic (Aabo Akademi University, Finland),
Ian Johnson (ATEC Engine Controls Ltd, UK),
Maciej Koutny(University of Newcastle upon Tyne, UK)
Linas Laibinis (Aabo Akademi University, Finland),
Sari Leppänen (Nokia, Finland),
Qaisar Malik (Aabo Akademi University, Finland),
Mats Neovius (Aabo Akademi University, Finland),
Ian Oliver (Nokia, Finland),
Mike Poppleton (University of Southampton, UK),
Abdolbaghi Rezazadeh (University of Southampton, UK),
Alexander Romanovsky (University of Newcastle upon Tyne, UK),
Kaisa Sere (Aabo Akademi University, Finland),
Colin Snook (University of Southampton, UK),
Jenny Sorge(University of Southampton, UK),
Elena Troubitsyna (Aabo Akademi University, Finland)

CONTENTS

SECTION 1. INTRODUCTION

In the deliverable D28 we present assessment of RODIN methods and tools according to the criteria defined in D2 -- Definition of case studies and evaluation criteria. The generic case study tool criteria have been defined as follows:
1) How much effect have the new tools had on the case study?
2) How hard is the case study work to do before the tools arrive?
3) How hard it is to learn to use the tools in the case study? (shape of learning curve)
4) What is it like when you stop using the tool, once you're used to it?
5) Contrast the experience gained between case studies, identifying which case studies have contributed unique measurements and which (if any) only repeat information gained from other studies?
6) Evaluate expressive convenience of the formal notations used

In each section of this deliverable we present the assessment performed according to these criteria by the case study leading organizations. Each section also presents concluding remarks assessing the achievements of the corresponding case study.

SECTION 2. CASE STUDY 1: FORMAL APPROACHES IN PROTOCOL ENGINEERING

2.1 Introduction

In this section we present the assessment of tools and methods used by case study 1. The section provides a response to the generic case study tool criteria for case studies given in the deliverable D2 [2.1]. The development of the case study is described in the final case study development report D26 [2.2]. The main assessment is provided in the final assessment report D34 [2.3].

2.2 Tools and Methods used by the case study

For most of the project the primary method being used in the case study has been Classical B. In the last year the results developed in Classical B have been translated into Event-B. The methodological advances of the case study are reported in D26 [2.2].

During our work on the case study we have used the Rodin platform, and, to smaller degree, the UML-B and ProB plug-ins. The latter plug-in has been also used in our development on the model-based testing plug-in. The individual assessment of these tools is given in D34 [2.3]. The generic criteria are addressed below.

2.3 Generic case study tool criteria

2.3.1 How much effect have the new tools had on the case study?

The new Rodin platform has helped us to verify formal Lyra development that is the main result of the case study. Of course, the major part of this development has been accomplished before the new Rodin platform arrived. However, we have successfully translated and replicated the previously achieved results in the new platform.

We have also used the ProB plug-in in the context of developing the model based testing (MBT) plug-in. The MBT plug-in uses the ProB engine to generate execution traces. ProB is easy to configure and use as an independent plug-in for the RODIN platform. However, the plug-in to plug-in interaction is not well defined. There was very little support available for such a purpose. The lack of documentation and unavailable application programming interface (API) should be addressed in the future.

2.3.2 How hard is the case study work to do before the tools arrive?

As mentioned before, we have used Classical B (and its tool support AtelierB) before the new tools arrived.

2.3.3 How hard it is to learn to use the tools in the case study?

The Rodin platform If you have some experience with B and Eclipse, it is relatively easy to get started with the Rodin platform. The interface is nicely structured and easy to use. A few hours of testing features and generally getting to know where to find the things you need is all that is needed to get a decent start. It would maybe be an good idea to have a manual of the Event-B language easily available, so you know, e.g., there is no point in trying to make a sequence.

UMLB Once the platform was installed, installing the UMLB plugin was not difficult. UMLB appears as a separate perspective in the Eclipse environment. Modeling in UMLB was, however, not so intuitive at the beginning, despite the solid knowledge of UML. The main difficulty in modeling in UMLB was caused by quite rigid design flow supported by the modeling tool.

With a proper guidebook, UMLB can be adopted quite fast and used in developments. However, we believe that it is necessary to have a prior knowledge of EventB in order to do successful developments in UMLB.

ProB The ProB plug-in for RODIN platform has proved itself to be very useful for animating Event-B specifications. The graphical user interface of ProB plug-in is quite intuitive and user friendly. It is easy to animate and generate execution traces of Event-B specifications. Although, there are still few bugs in this prototype version and it does not fully support the Event-B language at the moment.

More about assessment of the Rodin tools can be found in D34 [2.3].

2.3.4 What is it like when you stop using the tool, once you're used to it?

There was no need to stop using the tool. However, the new Rodin platform has a number advantages over the AtelierB tool that we have used for formal development in Classical B. The main one is that, each time when the project is saved, not only that type checking is performed but proof obligations are generated as well, and, if possible, automatically discharged. This really saves time comparing with how it is

done in AtelierB. Comparing to AtelierB, using Rodin platform we were able to get more proof obligations discharged automatically for the same development, i.e., based on the same specifications. However, when it comes to proving those of remaining proofs, not everything is that much intuitive. Here, we would really benefit from some detailed manual on proving, although the initial guidebook already partially explains it.

2.3.5 Unique measurements of the case study

Case study 1 has some unique measurements related to the design method Lyra, formalisation and automatic support of which is the main goal of the case study:

1. How well do the developed concepts, methods, and tools fit with the existing development framework?
2. How much support does the RODIN approach provide for a more rigorous development process? Specifically, how many new tasks in the development process can be tackled using the methods developed in RODIN?
3. How much support does RODIN provide for automation of the development process? Specifically, how many new tasks in the development process can be tackled using the methods developed in RODIN?

Full assessment based on these metrics is given in D34 [2.3].

2.3.6 Expressive convenience of the formal notations used

Event-B is very easy to learn and use in the Rodin platform. However, it is missing some data structures (like sequences), which would be very useful in expressing certain features of our models. Also, the absence of conditionals and sequential composition forces to split the model operations into smaller ones, which in some cases is somewhat unnatural and / or inconvenient.

2.4 Assessment Conclusion

The case study has mostly successfully applied the new methods and tools. The formal development modelling the Lyra design flow has been verified by using the new Rodin platform. The ProB plug-in has been used to implement test generation based on the developed model-based testing methodology. The full assessment of the case study and related tools can be found in the final assessment report D34 [2.3].

The tools have been relatively easy to install and use. However, as mentioned before, the plug-in to plug-in interaction is not well defined. Also, the lack of documentation and unavailable application programming interface (API) is also a major concern in some cases.

References

2.1 RODIN Deliverable D2: Definitions of Case Studies and Evaluation Criteria, Project IST-5111599. November 2004. Available at http://rodin.cs.ncl.ac.uk/.

2.2 RODIN Deliverable D26: Final Report on Case Study Developments, Project IST-5111599. September 2007. Available at http://rodin.cs.ncl.ac.uk/.

2.3 RODIN Deliverable D34: Final Assessment Report, Project IST-5111599. September 2007. Available at http://rodin.cs.ncl.ac.uk/.

SECTION 3. CASE STUDY 2: TOOLS AND METHODS ASSESMENT OVERVIEW

3.1. Introduction

This section of the D28 report is to contribute to the assessment of tools and methods used by case study 2 . It provides a response to the D2 evaluation [3.1]generic case study tool criteria for case study 2. The case study involved work on two cases the Engine Failure Management system case and a smaller second case the production acceptance test "PAT". The development work on each case is described in the final development report for the case study section 3 of D26 [3.2]. The main evaluation is provided in section 5.2 of the evaluation report D34 [3.3].

3.2. Tools and Methods used by the case study

The principal method being used by the case study was UML-B. The study also used Classic B and Event-B in its initial work. Additional issues in methodological model development have been explored and are reported on in D26 for the case study.

The following tools and plug-ins were used in each case. Their individual assessment in the case is given in section 5.3 of D34. The generic criteria are addressed below.

	FMS	PAT
Eventb platform	X	X
UML-B	X	X
PROB	X	X
B2RODIN	X	

Table 2: Case study tools used

3.3. Generic case study tool critieria

3.3.1. How much effect have the new tools had on the case study?

The new tools helped create a fully validated and verified FMS model. The object-oriented approach of the UML-B assisted the construction of a generic FMS model. Object-orientation is a very natural development approach for FMS, so UML-B could contribute positively to the development. The addition of *u*B annotations (this is an Event_B based action constraint language used in UML-B) enabled us to produce a fully verified model. The ProB plug-in was

used for animation purposes, which could be utilised to ensure the correct implementation of functionality at any time.

The partial specification of the PAT system was developed, verified and validated using the RODIN toolset. The ProB animation being particularly useful for validation. The generic editor of the PAT case used Rodin related technology. Here a structural model which defined the generic editor was successfully created in UML but could not be so easily defined in UML-B which limited the use of the toolset here. (This has been a useful basis for future UML-B research).

An evaluation of the tools used in the case study was provided in D34 (section 5.2 for the case study). It concludes that the toolset has been useful in developing and verifying models but in some cases the tools require further maturity before they can be used commercially by industry.

3.3.2. How hard is the case study work to do before the tools arrive?

FMS development used more traditional methods prior to use of the tools. The FMS domain was found to be difficult to maintain as the mapping of the domain to design was not always clear. The domain aims from the technology and its toolset were to be able to reduce the semantic gap and promote reuse.

The configurability of the PAT system would have been more entrenched in embedded code and would been difficult to have achieved the same amount of configurability. The automatic generation of the editor from the structural model was therefore of great benefit to the domain developer.

3.3.3. How hard is it to learn to use the tools in the case study?

ATEC found the methods difficult to learn as a novice without a background in formal methods and involved a steep learning curve. However over the period of the Project understanding of the methods became easier but still felt that more experience needed to be gained before being comfortable with applying model development on commercial timescales. (Ideally the PAT system would have been modeled completely and translated to an implementation using the methods).

The actual tools themselves were relatively straightforward to apply but would have benefited from some clearer documentation (at the time of developing). However ATEC still found interactive proving of models difficult, although the tool interface was a lot clearer. ATEC found the new reactive prover was an improvement over B4free for it allowed errors to be detected when they were introduced and automatic proving was powerfull.

A viewpoint from a student experienced in formal methods is given below.

“The tools used for this FMS case were the UML-B plug-in ,the ProB animator and the automatic and interactive prover of the Rodin platform.

Having modeling experience in UML, the concept was quite clear, so the basic model could be constructed quite easily. The Event-B annotations could be made while constructing the UML model. Prior to starting this project, I had background knowledge of B and other formal methods, so the concept of those was quite clear. The switch from B to Event-B is straightforward and does not take much of a learning curve. At first, the Rodin platform seems very cluttered and seems hard to navigate. However, after having used it for some time it became a lot easier to find things.

The real learning curve is in the use of the interactive prover, however, for users with substantial background in B4Free, this curve will be fairly low. The interactive prover provides a user friendly interface, which provides a lot of information on the current proof. The problem is however, that the novice won't be able to use all this information because it can seem overwhelming.

Better and more detailed documentation would help the novice user navigate the platform more easily and be aware of all its functionality."

3.3.4. What is it like when you stop using the tool, once you're used to it?

There has been no need to stop using the tools, other than in cases where partners required features which were not currently available.

3.3.5. Contrast the experience gained between case studies, identifying which case studies have contributed unique measurements and which (if any) only repeat information gained from other studies

Case study 2 has some unique subjective measurements over other studies

As
1) Novice industrial views were taken into consideration.
2) Certification standards were assessed in relation to the method.

These have been reported on in D34 and in this document.

3.3.6. Evaluate expressive convenience of the formal notations used

ATEC found UML-B features such as statemachines, contexts and the dot notation useful to express the functionality of the partial specification in its PAT case but found limitations when trying to use the notation as a Domain Language in developing the generic editor. (ref D26 an D34).

The FMS generic model was naturally expressed using UML-B classes and associations. However it was found that Event-B does not support all the data structures of Classic B, in particular sequences, which made modelling more

onerous as these structures had to be self-made. UML-B helped to maintain a specific structure to the model however it consequently restricts modelling freedom.

3.4. Assessment Conclusion

The case study successfully applied the new methods and tools to both cases. The FMS generic model benefited from UML-B object orientation to express the domain concepts. The Aabo model applied the methods but incured some delays in its model development due to availability of features in UML-B. ATEC found UML-B provided a better visualization representation of requirements than its B modeling in year 2.

The toolset has been relatively easy to apply in most cases and benefits have been gained.(The individual assessment of the actual toolset has been assessed in the D34 report). However at the time of evaluation it was felt that the tools were not quite mature enough to be used commercially as some bugs and features needed to be addressed. (Event_B platform version 0.7.4 , UML-B version 0.2.11 and ProB version 0.5.1 were used during evaluation). The bugs and feature requests have been stored in sourceforge and some are outlined in case study 2 sections of D26 and D34

3.5. References

[3.1] RODIN deliverable D2 : Definitions of Case Studies and Evaluation Criteria Project IST-5111599, November 2004.

[3.2] RODIN deliverable D26 : d1.5 Final Report on Case study Developments IST-5111599, September 2007.

[3.3] RODIN deliverable D34 : D7.4 Assessment report IST-5111599, September 2007.

SECTION 4. CASE STUDY 3: TOOLS AND METHODS ASSESMENT OVERVIEW

4.1. Introduction

This case study is concerned with the formalisation of various subsets of the MITA platform [MITA] and the formalisation of the infrastructure and techniques to allow MDA to be used more formally. This section complements and summarizes the results reported in the deliverables D26, D27 and D34 by specifically focusing on the assessment of the tools and methods developed within the project.

4.2. Tools and Methods used by the case study

The following set of tools have been used and assessed in CS3: the Rodin platform, the ProB and U2B plugins. An experimental version of the B2Bsw plugin supporting hardware design has been developed and assessed as well. The B method has been used in developing some functionalities of MITA and some experiments have been conducted on introducing it into one of the current development flows of the MITA systems.

Within this case study a method for introducing formal transformation of platform independent models (PIM) to platform specific models (PSM) in a model driven architecture (MDA) context was developed and applied [B1]. It uses a model transformation of the PIM in order to preserve refinement properties in the construction of the fault tolerant PSM using Event B as a formal framework for the reasoning.

4.3. Generic case study tool criteria

- *How much effect has the new tools had on the case study?*

The use of ProB was extremely successful, in particular, in validating the verified models: a number of complex error conditions have been removed using this tool. ProB and the validation style of development which it supports, provides a way of first constructing and demonstrating systems and then discovering properties later.

It was found that U2B overall provides a good compromise between the mathematical abstractness of B/Event B to the apparent "concreteness" of UML. It has helped the CS3 team to make UML modelling more rigorous.

- *How hard is the case study work to do before the tools arrive?*

It was found that ProB and U2B allow the CS3 team to enrich the development environment with the functionalities which were clearly missing and much needed.

- *How hard it is to learn to use the tools in the case study? (shape of learning curve)*

Learning of ProB was easy. It fits well into the way Nokia engineers work.

Learning U2B was slower as it makes it more difficult to use UML by requiring engineers to write actions/operations/invariants in a form more applicable to B/Event B rather than the object-oriented ideals of UML.

- *What is it like when you stop using the tool, once you're used to it?*

The CS3 team does not have this experience.

- *Contrast the experience gained between case studies, identifying which case studies have contributed unique measurements and which (if any) only repeat information gained from other studies*

CS3 gained a lot by using ProB. Nokia engineers use UML, so all Rodin efforts dedicated to making this development more rigorous and to supporting it by the tools are very important.

B2Bsw: Nokia experience in developing and integrating new plugins into the Rodin platform has been very positive. The team's work on hardware design has now moved outside Rodin, but the experience gained within the project using the B2Bsw plugin was extremely useful.

- *Evaluate expressive convenience of the formal notations used*

In the CS3 context the engineers experienced in using UML and OO design had some difficulties in using the Event B style of development. They clearly needed some training in the new development method.

4.4. Assessment Conclusion

Nokia consider the three years work on the RODIN Case Study 3 to be a partial success. They have obtained

- useful practical results in evaluating feasibility of applying formal methods in the context of MDA
- considerable experience with the use of B in a number of challenging applications
- extended skills in using the Rodin platform and the ProB and U2B plugins as the major support for formal modelling
- good experience in developing Rodin Eclipse plugins for the Rodin platform

However, the complete methodological support for Event B and fault-tolerance for the MITA type of applications is still lacking. More work on improving the U2B tool is necessary.

References:

[B1] P. Boström, M. Neovius, I. Oliver, M. Waldén. Formal Transformation of Platform Independent Models into Platform Specific Models. In Proceedings of the 7th International B Conference (B2007), Besançon, France, LNCS. 4355, pp. 186-200, January 2007. Springer-Verlag.

[MITA] Mobile Internet Technical Architecture. IT Press. 2002.

SECTION 5 - CASE STUDY 4:
CDIS AIR TRAFFIC CONTROL DISPLAY SYSTEM

5.1. Introduction

CDIS is an air traffic display information system. CDIS is real-life system responsible for displaying information to the air-traffic controllers. The controllers interact with the system to select the information they want to see. The information displayed includes:

- Arriving and departing flights
- Weather conditions
- Equipment status at the airports
- Information fed into the system manually In addition, the controller who managers the sequence of incoming flight uses a touch-screen to directly manipulate the sequence through the CDIS.

A subset of the initial CDIS specification has bee selected as a real case study to be redeveloped in RODIN. This case study provides RODIN with the opportunity to compare the capabilities of modern formal methods tools against what was commercially feasible ten years ago. The size of the specification was the first test of the RODIN tool platform, as it highlights any scalability issues that the platform might have. Once the specification has been developed in the RODIN tool, the secondary test was the degree of analysis that is possible for the specification.

The case study is using Event-B extensively. With a substantial subset of the original CDIS specification redeveloped in Event-B, it provides a good opportunity to asses many different aspects of the tool. Furthermore a distributed version of the specification with subsequent refinements has been produced to pave the path for linking the specification to a realistic distributed design and implementation. Different aspects of the CDIS redevelopment have been reported in many RODIN deliverables.

5.2. Tools and Methods used in Developing CDIS

CDIS was an industrial-strength case study with a high degree of complexity. Although as far as RODIN is concerned only a subset of the initial system has been chosen to be redeveloped, but still it intended to utilize the main platform and some plug-ins very heavily. The initial plan was that in addition of the main platform it has to assess ProB and U2B plug-ins. Due to some technical complexity in the current CDIS models the plug-ins do not provide the required level of support. Therefore we have not been able to assess these plug-ins. We intend to keep this possibility open for a near future.

A layered approach has been employed to introduce all the requirements of the chosen subset of CDIS into the Event-B specification. The layered development has considerably improved the comprehensibility because we were able to capture the essential functionality of the system in the abstract specification. The abstract model is just under 4 pages of Event-B and we claim that this abstract model allows the reader to quickly grasp the essence of the system. Six subsequent refinements were used to introduce additional features of the system. The main features of these refinements are that they add additional details to the information structures and introduce further constraints on the events' guard. The layered nature of their introduction means they can be absorbed in a stepwise fashion thus easing comprehensibility.

One of the main criticisms of the initial CDIS specification was that there were no formal link between the core specification and design-level models. Based on experiences obtained during the redevelopment of the idealised version in Event-B, we have developed a distributed version of the specification. This specification takes into account distribution and associated delays. It provides a practical approach to develop a more realistic specification and formally link it to a distributed refinement.

5.3. Generic Case Study Tool Criteria

During Year 3 of the project the B development was ported to the RODIN platform to help with its evaluation. Although some attempts were made during year two to mimic the newly introduced notation of the RODIN Event-B, it was not possible to take the full advantage of the new notation until the new tool platform became available. The main reason is that B4free tool only supports the standard B-method. Thus we had to amend the B models, which were developed during the second year to adjust them with the new Event-B constructs.

The process of porting from standard B to the new RODIN platform proved to be very challenging and it has provided the main platform developer with a wide variety of feedback and a wish list to be considered for future extensions. In the next section we intended to review the mutual effects of the tool and the case study on each other.

5.3.1. How much effect have the new tools had on the case study?

Both the supported notation of the new Event-B and the recommended methodology in the RODIN are different from what now is be recognised as standard B. Consequently the produced B models in Year 3 are different from B4free models which have been produced in Year 2. Some of these differences are:

- Removing input parameters with surrounding parentheses from the front of event's name and replacing them by variables inside ***ANY*** clauses.
- Removing ***PRE*** clauses and replace them with ***ANY*** clauses.
- Removing ***SELECT*** clauses and replace them with ***ANY*** clauses.
- Removing ***LET*** clauses and replace them with ***ANY*** clauses.
- Removing any nested combination of ***ANY*** clauses or nested combination of ***ANY*** with ***PRE/LET/SELECT*** and replace it with a single ***ANY*** clause.
- Adding a separate new ***INITALISATION*** event
- Some other small changes like changing the ***Remove Operator*** from set to "\"
- There is no need to define the operation of the refinement levels which they are ***skip*** in the specification level.

Beside differences in the style of modelling, the new RODIN tool now is capable of generating a broader range of proof obligations. Inline with this the ability of the tool to automatically discharge more proof obligation has increased noticeably. The efficiency of the tool generally and the prover efficiency more specificity also have improved in the recent release of the main RODIN platform.

5.3.2. How hard is the case study work to do before the tools arrive?

The B4free environment and other related tools were mainly based on batch processing analogy. Therefore in all the B tools before the RODIN tool, development process was a serial process and some time very time consuming. The integrated environment of the RODIN platform and its associated plug-ins has brought a lot of ease to the development process. In addition to that,

facilities such as on-fly proof generation and multi-view for exploring different aspects of the project such as proof tree, interactive prover view and the live linking between proofs and the source model are among very useful features. All of these new features should facilitate higher level of productivity.

5.3.3 How hard it is to learn to use the tools in the case study?

The IDE interface of the new RODIN tool has made it very easy and attractable to interact with the tool. In comparison with the primitive and not very straightforward interface of the B4free tool, the RODIN interface is very accessible. Almost all of the new tools facilities are complying with the standards of the modern tools. Therefore the learning process of general aspects of the RODIN tool should not be different from any other tool.

Other aspects of the tool which related to the use of formal notation and proof for system modelling are slightly different. Having good understanding of formal modelling, proof system and knowledge of supported methodology, a comprehensive user manual with some example case studies and finally have experience in dealing with interactive proofs are very useful. Some of the above aspects like experience with interactive prover are not very easily transferable and they need more time and patient. With improved support for discharging proofs automatically this should not be a major issue in majority of cases.

5.3.4 What is it like when you stop using the tool, once you're used to it?

We have started using the RODIN tool as soon as early versions of the main platform became available. The CDIS case study was intend to heavily employ the main platform and some other plug-ins. Also we have not been able to utilise the plug-ins to the initially expected level, but we have explored many aspect of the main RODIN platform. As the facilities and performance of the tool has gradually increased by the introduction of the new versions we found it very attractive and adoptable environment. We believe that the new platform convey a much more productive environment for formal development in comparison to the previous generation of tools.

5.3.5 Contrast the experience gained between case studies

The layer and stepwise development and refinement which has been recommended by the RODIN methodology is the basis which shared by all case studies. Despite this common ground the CDIS case study has had a specific contribution to the RODIN platform by introducing record type and the notion of gradual refinement of the record type by the means of constant mappings.
In addition to this the heavy use of the main platform has provided us with the opportunity to identify many key issues and provide sizable feedback to the developers of the RODIN tool. These feedbacks have resulted in the production of a better tool.

5.3.6 Evaluate expressive convenience of the formal notations used

The mathematical language of Event-B and VVSL are equally expressive. The key difference was not the notation; rather it was the style of specification used in the Event-B development, in particular the use of refinement to layer in details of the functionality, which led to a more comprehensible specification. The layered approach, along with the powerful B4free tool, made it possible to mechanically check and prove the models.

5.4. Assessment and Conclusion

After releasing a number of sub-versions by the main Platform developer, the tool has gradually reached a reasonable stage of stability, which can now produce and discharge a much wider set of proof obligations in comparison with the B4free tool. The new facilities resulted in several amendments and enrichments to the produced B models during third year. As another result of tool enhancement we were able to develop our B models further and add two further refinement levels, which now bring the total refinement levels to six in addition to the initial specification model. Most of these refinements are horizontal refinements, where we have inter-cooperated new features to the previous levels. A distributed version of the specification and its related refinements has paved the way to overcome one the major weakness of the initial CDIS modelling. This was the lack any formal linking between the core specification and subsequent design documents. This distributed version now has facilitated the formal link between a more realistic specification and its distributed refinements.

The CDIS case study has provided some methodological contribution to the construction of large formal specifications. Our experience shows that incremental construction through iterative refinement makes it feasible to apply tool-based formal analysis to large specifications. This increases our confidence in the specification greatly and provides the basis for tool-based formal development of a design and implementation. We also believe that this approach makes a large formal specification more accessible and comprehensible both to those constructing the specification and to others. We believe that the approach we have taken and the lessons learned can be applied to the construction of large formal specifications more generally.

Section 6. Case study 5 – Ambient Campus

6.1 Introduction

This case study aims at identifying the extent to which various parts of the RODIN approach can provide effective support for the most challenging stages of the formal design process of complex fault-tolerant mobile systems. In particular, the wireless communication medium, on which the implementation part of this case study is based, typically causes transmission errors leading to a whole range of critical faults that must be tolerated. Moreover, such mobile applications require dealing with a variety of abnormal and unpredictable events due to system openness, mobility of its participants and their dynamic nature.

The work on the Ambient Campus case study (from the Description of Work [**Error! Reference source not found.**]) has focussed on:

a) elucidation of the specific fault tolerance and modelling techniques appropriate for the application domain,
b) validation of the methodology developed in WP2 and the model checking plug-in for verification based on partial-order reductions, and
c) documentation of the experience in the forms of guidelines and fault tolerance patterns.

More specifically, in this case study we have been investigating how to use formal methods combined with advanced fault tolerance techniques in developing highly dependable Ambient Intelligence (AmI) applications.

6.2. Tools and Methods Used by the Case Study

RODIN Platform

The RODIN platform was used extensively by CS 5 in the work on Year 3 scenario. The modelling of the case study one of the first applications of the platform in the context of realistic, large-scale specifications. Few problems have been found, mainly with the tool interface and these were promptly addressed by the platform developers (bug reports and suggestion were submitted on a regular basis through the sourcefourge tracking facility).

B2RODIN Plug-In

This plugin has been developed to transfer AtelierB projects into the new RODIN platform. The plugin is extremely simple in use and no issues have been found. We have applied the B2RODIN plug-in to transfer previous AtelierB and Click'n'Proof developments into the new Rodin Platform. The plugin performance was satisfactory and it is very easy to use.

MobilityChecker

Motivated and inspired in a direct way by CS5, we developed a plug-in for the RODIN platform based on an automatic verification engine of proven efficiency

(developed for high-level Petri nets) that supports the model checking of a given specification of mobile systems. It has been used extensively in the work of CS5. The verifier checks for deadlock freeness and invariant violations and it is capable to provide feedback in case of discovering an error in the specification. These error traces can be visualised with the help of the included animator, providing further assistance to the designer.

ProB Plug-In

The integrated version of the ProB tool was to animate various stages of CS5 design directly from the platform. It is a very robust tool that works very well even with large and complex models. The interface is also very good. The only minor downside is that animation of complex models can be somewhat slow. ProB plug-in to the platform is essential tool for understanding complex models. Large, involved specifications are hard to read, even more so in Event-B which specifications tend to have large number of events due to absence of sequential composition. Model animation is an efficient and user friendly for model interpretation.

Model-Based Testing

Ambient Campus case study (CS5) provided a good test-bench for developing the theoretical foundations for model based testing approach (MBT). Part of CS5 models, consisting of formal Event-B [16] specifications of middleware, was used for testing the MBT methodology. The team working on CS5 was consulted on several occasions to derive testable information from formal models. One of the important results of this cooperation was identification of particular testing scenarios and refinement patterns. These testing scenarios were then refined on different refinement levels. In order to make automatic refinement of testing scenarios feasible, certain refinement steps and patterns were identified. As an improvement, the formal models for some of middleware specifications were rewritten to certain extent. These models were used as the inputs for the developed model-based testing process and, as a result, the corresponding test cases were derived. The results described above are published in [17].

6.3. Generic case study tool criteria

6.3.3.1How much effect have the new tools had on the case study?

The new tools were absolutely necessary in order to successfully complete the task of CS5. The RODIN platform was used as a support throughout the development in Year 3, and the available plug-ins (or their prototypes) helped to manage the complexity of various tasks.

6.3.3.2How hard is the case study work to do before the tools arrive?

Taking as a specific issue the problem of the verification of behavioural properties of mobile systems, the existing state-of-art model checkers were unable to cope with even small specifications.

6.3.3.3 How hard it is to learn to use the tools in the case study?

There were no identifiable problems with the use of various tools, other than those resulting from their concurrent development (and so small differences between different versions had to be carefully noted).

6.3.3.4What is it like when you stop using the tool, once you're used to it?

Again, taking as a specific issue the problem of the verification of behavioural properties of mobile systems, the mobility plug-in cannot be replaced by a manual code inspection due to the state explosion problem.

6.3.3.5Contrast the experience gained between case studies, identifying which case studies have contributed unique measurements and which (if any) only repeat information gained from other studies

CS5 contributed in a unique way to the development and subsequent evaluation of the mobility plug-in.

6.3.3.6 Evaluate expressive convenience of the formal notations used

The new programming notation developed in the context of CS5 is easily accessible to anyone familiar with B-like languages and some basic process algebra concepts.

6.4 Assessment Conclusions

In CS5 we developed and investigated a novel approach for modelling and verifying the correctness of complex mobile agent systems. There is no doubt that the success of this case study would not be possible without impact made by the RODIN platform, as well as various plug-ins developed within the project.

References

[1] B. Arief, J. Coleman, A. Hall, A. Hilton, A. Iliasov, I. Johnson, C. Jones, L. Laibinis, S. Leppanen, I. Oliver, A. Romanovsky, C. Snook, E. Troubitsyna, and J. Ziegler, "Rodin Deliverable D4: Traceable Requirements Document for Case Studies," Project IST-511599, School of Computing Science, Newcastle University 2005.

[2] "Rodin Deliverable D8: Initial Report on Case Study Development," E. Troubitsyna, Ed.: Project IST-511599, School of Computing Science, Newcastle University, 2005, pp. 63-75.

[3] "Rodin Deliverable D18: Intermediate Report on Case Study Development," E. Troubitsyna, Ed.: Project IST-511599, School of Computing Science, Newcastle University, 2006.

[4] A. Iliasov and A. Romanovsky, "Exception Handling in Coordination-based Mobile Environments," Proceedings of 29th Annual International Computer

Software and Applications Conference (COMPSAC 2005), IEEE Computer Society Press, 2005, pp. 341-350.

[5] A. Iliasov, L. Laibinis, A. Romanovsky, and E. Troubitsyna, "Towards Formal Development of Mobile Location-based Systems," Proceedings of Workshop on Rigorous Engineering of Fault-Tolerant Systems (REFT 2005), (http://rodin.cs.ncl.ac.uk/events.htm), Newcastle Upon Tyne, UK 2005, pp. 53-64.

[6] A. Iliasov and A. Romanovsky, "Structured Coordination Spaces for Fault Tolerant Mobile Agents," in *LNCS 4119*, C. Dony, J. L. Knudsen, A. Romanovsky, and A. Tripathi, Eds., 2006, pp. 181-199.

[7] A. Iliasov, V. Khomenko, M. Koutny, and A. Romanovsky, "On Specification and Verification of Location-based Fault Tolerant Mobile Systems," Proceedings of Workshop on Rigorous Engineering of Fault-Tolerant Systems (REFT 2005), (http://rodin.cs.ncl.ac.uk/events.htm), Newcastle Upon Tyne, UK 2005, pp. 129-140.

[8] B. Arief, A. Iliasov, and A. Romanovsky, "On Using the CAMA Framework for Developing Open Mobile Fault Tolerant Agent Systems," Proceedings of SELMAS 2006 workshop at ICSE 2006, Shanghai, China 2006, pp. 29-36.

[9] A. Iliasov, A. Romanovsky, B. Arief, L. Laibinis, and E. Troubitsyna, "A Framework for Open Distributed System Design," Proceedings of Computer Software & Applications Conference (COMPSAC 07), Volume II - Workshop Papers, 1st IEEE International Workshop on Software Patterns (SPAC 2007), Beijing, China, IEEE Computer Society, Conference Publishing Services, 27 July 2007, pp. 658-668.

[10] B. Arief, A. Iliasov, and A. Romanovsky, "On Developing Open Mobile Fault Tolerant Agent Systems," in *Software Engineering for Multi-Agent Systems V, LNCS 4408*, R. Choren, A. Garcia, H. Giese, H.-f. Leung, C. Lucena, and A. Romanovsky, Eds.: Springer, 2007, pp. 21-40.

[11] B. Arief, A. Iliasov, and A. Romanovsky, "Rigorous Development of Ambient Campus Applications that can Recover from Errors," Proceedings of Workshop on Methods, Models and Tools for Fault-Tolerance (MeMoT 2007), at the International Conference on Integrated Formal Methods 2007 (IFM 2007), Oxford, UK, 3 July 2007, pp. 103-110.

[12] A. Iliasov, A. Romanovsky, B. Arief, L. Laibinis, and E. Troubitsyna, "On Rigorous Design and Implementation of Fault Tolerant Ambient Systems," Proceedings of 10th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC07), Santorini Island, Greece, 7-9 May 2007, pp. 141-145.

[13] A. Iliasov, "Refinement patterns for rapid development of dependable systems," Proceedings of Engineering Fault Tolerant Systems Workshop (at ESEC/FSE), Croatia, ACM, 4 September 2007

[14] "Finer Plugin Introduction," online at http://www.iliasov.org/FinerPlugin.html (last accessed 15 August 2007).

[15] "Smartdust," online at http://en.wikipedia.org/wiki/Smartdust (last accessed 14 August 2007).

[16] C. Metayer, J. R. Abrial, and L. Voisin, "Rodin deliverable 3.2. Event-B language," Project IST-511599, School of Computing Science, Newcastle University, available from http://rodin.cs.ncl.ac.uk/deliverables/D7.pdf 2005.

[17] Q. A. Malik, J. Lilius, and L. Laibinis, "Model-Based Testing using Scenarios and Event-B Refinements," Proceedings of Workshop on Methods, Models and Tools for Fault-Tolerance (MeMoT 2007), at the International Conference on Integrated Formal Methods 2007 (IFM 2007), Oxford, UK, 3 July 2007, pp. 59-69.

[18] "ISTAG Scenarios for Ambient Intelligence in 2010," online at ftp://ftp.cordis.europa.eu/pub/ist/docs/istagscenarios2010.pdf (last accessed 15 August 2007).