Project DEPLOY
Grant Agreement 214158

"Industrial deployment of advanced system engineering methods for high productivity and dependability"

DEPLOY Deliverable D33

D10.4. Report on Training Experience

Public Document

31st January, 2011

http://www.deploy-project.eu

Contributors:

Michael Butler	Southampton
Thai Son Hoang	ETHZ
Alexei Iliasov	Newcastle
Cliff Jones	Newcastle
Linas Laibinis	Aabo
Michael Leuschel	UDUS
Laura Nummila	SSF
Daniel Plagge	UDUS
Tuomas Räsänen	SSF
Matthias Schmalz	ETHZ
Colin Snook	Southampton
Wei Wei	SAP

Editors:

Thai Son Hoang	ETHZ

Reviewers:

Stefan Hallerstede	UDUS
Carine Pascal	Systerel

Contents

Chapter 1

Introduction

The purpose of this deliverable was stated in the "Description of Work" (DoW) as follows.

> **D33** This deliverable is a report on the training courses. It will contain a detailed description of the positive or negative reactions from the various industrial partners. Some possibilities for the local extensions of these material and courses will be given by the industrial partners.

Recognizing the importance of technology transfer, during the "refocus" meeting in Southampton 2009, the project has agreed to extend the original Task 10.3, i.e., "Hold block courses for different audiences" until the end of the project (M48). However, it became apparent that "block courses" are no longer appropriate at this stage of the project. In particular, a proposal for an advance block course for early 2010 was not supported by any of the deployment partners. In fact, the deployment partners prefer more direct technology transfer tackling domain-specific problems. This is supported by a model within DEPLOY where each deployment partner has one main contact point: Newcastle for Bosch, Southampton for Siemens, Aabo for SSF, and ETHZ for SAP. This does not necessarily prevent other direct contact between other academic partners and deployment partners, for example the work on data validation between UDUS and Siemens, or the work on verifying BPMN models between Newcastle and SAP.

Also at the "refocus" meeting, the project agreed to have additional "DEPLOY Associates" (DAs). The DAs also need to be trained to be familiar with the DEPLOY methods and tools. The training was carried out by Southampton for the two companies in this group of DEPLOY Associates: Critical Systems Ltd., Southampton, U.K and AeS, Sao Paulo, Brazil.

Since May 2010, there are two new academic partners joining the DEPLOY project: University of Bucharest, Romania and University of Pitesti, Romania. After a successful kick-off meeting at SAP Darmstadt, Germany, a 2-day block course has been organized at the University of Bucarest to ensure that the new partners can integrate quickly into the DEPLOY working environment. The course was instructed by Stefan Hallerstede (UDUS) and Thai Son Hoang (ETHZ) and focused on the use of Event-B and the Rodin Platform. The training material is available on-line for further similar training purpose [HH10c, HH10a, HH10b].

In summary, some important aspects of our technology transfer experiences are as follows.

- **Technology transfer is bidirectional**. As recognized earlier in the project, technology transfer is both from academic partners to deployment partners and vice versa. Whereas in the first direction, technology transfer is about the methods and tools, in the reverse direction, technology transfer is about domain-specific topics.

- **Initial training is crucial**. At the beginning of the project, we organized a 3-day block course in Zurich for all the industrial partners. The course was judged useful; however, we could benefit from having a longer course with more intensive training.

- **Technology transfer should be driven by domain-specific problems**. In order to have an effective deployment, technology transfer should address domain-specific problems. Ultimately, our aim is to handle industrial applications of formal methods.

- **Technology transfer should offer multiple approaches**. We believe that there should be different approaches for different domain-specific problems. Within the DEPLOY project, several approaches have been proposed as answers to domain-specific problems.

This deliverable is structured as follows. Chapter 2 gives an overview of how technology transfer is managed within the DEPLOY project. Subsequent chapters from 3 to 6 describe our experience in knowledge transfer in different deployment sectors: the *Automotive Sector*, *Transportation Sector*, *Space Sector* and *Business Information Sector*. Chapter 7 gives a summary on the training experience for the two DAs.

Chapter 2

Technology Transfer in General

After Jean-Raymond Abrial left the DEPLOY project and in particular WP10, it became necessary to reorganize the procedures for training partners. This turned out to be a major challenge, because it was impossible to hire a person capable of seamlessly taking over Abrial's duties and because the persons who are qualified to write documentation and carry out on-site trainings were already committed to other tasks, in particular tool development. So investing more time in technology transfer means having less time for tool development.

To make sure that our industrial partners receive the training they need, we agreed on the following procedure of technology transfer in January 2010:

1. The requesting partner checks the project internal list of available material and pending requests (called "wish list").

2. If the request has not already been addressed, the partner sends a detailed description (including level of urgency) to ETHZ.

3. ETHZ forwards the request to a responsible partner. If no partner is willing to take the responsibility, the request is discussed at an appropriate telephone conference.

4. If necessary, the responsible partner further discusses with the requester and/or ETHZ about the particular request (including the expected date of completion).

5. The request is added to the wish list and announced on the WP10 mailing list.

6. Once the request is fulfilled, its status is updated in the wish list and announced on the WP10 mailing list.

7. There are different ways of fulfilling requests including: new documentation, pointers to existing documents, courses, further research, or private communication.

In addition, since August 2010, Rodin Platform users may submit documentation requests to a tracker on Sourceforge[1].

On the bright side, it was almost always possible to find a partner willing to take responsibility for the requests of industrial partners. There were also a few documentation requests on Sourceforge, helping us to correct minor problems in the existing documentation. From the fact that less and less requests were submitted and from the feedback of industrial partners, we conclude that the industrial partners are receiving the training and documentation they need.

The limitation of the chosen procedure is that documentation is created only when there is an explicit request for it. Our industrial partners have pointed out that the existing documentation suffices for carrying out their pilot projects and is better than the documentation of many other research tools, but does not meet common standards for documentation of industrial strength software and may thus be insufficient for prospective Rodin Platform users. Given that spending more resources in writing documentation means spending less resources in tool development, we have decided to improve documentation only if the improvement is relevant for our industrial partners. That allows us to maximize the ressources available for tool maintenance and implementing of new features.

[1] http://sourceforge.net/tracker/?group_id=108850&atid=651672

Chapter 3

Automotive Sector

The technology transfer experience for the automotive partner of DEPLOY is interesting because Bosch are attempting to undertake fairly strict top-down development of non-trivial applications. Moreover, they are facing the issue of cleaning up requirements as a bridge to their initial formal models.

3.1 The Knowledge Transfer Context

Initial training The number of people involved at Bosch has been constant, with one change of individual. The original team consisted of Rainer Gmehlich, Felix Loesch and Christine Rossa, all of whom attended the joint training session (known as the "ETH Block Course") held by Abrial and colleagues at ETH Zurich in April 2008. None of these three staff had significant formal methods expertise before they attended the Event-B/Rodin Platform education. All three judged the course useful. At least with the benefit of hindsight, this course was too short.[1] On a visit in December 2010, Rainer commented that a follow-up education cycle, a couple of weeks after the initial training, might have paid off. (A much later offer to hold an advanced course at ETH Zurich was however not taken up by any of the deployment partners.)

[1]For clarification, it is worth understanding how the choice to use a 3-day format came about. Before the first "executive" meeting, the industrial partners had assumed that the courses would be held at their sites. Not unreasonably, Jean-Raymond wanted to avoid repeating the course but, when he announced during the first executive meeting that there would be only one course and that it would be held in Zurich, the industrial partners could justify neither the travel costs nor the time away for a long course. Not only did this reduce the possible length, it also meant that only the core teams from each deployment partner attended.

Subsequent changes in staffing After about 20 months, Christine was replaced by Katrin Grau who had previous formal methods experience on the VeriSoft project. What is particularly interesting about this change is that Katrin has learnt both the method and the tools mainly from her colleagues (although she did of course also have access to the material from the initial ETH course). This is a clear example of the advantages of having on-site expertise.

3.2 Training Experience

Experience of the mini-pilot An early suggestion was that each deployment partner should offer its own "mini-pilot" as part of their first attempts to become familiar with the tools and methods. Again with hindsight, this was far less helpful with respect to method than tools. In fact, even on tools, a carefully constructed homework exercise might have been more useful.[2] As far as the method aspects were concerned, the mini-pilots gave no clue to the problems that were in store when facing models larger than in the published literature. To be clear, the comments here on "method" relate to the base Event-B method as explained in the ETH course; Bosch in particular have developed their own extensions in subsequent work.

Pilot 1 After considering several alternative projects, Bosch decided to undertake the development of a "Cruise Control" system for their main pilot. This work has been reported in more detail in D19 so only the points concerning technology transfer are addressed here. As described in §3 of D19, the decision was made to work with Jackson's "Problem Frame Approach" to help straighten out inevitable gaps and inconsistencies in the initial natural language requirements. (Abrial's notion of structuring requirements was considered but not thought to be appropriate.) The PFA exercise was seen to be extremely valuable — it was fortunate that Newcastle had a long-standing relationship with Michael Jackson and that he took a personal interest in the on-going work. This analysis was then taken as a starting point for a parallel decomposition in Event-B. This has given rise to one of the largest models put into the Rodin Platform and, not surprisingly, many problems in their use have been uncovered.[3] It is probably true that the tools teams were

[2]The range of approaches discussed for the "bouncing switch" also did little to convey a mature method.

[3]The cruise control model is certainly also among the largest considered with the Event-B method. Nothing in the course material prepared the deployment partners for the challenges of building models of this scale. In particular, neither A nor B-style decomposition

initially too slow to respond to some of these issues but there is now a much better turnaround on queries/bugs etc. Technology transfer can be severely impeded if the users feel that their concerns are not resulting in help. At the time of writing this deliverable, Bosch are some way from being able to complete the development of the cruise control system using the tools.

One unexpected deployment option has been the benefit from using ProB to help locate liveness issues in the cruise control model.

Another very useful interaction was Stefan Hallerstede's work-around for the lack of a way of defining records in the initial tools.

Bosch are also leading edge deployers of the plug-ins from the academic partners (Camille text editor, B-style decomposition, feature composition, etc.). One of the tools is the Flow plug-in — a Platform extension for checking event enabledness properties. The Bosch feedback provided requirements for a more advanced design of the tool. It was also the first large model for which a systematic analysis of enabledness properties was attempted. This has uncovered an array of issues with the Rodin Platform and the approach to verify enabledness properties. The other tool coming from Newcastle is the group refinement extension. The Bosch team had an idea of an alternative refinement style that would better fit the structure outlined in the problem frames specification.

Support from the academic partners Another aspect of technology transfer has been the level of non-course support. Members of the project from several academic partners have made many visits to the Bosch site (made all the easier by them being such a nice group of people to discuss problems with!). The academic partners put in place the strategy that each deployment partner should have one main point of contact: for Bosch, Newcastle was chosen and, specifically, Cliff Jones has been on nearly all of the visits. As mentioned above, Michael Jackson has also been a great help. All of the academic partners (Southampton, ETHZ, Aabo and UDUS) have at some point joined in visits. This has led to a productive exchange and the ability to discuss things that could have resulted in disagreements.

Reverse technology transfer Relating to the preceding point, it is important not to view technology transfer as a one-way street. It is undoubtedly

appear to be relevant to the issues confronted in the cruise control application. It could be argued that other ways of modeling cruise control might have reduced the difficulties but it is essential to realize that real industrial applications are not text book exercises on which one can afford to make repeated throw away experiments. The tools and methods need to adapt to use — not the other way around.

true that the academic partners have learnt a great deal from the deployment partners and this is especially true in the automotive sector.

Pilot 2 At the time of writing, Bosch are embarking on a second pilot. For the moment, the details of the application are unimportant. What is sought is an attempt to benefit from what was learnt in the first pilot and to gain a better interface between PFA and the Event-B method.

3.3 Summary

In summary, Bosch have been attempting a genuine top-down development of a significant — but by their standards, no means large — system. Their conclusion at this point in time is that the methods and tools on offer are such that they are unlikely to be able to roll them out to a wider automotive engineering function. (For one thing the documentation would be inadequate for hand on to larger group of engineers.[4]) In case this sounds excessively negative, the Bosch group feels that the DEPLOY project has provided valuable exposure to the state of the art.

Some of the lessons to be taken away include:

- it is important to realize in any such exercise that academics and industrialists have different motivations
- that technology transfer material should reflect at least some awareness of the scale of problem that is about to be faced
- a PFA tool would have been valuable to Bosch, but more generally it is crucial to think about how new tools can be interfaced with those in use by any deployment partner

[4] A specific suggestion in this area was a request for a centrally maintained project "glossary".

Chapter 4

Transportation Sector

This chapter describes the experience in technology transfer in the transportation sector about using ProB for data validation. More details of the approach is available in [LFFP10]. The following sections describe in detail the experience gained in different deployment areas, and the interactions between Siemens and the University of Düsseldorf. On top of the mentioned exchanges, a meeting was organized by the University of Düsseldorf at Siemens in Paris the 12th of January 2009, where (amongst others) ProB was introduced to several engineers at Siemens. Another more intensive training day was held the 25th of January 2010, where 5 engineers of Siemens were trained in using ProB for data validation. Also, the University of Düsseldorf has developed a tutorial[1] for ProB especially targeted at the pattern of usage of Siemens. These training activities seem to have been successful, as for the later deployments Siemens was able to use ProB independently of the University of Düsseldorf (see Section 4.5 below).

4.1 The Deployment Task

STS are successfully using the B-method and have over the years acquired considerable expertise in its application. STS use Atelier B, together with in-house developed automatic refinement tools, with great success. Indeed, starting from a high-level model of the control software, refinement is used to make the model more concrete. Each refinement step is formally proven correct. When the model is concrete enough, an Ada code generator is used. This results in a system ensuring a very high degree of safety, with SIL4 certification. Indeed, quoting [Sie09]: "*Since the commissioning of line 14*

[1] `http://www.stups.uni-duesseldorf.de/ProB/index.php5/Tutorial`

in Paris on 1998, not a single malfunction has been noted in the software developed using this principle."

The Property Verification Problem

One aspect of the current development process which is unfortunately still problematic is the validation of properties of parameters only known at deployment time, such as the rail network topology parameters. These parameters are typically represented as constants in the formal B model.

In order to avoid multiple developments, each software is made from a generic B-model and data parameters that are specific to a sub-section and a particular deployment. These data parameters take the form of B functions describing, e.g., the tracks, switches, traffic lights, electrical connections and possible routes. Adapting the data parameters is also used to "tune" the system.

The proofs of the generic B-model rely on assumptions about the data parameters, e.g., assumptions about the topology properties of the track. We therefore have to make sure that the parameters used for each sub-section and deployment actually satisfy the formal assumptions.

Siemens Existing Process

To solve this problem, Siemens Transportation Systems (STS) had initially developed the following approach:

1. The parameters and topology is extracted from the concrete Ada program and encoded in B syntax, written into Atelier B definition files. (Definition files contain only B DEFINITIONS, i.e., B macros.) This is done with the aid of a tool written in `lex`.

   Note, that Siemens not only wants to check that the assumptions about the data parameters hold, but also that they have been correctly encoded in the Ada code. Hence, the data is extracted from the Ada program, rather than directly from the higher-level description (which was used to generate the Ada code).

2. The relevant part of the B model is extracted and merged with the definition files containing the topology and the other parameters. The properties from the original B model on the concrete topology and parameters are translated into B assertions.

   In B assertions are predicates which should follow from the properties, and have to be proven. Properties themselves do not have to be proven,

but can be used by the prover. By translating the topology properties into B assertions, we thus create proof obligations which stipulate that the topology and parameter properties must follow from the concrete values of the constants.

3. STS tries to prove the assertions with Atelier B, using custom proof rules and tactics, dedicated to dealing with explicit data values.

4. Those assertions for which proof is unsuccessful are investigated manually.

Problems with the Existing Process

This approach initially worked quite well for Siemens, but ran into considerable problems:

- First, if the proof of a property fails, the feedback of the prover is not very useful in locating the problem (and it may be unclear whether there actually is a problem with the topology or "simply" with the power of the prover).

- Second, and more importantly, the constants are nowadays becoming so large (relations with thousands of tuples) that Atelier B quite often runs out of memory, even with the dedicated proof rules and with maximum memory allocated. In some of the bigger, more recent models, even simply substituting values for variables fails with out-of-memory conditions.

This is especially difficult, as some of the properties are very large and complicated, and the prover typically fails on these properties.

The second point means that these properties have to be checked by hand (e.g., by creating huge spreadsheets on paper for the compatibility constraints of all possible itineraries), which is very costly and arguably less reliable than automated checking. For the San Juan development, this meant about one man month of effort, which is likely to grow further for larger developments.

The starting point of this deployment task was to try to automate this task, by using alternative technology. Indeed, the ProB tool has to be capable of dealing with B properties in order to animate and model check B models. The question was, whether the technology would scale to deal with the industrial models and the large constants in this case study.

4.2 Initial San Juan Case Study

In order to evaluate the feasibility of using ProB for checking the topology properties, Siemens sent to the STUPS team at the University of Düsseldorf the models for the San Juan case study on the 8th of July 2008. There were 23,000 lines of B spread over 79 files, two of which were to be analysed: a simpler model (acs_as_env_cfg_aiguille.mch) and a complete model (acs_as_env_cfg_ipart.mch). The complete model contains 226 properties and 147 assertions. It then took us a while to understand the models and get them through our new parser, whose development was being finalised at that time.

On 14th of November 2008 we were able to animate and analyse the first model. This uncovered one error in the assertions. However, at that point it became apparent that a new data structure would be needed to validate bigger models. Thus, the developments described in [LFFP10] were undertaken. On the 8th of December 2008 we were finally able to animate and validate the complete model acs_as_env_cfg_ipart.mch. This revealed four errors.

Note that we (the STUPS team) were not told about the presence of errors in the models (they were not even hinted at by Siemens), and initially we believed that there was still a bug in ProB. Luckily, the errors were in the concrete data values. Furthermore, ProB found *exactly* the same errors that Siemens had uncovered themselves by manual inspection.

The manual inspection of the properties took Siemens several weeks (about a man month of effort). Checking the properties took 4.15 seconds, and checking the assertions took 1017.7 seconds (i.e., roughly 17 minutes) using ProB 1.3.0-final.4 on a MacBook Pro with 2.33 GHz Core2 Duo. With version 1.3.1, the runtime was further improved, to below 5 minutes, and now with version 1.3.2 on a more recent laptop the assertion checking is done in less than one minute.

Once our tool has uncovered unexpected properties of a model, the user obviously wants to have more information about the exact source of the problem. This was one problem in the Atelier B approach: when a proof fails it is very difficult to find out why the proof has failed, especially when large and complicated constants are present. To address this issue, we have developed an algorithm to inspect the truth-values of B predicates, as well as all sub-expressions and sub-predicates. The whole is assembled into a graphical tree representation.

In summary, the outcome of this case study was extremely positive: a man-month of effort has been replaced by a few minutes of computation on a laptop. Siemens are incorporating ProB into their development life cycle, and

they are hoping to save a considerable amount of resources and money. For this, certification of the ProB tool is an important aspect, which is discussed in [LFFP10] and which is still ongoing work.

4.3 Paris Line 1 Deployment

In October 2009, ProB was applied for the first time on an active development, concurrently to the classical approach using Atelier B and manual inspection.

Siemens is involved in the automatisation of the Line 1 of the Paris Métro. The line will be gradually upgraded to driverless trains, while the line remains in operation. So far, we have inspected the first component to be delivered by Siemens, namely the PAL (Pilote Automatique Ligne). The B models of the PAL consisted of 74 files with over 10,000 lines of B. In all 2024 assertions about the concrete data of the PAL needed to be checked.

Again, ProB found all problems (12 in all) in under 5 minutes. These problems have of course been examined and fixed by Siemens before delivery. There were again no false alarms or mistakes by ProB, compared to the validation done by Siemens.

Initially, few minor tweaks were required to get the models through our type checker. Indeed, some of the models introduced definitions which were overwriting visible constants from included machines. ProB did consider this to be an error, whereas Atelier B accepts this. This has been fixed now: ProB only generates a warning in those cases. Now, the PAL models can be loaded without modification into ProB.

Also note that we had to improve the ProB kernel for a few operators not yet encountered in the San Juan models (such as the `iterate` operator for relations, which did not yet use our new data structure for large relations). This took only about a couple of hours.

In response to a Siemens requirement, we have also made assertion checking possible from within a command-line version of ProB. This allows to run our tool in batch mode on a large number of files, and collect the results.

4.4 São Paulo Line 4

The next task was the validation of the CBTC Ground controller for Line 4 of São Paulo, which began operation in May 2010.

UDUS received the models on March 11 2010, because Siemens was unable to validate one crucial property (neither with Atelier B, nor by "hand").

The models consisted of 210 files with over 30,000 lines of B and over 2500 assertions. It then took about a day for UDUS to validate the crucial property using ProB.

The reasons the validation took a day rather than minutes were the following:

- The generated B models contained syntax errors and locating the error within 210 files was difficult. The parser of ProB has now been improved to also output information about the file containing the errors. The syntax errors (missing semicolons after definition file imports) were not found by Atelier B due to a bug in the parser. Siemens have now fixed their tool to avoid those syntax errors in the future.

- The models highlighted performance and memory issues with some of the B operators. Indeed, these operators were not required in the previous case studies and were not yet using the new data structures (meaning that the AVL tree representation for sets were expanded into the old list representation). The ProB kernel has again been improved to deal with those operators.

- The models contained many inconsistencies inside the properties. This makes starting ProB more difficult. Note that the Paris Line 1 models from Section 4.3 contained no inconsistencies, and the San Juan case study contained only a single inconsistency.

  To address this issue better in future applications, ProB now partitions the properties into independent components and detects inconsistent components. This helps the user to isolate the problem more quickly.

Some of the other B machines of the São Paulo Line 4 made use of infinite "complement" sets (e.g., setting `s = INTEGER - {x}` and then later checking `y:s`). ProB now detects those infinite complement sets and keeps them symbolic, even if ProB is not in symbolic mode. There is also support for performing certain B operations on those complement sets (union, intersection, membership test, ...). All of the additional improvements to ProB took about a week to implement.

4.5 Barcelona Line 9 and CDGVAL

After the delivery of the São Paulo line, ProB was applied in May 2010 to validate the data of the zone controller of the Barcelona Line 9, which "will be one of the longest automatic metro lines in Europe."[2].

[2] `http://en.wikipedia.org/wiki/Barcelona_Metro_line_9`

First, ProB detected syntax errors in the definitions of the generated B machines. For example, one definition file contained the following line, where an operator is missing:

```
co_nb_max_heure_il_par_pas  ==   2 ** 24  1;
```

These errors were not detected by the Atelier B parser, as the offending definitions were not actually used in the rest of the model.

Once the syntax errors were corrected, the data validation was performed independently and with success by Siemens. As a minor improvement, we made sure that ProB tries to detect infinite lambda expressions and set comprehensions, ensuring that those are not expanded, even if ProB is not in symbolic mode. For example, the following two lambda expressions appear in the Barcelona Line 9 models:

```
abs = %xx.(xx : INTEGER | max ({xx, -xx})) &
sqrt = %xx.(xx : NATURAL | max({yy | yy : INTEGER & yy * yy <= xx}))
```

After that, Siemens used ProB for data validation for the CDGVAL (Charles de Gaulle Véhicule Automatique Léger) automated shuttle at the Charles de Gaulle airport in Paris. The validation was performed completely independently and successfully by Siemens; no adaptation of ProB was performed.

4.6 Summary

Overall, Siemens has successfully applied ProB with most CBTC projects. The experience with data validation illustrates that the support from UDUS in terms of using ProB is adequate for Siemens.

Chapter 5

Space Sector

The following chapter contains an evaluation report prepared by Space Systems Finland (SSF) on the training undertaken during the last year of the DEPLOY project. The purpose of this training was to find how much effort is needed for an engineer without background in formal methods to become a competent user of Event-B, and to evaluate the currently available training material, documentation and tools. The work also provides input for analyzing some hypotheses (TSP-HM-1, TSP-EA-1) [DEP10a] concerning the use of formal methods.

During the performed training, two engineers without previous background in formal methods have learned to use Event-B in the Rodin Platform. As a result of their studies, they were able to independently build simple models. At a later stage, more complex models have been built by using the Modularization Toolbox. The Record Types plug-in has also been studied to some extent, however the modeling work was mainly limited to learning from the existing examples.

Depending on the engineer, it took about one man-month (1.3 and 0.9 months, respectively) of a partly tutored study to be able to work with reasonable independence. By "reasonably independent", we mean an ability to design and implement simple models and also to learn more by consulting documentation and occasionally an expert. The tools used in the learning process (Rodin Platform and the associated plug-ins) were generally considered good for their purpose, but the available training material appeared to be incomplete. The use of the Modularization Toolbox seemed to help in building more complicated models, but, at least from the beginners point of view, it also complicated the modeling process.

5.1 The Knowledge Transfer Context

Ms. Nummila Ms. Nummila is a software engineer having a M.Sc. degree in computer engineering. Her university studies have been focused on embedded systems and software production.

Except for an introductory university course in formal logic and fundamental structures, Ms. Nummila has no background in the use of formal methods. Most recent work experience is on onboard software component implementation, unit and validation test design, unit and validation test implementation, and quality assurance in high reliability software projects. She has been working at SSF since October 2006.

Mr. Räsänen Mr. Räsänen is a software engineer with a M.Sc. degree in medical physics and electronics. His studies have have been focused on digital signal processing. Mr. Räsänen has no formal background in Computer Science. His programming skills are a result of a long practice in building computer-controlled measurement and test systems. At the time of the study, he had worked about 11 years at SSF, doing mainly functional testing of space software. This work has included building test environments. Recently, he has also worked on qualifying an embedded system according to the industry standards. The current study is his first contact with formal methods.

5.2 Training Experience

5.2.1 Tools, Training and Assistance

The basic modeling environment was the Eclipse-based Rodin Platform. The platform is continually being updated to answer the needs of the users and the development of Event-B.

The Rodin Platform has an internal Event-B editor. It is interactive, i.e., the user can create templates for different Event-B language components and fill them in. The editor also contains a Pretty Print – a tool that displays the Event-B code with all its special symbols.

Another alternative for editor is Camille that looks and behaves like a text editor with syntax highlighting. Since spring 2010, two new plug-ins have been added: Modularization and Record Types. Some documentation for them can be found in the Event-B Wiki [eve].

The training material has consisted of the Event-B Wiki [RD4], the teaching material from the training course held in Zürich in April 2008, World

Wide Web, and general University teaching material about formal logic. In particular, the Zürich course contains lectures, exercises, summary presentations and Event-B models. Some further material about the B Method have been used as a reference.

The assistance from more experienced Event-B specialists was needed, especially in the beginning of the training period. From SSF, Dr. Dubravka Ilic and Dr. Timo Latvala were the persons to contact for help. Also Prof. Elena Troubitsyna and Dr. Linas Laibinis from Aabo Akademi (Aabo) were available for assistance via email.

5.2.2 The Effort in Different Phases of the Work

Timeline and Events – Mr. Räsänen

In the early period of the training, Mr. Räsänen has consulted colleagues, who had experience with Event-B and the Rodin Platform, particularly Dr. Ilic. He has acquired the above mentioned learning material gradually, over the early spring 2010 (the Modularization Toolbox documentation became available later).

04.01.2010 Mr. Räsänen joined the project, and started the studies by installing the Rodin Platform and learning the basics of its use. The work went on intermittently, other projects allowing, and continued in gathering and reading different types of the Event-B documentation. A major part of the work was studying lecture notes of available examples, and at the same time using them as an extensive tutorial of the Event-B language. This phase took the effort of 66.5 hours, or about nine days.

24.03.2010 Mr. Räsänen continued training by trying out in the Rodin Platform a complete example that was described in the lecture on Bounded Retransmission Protocol. The work was finished by discharging all the generated proof obligations. This took 32.3 hours, or about 4.5 days.

06.04.2010 The next phase for Mr. Räsänen was to independently build an own model based on the packet exchange protocol taken from another project. This phase included a 3 hour tutorial session, where more advanced Event-B techniques were explained. The model in question has been repeatedly rewritten. The resulting development consisted of the initial models and two of its refinements. This phase took 111.2 hours, or about 15 days.

05.05.2010 A training meeting with Prof. Troubitsyna and Dr. Laibinis from Aabo Akademi was arranged. Mr. Räsänen had a presentation explaining his progress so far, as well as some analysis of the learning process.

06.05.2010 Mr. Räsänen has continued by further developing the previously mentioned packet protocol model. This phase took 22.5 hours or about three days.

20.05.2010 An additional training session organized by Prof. Troubitsyna and Dr. Laibinis. The training was focused on basic concepts of Event-B and the Rodin Platform as well as the Modularization Toolbox.

24.05.2010 Mr. Räsänen studied the Modularization Toolbox, using the material delivered during the meeting on 20.05.2010 (in particular, the parking gate example). This phase included downloading the model implementing the example and investigating it in the Rodin Platform. The effort of this phase was 24.4 hours, or a bit over three days.

Timeline and Events – Ms. Nummila

Ms. Nummila joined the project at a later phase in May 2010. Due to Dr. Ilic's commitments to other projects, she was not available for consultancy, and the learning process was mainly self-study. In case severe problems were encountered in the modeling work, Dr. Laibinis and Dr. Latvala were available for getting the needed assistance.

05.05.2010 Meeting: Ms. Nummila joined the DEPLOY project by attending a meeting with researchers from Aabo Akademi, Prof. Troubitsyna and Dr. Laibinis. The meeting mainly concerned the basic training and planning of the future activities.

18 – 19.05.2010 Training: To prepare the following training session, Ms. Nummila spent 14.5 hours installing the Rodin Platform and getting familiar with the available documentation (Event-B Wiki, university teaching material).

20.05.2010 Meeting: A training session organized by Prof. Troubitsyna and Dr. Laibinis. The first part of the training consisted of learning the basics of the Rodin Platform and building a couple of simple models with some assistance from Dr. Laibinis. The latter part was concentrated on a presentation of a modular model description (the Parking Lot example).

21.05 – 28.06.2010 Modeling: Based on the description of the modular model, Ms. Nummila has first built a monolithic model of the Parking Lot. When the monolithic model was ready and she was more experienced in using the Rodin Platform and the Event-B language, the modular plug-in was installed and a modular model of the same Parking Lot example has been built. The work was quite independent, but a few questions were asked from Dr. Laibinis and Dr. Latvala about some problems faced in the use of the Rodin Platform and the Modularization plug-in. The amount of time used for the building of the two models was 109 hours or approximately 14.5 days.

Current Status

The engineers have acquired an ability of independently developing Event-B models, and implementing them in the Rodin Platform. As a result, they were able to study further the Modularization Toolbox, and to use it, at least slowly in the beginning, in developing more complex models.

5.3 Assessment

Training Material The lecture material and presentations in the Zürich Course are good in explaining many different aspects of Event-B and showing its use by examples. The examples are also implemented as ready-to-use models.

Event-B Wiki is a fairly comprehensive tutorial for both the Rodin Platform and the Event-B language, however the documentation for some of the latest plug-ins (e.g., record types, team-working) seems unfinished. Any information not available in the Event-B Wiki was very difficult to find elsewhere either.

At the present, a beginner will have difficulties in deciding where to start. An outline telling about the general features of Event-B, and showing from which material to start, would be useful for such a novice user.

A systematic reference material going through all features of the Event-B language would also be very helpful. During the work, no description was found for EQL, a fairly common proof obligation. This was the only obvious omission, although it was rather difficult to find detailed information about other proof obligations as well.

Tools The basic features of the Rodin Platform are easy to use, once the user is familiar with the Eclipse platform (it is a good idea though to go through the Eclipse tutorial first). The Event-B Editor is interactive, and supports all new features (e.g., modularization). Camille is quicker to use for more experienced people, and the model text is much easier to read than in the Event-B Editor or Pretty Print. On the other hand, Camille does not show all symbols used by Event-B. Currently the Modularisation Toolbox is not compatible with Camille or Pretty Print.

The Rodin Platform, and the tools and documentation incorporated into it, follow closely the development of Event-B. However, the tools and the documentation have not had the time to catch up with the development of the method. Some issues worth noting are:

- Some minor practical matters about the use of the platform seem to be omitted that are self-evident to the writer of the documentation, but far from obvious to a novice user (e.g., how the action numbering does not start from zero in a refined Initialization Event, etc).
- The Camille editor has some stability problems. One must also be very careful, when copying and pasting; the structure of the model text may get mixed. While encountering mysterious errors, it is worthwhile to check in the Event-B Editor how the model actually looks like.
- The Rodin Platform Help is not quite complete. This is particularly obvious in the Proof Control part.
- At least on Windows, the Rodin Platform sometimes gets frozen. The usual cause for this was having several machines/contexts open at the same time, and then trying to expand one of them. The only recovery is then restarting the whole platform.

Notable Difficulties At the beginning, it is difficult to learn the right approach. A programmer's instinct is to start from a basic functionality, and expand the system to have more and more properties. On the other hand, the approach of Event-B is to look at the whole system from the beginning, first from a very general, abstract outline. The general, abstract features are then gradually filled in with finer detail.

It is not easy to find information for a particular need from the current material. The most effective way is often to search the Internet for answers; this tends to lead to Event-B Wiki. Sometimes the answer is not found. At the current stage of the documentation, help from a more experienced user is vital. One advantage brought by the graphical user interface is that some

of the problems could also be solved with the trial and error method. Of course this is more time consuming, but is also the only option left if the documentation does not provide the answers and no experienced colleagues are available for consultancy.

While working on the modular model, some of the Rodin problems (errors/warnings) were difficult to track. The error messages were too vague, and did not clearly specify in which part of the model the problem was. After gaining some more experience of using the modularization plug-in, it became easier to associate the errors with their origin, but this was only based on simple knowledge of which trigger to pull to make a certain error message disappear.

Suggestions A comprehensive Event-B and Rodin Platform reference manual, including, e.g., the complete Event-B notation and all types of proof obligations fully explained, would be very useful, especially when the learning user begins to gain independence. This reduces the need to consult an expert every once in a while. The documentation would need to be organized so that the links to all the different topics can be found from a central chapter or a document.

The Event-B wiki page has a short FAQ, but it might be useful to expand it by adding some generally known problems in the use of the Rodin Platform. This would help the new users to overcome some of the difficulties the others have already encountered.

Proving is largely an automatic mechanism, but it is central to Event-B. For novice users, it is largely a black box. Even somewhat more advanced users cannot yet understand how the proof obligations get proven. Therefore, one should put a special emphasis on the documentation about proving and of the Proof Control part of the Rodin Platform.

It should be noted that, as both Event-B and the Rodin Platform are still under development, it is understandable that the documentation for all the latest plug-ins is not always up-to-date. Anyway, from the novice user's point of view, it might be better to omit the new features until their documentation has been completed. On the other hand, it is understandable that the researchers working on the development of Rodin Platform want to get the latest updates from everybody as soon as possible, documented or not. This problem could be solved by clearly declaring which plug-ins are suitable for a regular user with all the existing features explained in their documentation. The newly added plug-ins that are still beta versions could still be available for the developers to test and improve on the latest additions, while the novice user would not get confused about unfinished features.

Learning Effort It took about a month's (respectively, 1.3 and 0.9 months) working time to learn Event-B and Rodin Platform well enough for reasonably independent work. Since both students were at the time occupied by other projects, the overall calendar time was about four and two months respectively. This information has been used to give numerical values to the hypothesis TSP-HM-1.

Hypotheses about Event-B **TSP-HM-1-Hyp3** *For developers and analysts with no background in FM, a training programme and/or individual coaching needed to become autonomous with Event-B modeling and proving takes at most XXX months and requires YY% of effort (per engineer) over that period.*

The participants of the current study used approximately 2 – 4 months with 50% - 25% effort to achieve a reasonable independence.

TSP-EA-1-Hyp2 *An engineer with no or little FM background takes at least 6 months to become autonomous when spending at least 50% of her/his time on training and practicing with a new formalism. (Some advanced training or monthly individual coaching from experts will also be required during these 6 months).*

The experience from the current study suggests that the time of 6 months can be cut into half. Consultation with an expert user will occasionally be necessary in order to obtain required advice and feedback.

5.4 Summary

It has taken in total about one working month to attain the ability of working with Event-B and Rodin Platform in a reasonably independent fashion. A consultation with expert users is still needed at this level.

Almost all necessary information can be found from the documentation listed in Section 5.2.1 but searching for it can be quite difficult. A more organized documentation is needed, including a full reference manual, where all the details of Event-B and properties of the tools can be found. This will help disseminating Event-B and its tools to wider circles, as the need for expert consultation is reduced.

The skills learned during the period of this study may be of use, when designing a complex system. It is not possible, however, to estimate the benefits with the present level of experience.

Chapter 6

Business Information Sector

This chapter reports on the training experience in the business information sector, i.e., for SAP. The challenge for deployment at SAP is to integrate formal methods with the existing development process. This is a highly interesting area for applying formal methods because properties of business software are not necessary safety critical which are the traditional application domain for formal methods. Nevertheless, business software still needs to be developed efficiently with high quality assurance.

6.1 The Knowledge Transfer Context

Initially, only Andreas Roth from SAP attended the block course in Zurich (2008), when he already had significant experience in formal specification and deductive verification. The course helped Andreas to understand the Event-B method and to use Rodin Platform for modeling. Later, Vitaly Kozyura joined the DEPLOY team at SAP and received training on Event-B and Rodin Platform directly from Andreas with some additional supports from ETHZ. This initial training proved to be adequate, in particular for pilot deployment focusing on the translation of Message Choreography Models (MCMs) into Event-B, in order to verify the consistency between the global and local models [DEP10b].

Since mid-2009, Wei Wei has joined the DEPLOY team at SAP and also received "in-house" training on Event-B and Rodin Platform from the other members of the SAP teams. A two-day visit to ETH has helped Wei to consolidate his understanding of Event-B, in particular of system development using refinement, and to do interactive proofs with the Rodin Platform.

Since October 2010, the role of the DEPLOY project lead at SAP has been handed over from Andreas Roth to Sebastian Wieczorek. Sebastian has

vast experience in conducting technology transfers and is also an expert on model-based testing.

6.2 Training Experience

Application of Event-B to the mini-pilots and the pilots from SAP showed that Event-B can be used in verifying certain consistency of message choreography models. In particular, plug-ins such as UML-B and ProB have inspired the integration of Rodin Platform with the diagrammatic front-end for Message Choreography Models (MCMs). The result has been reported in our early deliverable [DEP10b]. This report focuses on the technology transfer on three different lines of on-going work at SAP: verifying Business Process Modeling Notation (BPMN) models (Section 6.2.1), model-based testing (Section 6.2.2), and checking consistency between message choreography and their implementation models (Section 6.2.3).

6.2.1 BPMN

The main goal of the work is to be able to formalize BPMN models in Event-B and to verify specific properties of these models. Initially, the work started with collaboration between SAP and Newcastle. Later, ETHZ and Southampton were also involved in further investigating different approaches.

Technology transfer is mainly done by having academic and deployment partners working together on some exemplary case studies. These case studies have dual purposes. On the one hand, they serve as illustrated examples for SAP to evaluate the applicability of Event-B and the Rodin Platform to the problem. On the other hand, they help academic partners to understand domain specific problems which could require further enhancement of the available methods and tools. Below, we give a summary of the exemplary case studies and the application of modularisation approach, focusing on the aspect of technology transfer.

The Exemplary Case Studies Two separate lines of case study activities were conducted at SAP and at Newcastle University. At SAP, the main purpose was to identify the set of most used BPMN features could be modeled and verified using Event-B. At Newcastle University, a further investigation was made to see how their previous work on a timed error recovery pattern [BFRR10] can be applied to BPMN analysis.

At SAP, a number of BPMN models were constructed to represent use cases in different areas of business process management, such as purchasing,

factory management, banking services, etc. These models were translated into Event-B using various strategies in order to find out which translation approaches might lead to better model structures in Event-B and easier verification that requires less human intervention. These experiments resulted in a mechanic translation procedure that produces Event-B models with good readability and provability. The design of the translation was independently carried out at SAP, with improvements thanks to some useful feedbacks being obtained through 2-day discussions with ETH partners.

At Newcastle University, the Event-B pattern of timed error recovery was applied to a model of credit request processing. The original model was published in [PQZ08] and slightly modified to fit the goal of pattern application. The preliminary result of this case study was already described in [BFRR10].

During the development of the case studies, both theorem proving and model checking techniques are applied. On the one hand, ETH provided some useful proving strategies for interactively discharging proof obligations using the Rodin Platform. On the other hand, the ProB model checker was used to find counterexample for a violated property.

SAP came to recognize the following problems via analyzing the resulting formal models: (1) The translation procedure needs to be improved by using proper decomposition techniques, in order to better reflect the structure of the original BPMN models and to allow compositional verification. (2) We need to develop more automated solutions for the analysis of BPMN models. (3) It is still too demanding for the verification of the BPMN features that we currently consider. One solution is to further narrow down the part of BPMN to be analyzed. Alternatively, a better understanding on how and when different decomposition techniques can be applied might help to solve the above problems.

Modularisation The modeling exercise done by SAP has some interesting relations to model and proof structuring. A source protocol diagram already defines some structure and along it visibility scopes that can be turned into model variables in Event-B. Naturally, it was interesting to see how this structure can be depicted in an Event-B development. SAP has conducted a number of experiments with various decomposition techniques among which the modularisation approach was deemed more suitable for the problem. One interesting outcome is some experience in doing a parallel development using different decomposition styles. At the moment, it is too early to draw any conclusion on the comparative strengths of the methods, however it was agreed that SAP together with ETHZ, Southampton and Newcastle would

investigate further. It is important to give guidance to potential Rodin Platform and Event-B users on how to choose a decomposition style best suited for their problem.

Initial training was self-education by following the parking lot tutorial slides (available from the event-b.org repository) and the plug-in wiki. A half-day training session on the use of modularisation was given in September 2010 and afterwards SAP were largely independent in their modeling efforts. Prior to that, there was no real output with modularisation style approach. It is important to note here that neither decomposition technique is supported by a comprehensive user reference and hence education by reading examples and research papers is a slow and demanding process. Within the project this can be addressed by training sessions. Outside of the project, one should not expect a widespread pick-up of decomposition tools.

SAP has provided some feedback on the modularisation plug-in and this has been largely addressed in the version released in Nov 2010. There was also feedback and a wishlist for the new version of the Flow plug-in.

An important point for Newcastle is to learn how the modularisation decomposition is applied and what are the typical mistakes made during decomposition. One possibility is that the modularisation technique may be narrowed down to a more limited mechanism tailored to the kind of modeling done by SAP.

6.2.2 Model-Based Testing

SAP Research developed a language for modeling service choreographies called Message Choreography Modeling (MCM) [WRS+09]. It basically consists of (1) an abstract *Global Choreography Model* which specifies a high-level view of the transformation between service components, (2) a *Local Partner Model* which describes the behavior of a single component and (3) a *Channel Model* which models the communication between partners.

SAP Research implemented a conversation from MCM to Event-B. Based on ProB, they also implemented an animator for their formalism by transforming the results of ProB from Event-B back to the original model.

SAP is interested in generating test cases that can be automatically derived from the model. To achieve this, UDUS implemented two functionalities:

1. Generating traces (i.e. sequences of events) for the global choreography model. Each trace begins with the initialisation of the system, covers a specific event and ends in a state fulfilling a user-given property.

2. For a given trace of the global choreography model, ProB tries to find corresponding events in the local partner model.

[WKR+09] describes the approach in more details.

Beside communication via email and telephone, the UDUS team visited SAP Research in Darmstadt on January 1 and April 4, 2009. For the UDUS's team members, the aim of the meetings was to understand SAP's approach to choreography models and their requirements for the generated test cases. For SAP's team members the aim was to get a deeper insight into how ProB can deal with the models and how its API (application programming interface) can be instrumented to support animation and test generation.

Further work is aiming to combine the two steps outlined above and to refine SAP's requirements to test case generation and incorporate them into ProB and SAP's tools.

6.2.3 Consistency Between Message Choreographies and Their Implementation Models

Developing business applications depends heavily on layered design and implementation. One of the goal of SAP is to verify the consistency between such different layers. In particular, the consistency between Message Choreography Models (MCM) and their corresponding implementation models (represented by Business Objects) can be verified by translating both of these models into Event-B and proving the refinement relationship between them. We focus here on the interaction related to technology transfer for the work. More detail report of the approach is available in [KRWW10].

The work has been done independently by Vitaly at SAP with some supports of Michael Butler from Southampton. The communication is mostly done via email, in particular on the optimization of the Event-B models. Different suggestions for improving the models include:

- Using parameters instead of non-deterministic action.
- Using implications instead of disjunctions in specifying conditions on the after-state.
- Using set theoretical operators in place of quantified first-order predicates.

The result of the optimization is that not only the model is more readable but also the automatic proving rate increases dramatically: from almost no automatic proofs to 90% of the total proof obligations proved automatically in the experimental model.

6.3 Summary

In summary, SAP's approach in DEPLOY is to integrate Event-B and its associated Rodin Platform within the existing development process. Whereas the initial result is encouraging, there are still several on-going investigations on applying formal methods to business software developments.

Some of the experiences in technology transfer in this particular area include:

- training on Event-B and Rodin Platform in general can be passed on with additional support from experts;
- it is important to develop method and tool targeting domain-specific problems, e.g. automatic invariants generation;
- it might be necessary to have domain experts and formal method experts working together to achieve better results.

Chapter 7

DEPLOY Associates

During 2010, training was provided to two DEPLOY Associates: Critical Software Technologies Ltd., Southampton, UK and AeS, São Paulo, Brazil. The University of Southampton has provided training courses to groups of employees f rom these DEPLOY Associates at their business premises.

7.1 Event-B Training

Training on the Event-B Language and Rodin Platform was provided for both DEPLOY Associates included the following sessions. The training material is available on-line [But10].

An introductory session discussed the benefits of using formal specifications such as the cost benefits of detecting errors earlier in the design process. The Event-B specification language was then introduced emphasizing the usefulness of abstracting the essential concepts of a problem and developing the full specification through refinements. The Rodin Platform and the DEPLOY project were introduced.

The training then consisted of introducing the Event-B language in stages starting with sets, then introducing relations and then functions. The mathematical basis of these concepts was provided and the modeling concepts were illustrated with examples.

An example problem was then attempted in Event-B by the trainees. The example involved modeling a shipping port docking system using the modeling concepts introduced in the previous sections.

The basis behind consistency proofs was then explained in terms of the preservation of invariants and how this can be established from the substitution given assumptions about the invariants and the guards of the event making the substitution.

Finally, the concept and techniques of refinement were explained in the context of the Event-B language.

7.2 UML-B Training

For AeS only, training on the use of UML-B was also provided [Sno10]. After an introduction to the concept and motivation behind UML-B, the training was arranged into three sections.

Firstly, UML-B class diagrams were introduced including an explanation of the concept of lifting models to a set of instances and how it relates to Event-B modeling.

Secondly, the use of UML-B state-machines were explained focusing on how to model behavior and how this can be used with or without classes. The semantics of state-machines were illustrated by showing their translation to Event-B.

At this stage, an example problem was attempted in UML-B by the trainees. The example was a simplified version of a real case study involving a railway interlocking system and the preservation of safety invariants.

The final section described how refinement can be carried out in UML-B including refinement techniques on class diagrams and refinement of state-machine diagrams by adding nested state-machines.

7.3 Feedback from the DEPLOY Associates

Critical Software Technologies: The training sessions provided gave Critical Software Technologies a good understanding of the concepts associated with Formal methods (in general) and Event-B. The various examples and case studies provided made us think in more abstract terms and allowed us to practice and understand the implementation side of the Event-B and the tool Rodin. The duration and periodicity of the sessions was good and together with the materials provided gave us the means to develop the pilot work. Overall very good!

AeS: The training course was held in AeS, São Paulo, over a single week. During the first two days we had a good overview of the mathematical basis, as well an introduction of Event-B. The last days were dedicated to some examples on the application of Event-B and an introduction on UML-B plugin. At the end we were able to apply what was presented in a simple real example on the railway domain. It is important to tell that we had around 12 attendees, most of them engineers with no formal method background,

and at the end most of them succeeded in the tasks presented. Moreover, even if not all trainees are using the method itself nowadays, the concept presented (correct-by-construction, blueprints analogy, etc) changed the way of thinking during the development process. I can tell that the objective was achieved. The material and professors were perfect and prepared everything to meet our needs.

Bibliography

[BFRR10] Jeremy W. Bryans, John S. Fitzgerald, Alexander Romanovsky, and Andreas Roth. Patterns for modelling time and consistency in business information systems. In Radu Calinescu, Richard F. Paige, and Marta Z. Kwiatkowska, editors, *ICECCS*, pages 105–114. IEEE Computer Society, 2010.

[But10] M. Butler. Slides on Event-B used for DEPLOY associate training. http://deploy-eprints.ecs.soton.ac.uk/264/, 2010.

[DEP10a] DEPLOY Project. D11.3 Initial Evidence Repository, February 2010.

[DEP10b] DEPLOY Project. D4.1 report on pilot deployment in business information sector. http://www.deploy-project.eu/pdf/D21-pilot-deployment-in-business-information-software(4).pdf, February 2010.

[eve] event-b.org. Event-B wiki. http://www.event-b.org.

[HH10a] Stefan Hallerstede and Thai Son Hoang. Bucharest DEPLOY 2-day course. http://deploy-eprints.ecs.soton.ac.uk/238/, July 2010.

[HH10b] Stefan Hallerstede and Thai Son Hoang. Post-material for Bucharest DEPLOY 2-day course. http://deploy-eprints.ecs.soton.ac.uk/241/, July 2010.

[HH10c] Stefan Hallerstede and Thai Son Hoang. Pre-reading material for the Bucharest DEPLOY 2-day course. http://deploy-eprints.ecs.soton.ac.uk/234/, July 2010.

[KRWW10] V. Kozyura, A. Roth, S. Wieczorek, and W. Wei. Checking consistency between message choreographies and their implementation models. In *Proceedings of the 10th International Workshop on Automated Verification of Critical Systems*, 2010.

[LFFP10] Michael Leuschel, Jérôme Falampin, Fabian Fritz, and Daniel Plagge. Automated property verification for large scale B models. *Formal Aspects Computing*, 2010. Accepted for publication.

[PQZ08] Davide Prandi, Paola Quaglia, and Nicola Zannone. Formal analysis of BPMN via a translation into COWS. In Doug Lea and Gianluigi Zavattaro, editors, *COORDINATION*, volume 5052 of *Lecture Notes in Computer Science*, pages 249–263. Springer, 2008.

[Sie09] Siemens. B method - optimum safety guaranteed. *Imagine*, 10:12–13, June 2009.

[Sno10] C. Snook. Slides on UML-B used for DEPLOY associate training. http://deploy-eprints.ecs.soton.ac.uk/265/, 2010.

[WKR+09] Sebastian Wieczorek, Vitaly Kozyura, Andreas Roth, Michael Leuschel, Jens Bendisposto, Daniel Plagge, and Ina Schieferdecker. Applying model checking to generate model-based integration tests from choreography models. In *Proceedings TESTCOM/FATES 2009*, volume 5826 of *Lecture Notes of Computer Science*, pages 179–194. Springer-Verlag, 2009.

[WRS+09] Sebastian Wieczorek, Andreas Roth, Alin Stefanescu, Vitaly Kozyura, Anis Charfi, Frank Michael Kraft, and Ina Schieferdecker. Viewpoints for modeling choreographies in service-oriented architectures. In *WICSA/ECSA*, pages 11–20. IEEE, 2009.