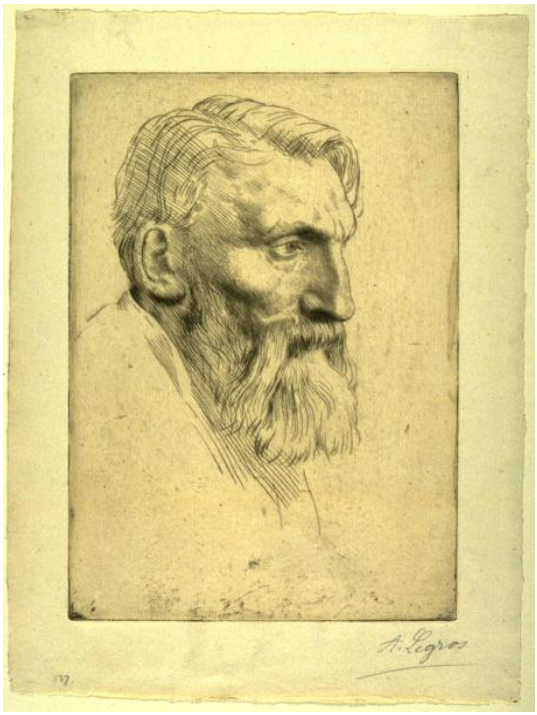
Project IST-511599
RODIN
"Rigorous Open Development Environment for Complex Systems"



RODIN Deliverable D2

Definitions of Case Studies and Evaluation Criteria for Case Studies

Adrian Hilton(Praxis High Integrity Systems, UK) , Ian Johnson (VT Engine Controls Ltd, UK), Cliff Jones (University of Newcastle upon Tyne, UK),
Sari Leppänen (Nokia, Finland), Ian Oliver (Nokia, Finland),
Alexander Romanovsky(University of Newcastle upon Tyne, UK),
Elena Troubitsyna (Aabo Akademi University, Finland)

Public Document

30th November 2004

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

Contents

1	Introduction……	2
2	Case study 1: Formal Approaches to Protocol Engineering ……..…….	3
3	Case study 2: Engine Failure Management System ……..…….	7
4	Case study 3: Formal Techniques within an MDA Context ……	11
5	Case study 4: CDIS Air Traffic Control Display System……	16
6	Case study 5: Ambient Campus …….....	19
7	Generic Evaluation Criteria …….	23

1. Introduction

This document presents the definition of case studies of the project and evaluation criteria for assessing RODIN methodology. We also define evaluation criteria for assessing the contribution of each case study. The aim of case studies is to drive the development of RODIN methodology and supporting platform, validate it and evaluate its cost-effectiveness. To attain this goal, in D2 the partners define their case studies and identify the key criteria indicating success of RODIN methodology. To give a clear understanding of problems to be tackled, the description of each case study includes a brief introduction into the problem domain, the available background material, the case study setting and the list of evaluation criteria specific to the corresponding domain area. Moreover, this document also identifies a set of evaluation criteria common to all case studies in the project.

The case studies should consolidate the efforts of project partners in developing the unified methodology and supporting tools. This document sets a scene for conceptualizing the problem areas and joining partner's forces to develop methods and tools tackling them. Moreover, it also defines the essential criteria guiding this development. The research questions and evaluation criteria defined in D2 will be further refined and extended in duration of the project.

2. Case Study 1: Formal Approaches to Protocol Engineering

2.1. Background

The *Lyra* method developed within the protocol engineering group at Nokia Research Center supports model-based approach in the development of distributed communicating systems and communication protocols. The method covers all stages of the development. It supports service-oriented approach and the main technique used in the method is model decomposition/composition. The method consists of the four phases: Service Specification, Service Decomposition, Service Distribution and Service Implementation. The Service Specification model(s) providing the correctness criteria for later development phases is verified using model-checking techniques. In the following development phases, algorithmic verification is currently used to verify the correctness of the decomposition and composition steps.

The case study will be centered on development of a *Position Calculation Application Part (PCAP)* specified by the *Third Generation Partnership Project (3GPP)*. PCAP is part of the *User Equipment (UE)* positioning system in a *UMTS (Universal Mobile Telecommunications System)* radio access network. PCAP is specified to manage the communication related to positioning service between the network elements *Radio Network Controller (RNC)* and *Stand-alone Assisted Global Positioning System Serving Mobile Location Center (*SAS).

2.2. Current State of Work

Lyra development method has been described in a PhD thesis draft and a series of articles. Some of the articles have already been published and the rest will be published during year 2005. PhD thesis will be published in the first half of year 2005. Case study on using UML2 [4] and Lyra in modelling the 3GPP Positioning system and PCAP has been published as an article of FDL'04 conference [5]. This case study considers only the design part of Lyra and does not consider verification or testing of design.

The model of PCAP in RODIN should cover the same set of functionality as the model in [5]. This set has been chosen so that it represents a typical example of an industrial application and allows covering of all relevant development phases.

Before starting the actual case study, knowledge on the Lyra method has to be first transferred to project members. For teaching the Lyra method a technical report with a simple example will be extracted from the PhD thesis draft by the mid-November 2004. An article on Lyra/PCAP modelling is already available for project members. The 3GPP specifications [1] and [2] are also accessible through the www-pages of the 3GPP organization. More material on the method will be delivered to project members later on.

2.3. Available Material

3GPP Specifications related to Positioning and PCAP:

[1] 3GPP. Technical specification 25.305: Stage 2 functional specification of UE positioning in UTRAN.

[2] 3GPP. Technical specification 25.453: UTRAN Iupc interface position calculation application part (pcap) signaling.

UML2 language and tools:

[3] www.omg.org/technology/documents/modeling_spec_catalog.htm

[4] Telelogic TAU Generation2. www.taug2.com.

Case study on modelling 3GPP Positioning System and PCAP with UML2 and Lyra:

[5] S. Leppänen, M. Turunen and I. Oliver. Application Driven Methodology for Development of Communicating Systems. In Forum on Specification and Design Languages 2004 (FDL'04). ECSI, September 2004, Lille, France.

2.4. Formal Methods Experience in Nokia Research Center

Research on applying formal methods in protocol engineering has been going on in Nokia Research Center (NRC) since 1997. In close co-operation with various universities NRC has developed several verification and formal testing tools (see e.g. www.nokia.com -> *About Nokia* -> R*esearch* -> *Research Projects* -> *Europe*). These methods and tools, and also some other existing tools and methods, have been evaluated with numerous case studies. The example protocols for the case studies have been taken from the systems currently under development, mostly from the 3G Radio Access network protocol suite. The results of the case studies have been reported at international conferences and contributed in 3GPP standardization. Integration of formal methods with advanced development languages and methods (e.g., SDL, UML 2.0) was started around 1998 in collaboration with the related standardization bodies and tool vendors.

2.5. Expected Contribution to RODIN

In Lyra the Service Specification model(s) providing the correctness criteria for the later development phases is verified using model-checking technique. In the following development phases algorithmic verification techniques are used to verify the correctness of the decomposition and composition steps. However, telecommunication systems tend to be very large and data intensive so that the use of algorithmic verification (e.g. model checking) is prone to the state explosion problem.

Nokia is interested in investigating the use of refinement techniques to prove and automate the decomposition steps. Important issue will also be combining of top-down (refinement) and bottom-up (algorithmic verification) verification techniques in the development of communicating systems and communication protocols. Another important research focus will be on using proof techniques in automated data abstractions. This would enable automated transformation of design models to verification and testing models. Also the applicability of techniques for formal reasoning about fault tolerance in this application area is investigated within the case study.

The key research tasks in this case study can be summarized as follows:

- Automation of rigorous design flow

- Combining refinement and algorithmic verification techniques in the development of distributed communicating systems and communication protocols

- Automation of data abstractions

- Application of formal reasoning techniques for fault tolerance in the distributed communicating systems and communication protocols domain

2.6. Functional Description of 3GPP Positioning System and PCAP

The *Third Generation Partnership Project (3GPP) Positioning system* provides a positioning service, which calculates the physical location of a *UE (User Equipment)* in a UMTS network. In Figure 2.1 we present the relevant part of a UMTS network for the positioning service. A UMTS network can be divided into *Core Network (CN)* and *Radio Access Network (RAN)*. In Figure 2.1 we have focused on presenting the part of RAN, which contains most of the positioning functionality. The whole CN is presented as a single counterpart module. *Position Calculation Application Part (PCAP)* is the protocol specified for communication between the distributed positioning service parts. PCAP manages the communication on the *Iupc* interface between the *Radio Network Controller (RNC)* and the *Stand-alone Assisted Global Positioning System Serving Mobile Location Center (SAS)* network elements (shaded area in Figure 2.1). RNC manages and controls all communication in the radio access network. Communication is specified for *Iub, Iur and Uu* (i.e. the air interface) interfaces. SAS is a network element dedicated for positioning service and contains, e.g., positioning related algorithms. The reference measurement units (i.e., LMUs) provide information for position calculation. The functional requirements for the RNC-SAS communication have been specified in [1]. The PCAP specification [2] defines the *Iupc* interface and the corresponding signalling procedures.

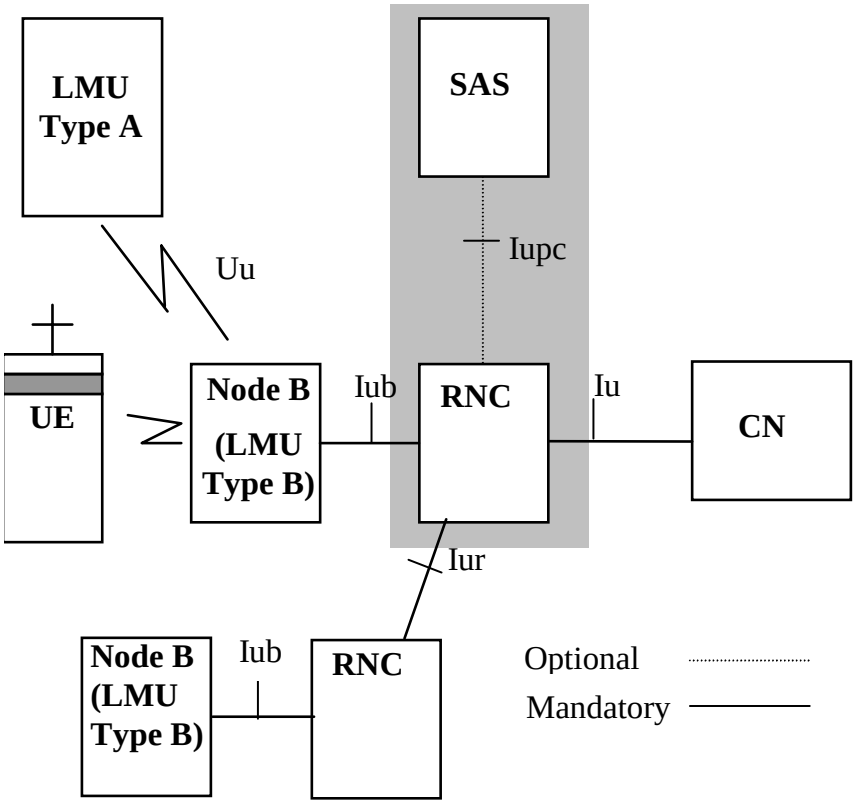


Figure 2.1: UMTS Network Architecture for Positioning

2.7. Evaluation Criteria

To allow fast and efficient knowledge and technology transfer to Nokia, the developed methods are tied to Lyra framework. Industrial applicability of developed methods and RODIN tools will be evaluated according to:

- How well they fit with the existing development framework?
- How much support they provide for more rigorous development process?
    - How many new tasks in the development process can be tackled using the methods developed in RODIN?
- How much support they provide for automation of the development process?
    - How many new tasks in the development process can be tackled using the methods developed in RODIN?

3. Case Study 2: Engine Failure Management System

3.1. Background

Failure management is a safety-critical subsystem of a real-time, embedded controller. The subsystem is concerned with detecting failures of external physical devices and taking appropriate remedial actions. Within the context of the complete system (machinery and controller) the failure management subsystem can be regarded as a component that contributes to a fault tolerant system.

A concern with engine failure management systems is that dynamic characteristics of failure of devices are often not determined until late in development. Consequently changes are costly and there is a need to address modifications and maintenance efficiently. Furthermore a common concern in the avionics domain is in supporting long lifetime products in the face of component obsolescence. This makes portability to new hardware platforms a desirable feature.

3.2. Current State of Work

The case study will not be based on an existing product. Previous applications have been developed by VTEC and this experience will be used to invent an imaginary but realistic example for the case study. The functional requirements specification for the imaginary example will be available publicly.

3.3. Available Material

Since the case study is not based on a particular product, no public material describing the functionality is available. However, it may be possible to estimate comparative measures based on prior experience in this domain. Moreover, the preliminary work on the case study has been reported in [1].

[1] I.Johnson, C.Snook, A.Edmunds, and M.Butler. Rigorous development of reusable, domain-specific components, for complex applications. In Proc. of 3rd International Workshop on Critical Systems Development with UML. Lisbon, Portigal, October 2004.

3.4. Formal Methods Experience in VTEC

VTEC – the company leading the case study has no prior experience of any of the tools or techniques (including formal methods, UML, Eclipse) to be used in the case study. The company has experience of developing safety-critical embedded control products including failure management subsystems.

3.5. Expected Contribution to RODIN

The case study aims to explore the use of UML in combination with the formal method B and thereby provide direction and evaluation for the development of the UML-B based plugin tools within the application domain. The case study will also make use of the ProB plugin tool and the B kernel to provide direction and evaluation to the development of these tools. We aim at assessing the usability of such tools and techniques for novice users and providing feedback for improvement in this respect.

The case study focuses specifically on the development of configurable generic models (involving multiple levels of abstraction) and ways of generating easily verified, specific models from them. In this respect the case study will direct and assess the re-use features provided by the tools.

In addition to its main role as a research driver and evaluator, the relative merits of different mechanisms for the specification of a domain specific language will be investigated. The usefulness of the platform independent, generic and specific models for dealing with portability between hardware platforms will be considered.

3.6. System Functionality

A common functionality required of many systems is to manage failure of its inputs. This is particularly pertinent in aviation applications where availability is critical. The failure management context for an embedded controller is described in Figure 3.1.

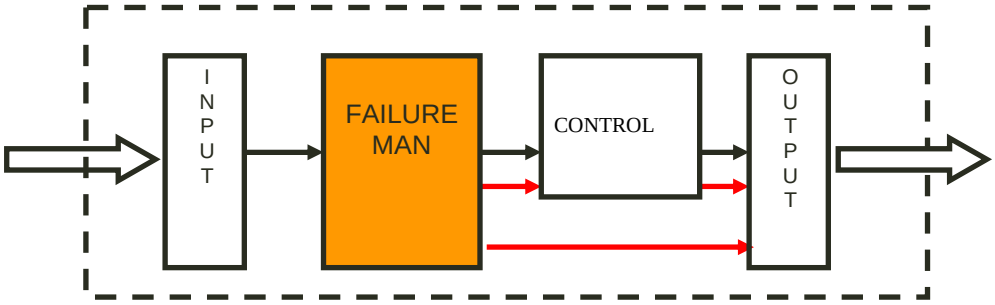


Figure 3.1. Failure Management Context

The diagram illustrates where the failure management subsystem resides in a typical controller. The input signals are validated and if good, are passed unaltered to the control subsystem otherwise the signal is failure managed which may involve substituting values, and taking alternative actions. There are two aspects to the subsystem Failure detection and remedial actions. Typically system components that may have failed resulting in invalid signals would include sensors and actuators. Failure detection involves checks for input validity and typically includes out of range, rate of change and more specialised combinatory checks. Most checking requires persistence of a failed condition before confirmation of a failure. Once a failure is confirmed it cannot be reset. Remedial actions

vary according to input and the state of the system at the time of the failure. Different remedial actions are taken while a failure is being confirmed. Typically an input's value may be frozen temporarily while an input failure is being confirmed. A confirmed failure may result in more permanent actions based on its importance in the control system and may result in graceful degradation or freezing the system in severe cases. Failure management can become functionally complex due to interdependencies between tests dependencies on system operating states.

3.7. Evaluation Criteria

- Evaluate the reduction of cost of late requirement changes
  - Do methods enable early validation of requirements?
  - Is the system design easy to understand?
  - What is the impact of a typical change?

- Evaluate the reduction of cost of development of future products
  - What is the cost of learning the methods?
  - How do costs of using methods compare with existing methods (at each stage of lifecycle)?
  - How effective are reusable components?

- Evaluate the impact on maintaining current quality levels
  - How effective are methods at detecting errors?
  - How effective are the methods at avoiding errors?

- Evaluate whether the improvement of portability has been achieved
  - Do we have a fully functional PIM (Platform independent models)?
  - What is effort to transfer to a PSM (Platform specific models)?

- Evaluate whether the improvement of ease of certification has been achieved
  - How do methods contribute to certification standards?
  - Are the products suitable for independent verification?

4. Case Study 3: Formal Techniques within an MDA Context

4.1. Background

Modern mobile systems have to co-exist with and inside many frameworks that specify the mobile world. These frameworks address a large number of issues such as:

- End-to-end connectivity
- Product architecture
- Security
- Fail-safe functionality and reliability
- Applications
- User interface

When developing new systems (from individual software components to additional high-level frameworks) it is necessary to ensure the consistency and validity of these against what already exists. We must investigate whether a new item is a refinement, development, extension or even an entity conflicting with existing work. For example, the current security frameworks may be compromised by new applications/architectures etc - it is necessary (and critical) to understand how the security might be compromised and what the scope such an effect might be.

As one can easily see the amount of complexity in coordinating these frameworks, their interactions and developments are immense. It is understanding how formal methods can be applied in this area that is critical to the coherent and correct use of such frameworks.

4.2. Current State of Work

The case study here is ambitious but parts of these frameworks are being developed inside Nokia separately from this, while we concentrate on a certain sub-set of frameworks, namely: a subset of MITA, security and a simple application.

We envisage two distinct parts to this case study: modelling itself and the management of those models produced. These are explained in more detail in the forthcoming sections.

4.3. Available Material

The available material for this case study is presented as a list of references at the end of this section.

4.4. Experience with Formal Methods

Nokia is a large company with very wide needs and experiences. In this respect it is impractical to provide an ideal of overall general competence. Nokia does have well

defined processes for software and hardware construction and much experience related with methods and tools for constructing and testing of such systems.

However, the explicit use of formal methods is not practiced widely and so knowledge of the use and applications of tools, languages etc, such as B, Z, VDM etc. is less known. We have a number of programmers currently either investigating or transferring technologies related to formal development within the company's business units. The main focus of much of this work is around the correct usage of UML/OO and techniques such as design-by-contract.

This case study is being made within the Nokia Research Center which does have a deep knowledge of such techniques and whose job it is to develop and transfer such techniques into the company best practice as a whole. In addition there is also involvement persons outside Nokia Research who are responsible for the various product developments in Nokia and of the requirements of the case study material.

4.5. Expected Contribution to RODIN

We see the tools and techniques constructed as part of RODIN to be components in a much larger overall methodology framework.

We believe that construction of a methodology should be made from a set of parts - rather than adopting a single, monolithic method - and that this construction is dictated by the availability of testing technologies (e.g., theorem proving) and the overall requirements of systems to be built using that methodology (e.g., safety critical, real-time).

This case study aims to investigate how these technologies can be integrated into existing methodologies and the effects they have on the development of systems constructed from these methodologies.

We also investigate how object-orientation and related development technologies (e.g., UML, MDA) affect the choice of formal development methods and the complexity/usability of the development and analysis.

4.6. Case Study Description

We propose to investigate a number of layers of frameworks of varying degrees of completeness and formality. These are driven from three distinct viewpoints described below and pictured in figure 4.1.

• The Mobile Technical Internet Architecture

  MITA describes a vision and potential architecture of the whole end-to-end mobile environment at a very high level of detail.

• "Second Level Frameworks"

These are frameworks for particular aspects, e.g., security, persistence, protocols, which are specified at a much greater level of detail. These frameworks must conform to (though not necessary 100%) with the MITA framework. In the cases where these frameworks do not conform it is critical to understand where the conflicts exist and what effect they might have upon the system as a whole.

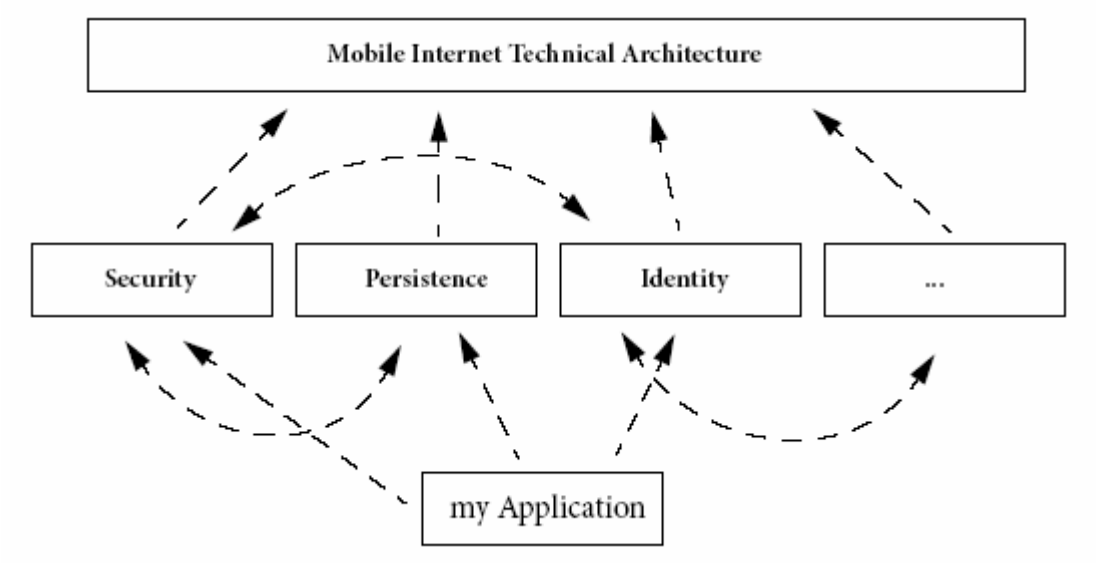


Figure 4.1. Relationship between frameworks

• Applications

These are individual applications that instantiate the various frameworks. For example, a simple application that talks with some server might use connectivity frameworks (e.g., sockets, SMS, GPRS, SIP, UMTS), security frameworks (eg: Liberty), data formats (e.g., XML, Webservices) as well as their own application and implementation frameworks (e.g., J2ME, Symbian etc).

It is in particular in the area of the "second level frameworks" and of a security framework that we are most interested in this case study. The MITA level is an informal structuring layer that we must conform to. A simple application will be constructed outside of this particular project for testing purposes.

4.6.1. Modelling and transformation

Whatever is being produced, one is required to model one's wishes. Regarding the modelling, we will concentrate on utilising the UML (OO semantics) as the base language for the models. Ideally we wish to use the OCL and an Action Language (based upon the UML Action Semantics) but tool support for OCL is lacking and action languages not yet taken into any widespread use. An alternative approach is to map OCL

into a more readily available language such as Alloy or B, or, utilise these languages within the UML as can be seen with the U2B tool.

Model Transformation is an integral part of the OMG's MDA idea. However, models are not translated on a syntactic basis alone (e.g., UML classes = Java or C++ classes) but are transformed via some platform (or architecture) such that the overall semantics of the source models are preserved in the target models.

In this case study we will construct a number of models, in particular the following:

• MITA E2E "domain" model

This describes the overall end-to-end structure using the concepts and relationships defined by the Mobile Internet Technical Architecture.

• Security Framework "domain" model

This describes the concepts and relationships required to capture the various security needs (e.g., user identification, session, keys etc.). This model can be treated later as a framework/platform/architecture or even language.

• UMTS Security Architecture model

The Universal Mobile Telecommunications System (3G) defines various security concepts and protocols. Some concepts in the security framework may be mappable or be mapped to the concepts at this layer.

• UE (User Equipment) Application and Architectural models

In order to validate a framework it is necessary to construct a test application. These models depict this application. In effect this might be some kind of simple users application that runs on some mobile device. This application then has to utilise the security framework and be mapped onto various platforms, e.g., J2ME v2, UMTS etc.

4.6.2. Model management

Any "model driven" approach requires the management of the models within the chosen methodology for that particular development. We will utilise here the basic OMG Model Driven Architecture (MDA) ideas for the development flow and relationship between models. We have already developed a model of this development flow which is conformant to the MDA.

Given the number of models we have and the development process we have a number of issues should be investigated. Namely,

• Generic Model Management

How does one relate models together in terms of composition, decomposition and extraction of information? How does the development process manifest itself here in terms of the structure of models produced?

• Transformations

How does one construct and utilise transformations between models in terms of semantic mappings? How does one preserve the traceability information across models? What is the granularity of transformation, i.e., is adding a class considered a transformation or does the term specifically relate to large "macro" changes across platforms?

• Formal Relationships
Refinement is the major driver here such that ideally any pair of models across some development step should preserve refinement. However the notion of refinement becomes more complex when one takes into consideration multiple aspects, some of which do not refine in the naive sense. In addition the ideas of retrenchment allow weakening under some circumstances, can this be easily integrated into this approach?

4.7. Evaluation Criteria

The case study will be judged upon the following criteria:

• How well does this formal approach fit with existing processes?
• How well do these techniques integrate with an object-oriented approach?
• In what form are these technologies transferred to the Nokia Business Units?
• Can this approach check components against a (very loose) specifications?
• What is required to understand over constrained models, their causes and effects?
• Can this approach deal with requirements volatility?

Particularly important is the amount of technology transfer that is made into our current development practices. Measuring such transfer, especially when it involves ideas rather than concrete tools, methods or processes is difficult and real metrics for the quantification of this do not exist.

References

[1] Heikki Kaaranen, Ari Ahtiainen, Lauri Laitinen, Samak Haghian, Valtteri Niemi (2001). UMTS Networks - Architecture, Mobility and Services. Wiley.
[2] Sergey Melnik (2004). Generic Model Management, Concepts and Algorithms. LNCS 2967. Springer.
[3] Ian Oliver (2004). Model Based Development and Embedded System Design. Session on Model Driven Architecture. Forum on Design Languages 2004. Lille, France.
[4] Jurgen Ziegler (2003). End-to-End Concepts Reference Model. Nokia.
[5] Jurgen Zielger (2004). An End-to-End Model for Mobile Systems. Nokia.
[6] MITA (2002). Mobile Internet Technical Architecture. IT Press.

[7] UML 2.0 Superstructure, Object Management Group.
[8] UML 2.0 Infrastructure, Object Management Group.
[9] Model Driven Architecture, http:/www.omg.org/mda, Object Management Group.

5. Case Study 4: CDIS Air Traffic Control Display System

5.1 Background

The domain of this study is the specification and design of high-integrity software systems. The particular area of this study is civil aviation information systems.

The original CDIS was a display information system supporting a new terminal control room at the London Area and Terminal Control Centre in the early 1990s. It was a distributed real-time information system for air traffic controllers. CDIS was developed using advanced software engineering techniques including the extensive use of formal methods for specification and design. Praxis (now Praxis High Integrity Systems) developed CDIS for the Civil Aviation Authority (now National Air Traffic Services).

The CDIS development produced a number of formal specifications, including a 1000+ page main specification written in the VVSL variant of VDM-SL (a model-based specification language). The key concurrent aspects of the system are captured in a separate specification written in CSP (a process algebra).

There are a number of publications on the development of the original CDIS software. The primary reference is "Using Formal Methods to Develop an ATC Information System" by Anthony Hall; IEEE Software, March 1996.

5.2. Available Material

The case study will be based on a subset of between 20% and 30% of the original CDIS specification. This subset will incorporate the key parts of the original system specification, including the comments attached to the various modules and operations. Initial documents will be available only in the VVSL dialect of VDM-SL; both PDF and TeX formats can be provided.

Supporting data that is available for the project includes:

- design, development and testing effort metrics;
- the detailed design that resulted from the specifications (original documents held by Praxis; contents available to partners in summary form); and
- system requirements documents.

Note that the IP rights to the original CDIS specification are held by National Air Traffic Services (NATS) who will have to approve the subset document for dissemination among the RODIN partners.

5.3. Current State of Work

The original specification and design documents have been retrieved and we have validated our ability to regenerate them from source. The subset is currently being selected and extracted by the CDIS Technical Authority, Anthony Hall.

5.4. Formal Methods Experience

Praxis HIS is an experienced industrial practitioner of formal methods. The company has developed a number of high-integrity industrial systems using rigorous proof and analysis techniques, including:

- the CDIS air traffic information system;
- the SHOLIS naval helicopter landing system; and
- the MULTOS smart card certification authority.

Praxis develops and markets the SPARK Examiner static analysis tool and associated SPADE proof tools. The Praxis staff has a wide range of publications in various areas of formal methods in software and systems, and regularly present keynote speeches at major formal methods conferences.

5.5. Contribution to RODIN

The case study will provide 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 will be the first test of the RODIN tool platform, as it will highlight any scalability issues that the platform might have. Once the specification can be accepted by the RODIN tool, the secondary test will be the degree of analysis that is possible for the specification.

One of the key tests for the tool platform will be the degree to which the tool supports the refinement of the specification to a detailed design. The CDIS design is concurrent and heterogeneous, with a number of different classes of workstation and system support devices. The model-based specification is silent about concurrency, and during the original development the concurrency aspects were introduced during manual refinement. If the tool can facilitate the introduction of concurrency then it will represent a major advance in the system development process.

5.6. Description of Functionality

The CDIS case study is an air traffic information system. CDIS receives information in real time from the National Airspace System and from the Airport Display Information System at the major airports. In addition, CDIS can send information to the closed circuit television system. CDIS also has its own store of information produced and edited within CDIS itself using administrative workstations provided with a graphical and text editor. CDIS is responsible for displaying to the controllers information about arriving and departing flights, weather conditions and equipment status at airports and other support information provided by CDIS data entry staff. It also maintains real-time displays of its own status to allow the engineers to control the system.

The original system included fault tolerance via dual networks, dualled hardware and health monitoring equipment. It had a basic operational requirement of being available

99.97% of the time. The precise scope of the specification that will be recreated by this case study will depend on the final contents of the subset.

5.7. Evaluation Criteria

- How well has the specification complexity and size challenged the space and time limitations of the RODIN tools?
- How many errors and inconsistencies in the original specification have been identified?
- How much of the concurrent design of the system has emerged from the specification-to-design refinement, and how much was this supported by the tool?
- How much more quickly was the design done compared to the known design time for the original CDIS development?
- How much additional verification has been made possible by the tool?
- Comment on the expressive power of the formal notations relative to the specification

6. Case Study 5: Ambient Campus

6.1. Background

The most general version of the objective is to study the fault-tolerant behaviour of software which could enhance the educational process. From the point of view of the project, we seek to "stress" the RODIN (evolving) method(s) and tools in the area of Ambient Computing. We proposed in the RODIN project DoW to look at this difficult area with the use of Wireless linked devices in a university campus.

6.2. Current State of Work

Our current work focuses on defining the specific Ambient Campus applications and scenarios to be developed, on planning the steps of our experimental work, which will possibly involve groups of lecturers and students in the School of Computing Science over the 3 year period, and on initial experiments with the equipment to be used (such as wireless connections available at the university, PDAs coming from different providers: HP, Dell, etc.) and with the available underlying middleware, on top of which we will build the Ambient Campus applications.

A number of relevant applications have been analysed to allow us to finalise a set of appropriate RODIN scenarios. This includes Active Campus and Active Class (http://activecampus.ucsd.edu/), the Draiocht Project (http://www.dsg.cs.tcd.ie/), Cool Campus (http://infotech.monash.edu/coolcampus/index.html) and Wireless Campus (http://www.soft.uni-linz.ac.at/Research/Projects/Wireless_Campus/index.php).

We have recently identified Linda-based mobile systems as the most relevant communication paradigm for developing these applications and have extended the well-known Lime middleware [1] with advanced mechanisms that allow asynchronous loosely-coupled handling of abnormal situations [2]. We are now working on introducing error confinement scopes into this middleware and on developing a basic set of abstractions (such as location, cope, agent, interface) to be used in designing the Ambient Campus applications.

We intend to evolve the formal specification of the applications over the next six months. This process will exercise the state-based and process-algebraic specification formalisms.

6.3. Available Material

There is no material currently fixed for the specification of this case study. One of the things we wish to investigate is the utility of RODIN notation(s) in pinning down our understanding of the system (and in particular the faults which will be tolerated).

We are investigating work of others in the areas of

- computer-aided teaching environments

- wireless support for PDAs
- virtual reality (for teaching difficult concepts).

[1] A. L. Murphy, G. P. Picco, G.C. Roman. “Lime: A Middleware for Physical and Logical Mobility”. Proc of the 21st International Conference on Distributed Computing Systems (ICDCS-21), Phoenix, AZ, USA, April 16-19 2001, pp. 524-233.
[2] A. Iliasov, A. Romanovsky. “Exception Handling in Coordination-based Mobile Environments”. November 2004. To be submitted to COORDINATION 2005.

6.4. Formal Methods and Fault Tolerance Experience

The team at Newcastle have expertise in a range of “formal methods”. Some of the communication problems might well be studied using process algebras and both Prof Koutny and Jones have worked on such formalisms. The knowledge of both process algebras and model checking via unfolding Petri-nets is a key expertise of Dr Khomenko and Prof Koutny. The more general task of scoping the problem is likely to be aided by state-based specification techniques: there is considerable expertise on VDM in Newcastle and three members of the team are to attend Abrial’s course on “Event-B” in December 2004.

The team at Newcastle have been involved in developing a wide range of fault tolerance techniques (Dr Romanovsky). Mr Iliasov has recently gained considerable experience in developing fault tolerant systems which can parallelise their execution over a number of computers including the portable ones. Mr Coleman has been doing relevant work on the formalisation of fault tolerance constructs, including the way they are manifested in BPEL-style languages.

6.5. Contribution to RODIN

Our ideas have evolved in discussions since the original proposal was written. In particular we have extended the scope of the proposed project to cover two key facets of fault tolerance.

For all case studies (or at least in their totality), the project wishes to check that the yet-to-be-finalised RODIN development methods are reasonably general; similarly, we want to check that there are tools to support most difficult parts of the formal design process for fault-tolerant systems. Wireless connection itself is almost bound to provide faults which have to be tolerated. Besides, the applications of this type will inevitably require dealing with a variety of abnormal and non-anticipated events due to the system openness, mobility of its participants and their dynamic nature.

It is clearly beyond the scope of RODIN to investigate the whole area of “Ambient Computing” but we felt that support for some aspect of teaching/learning would stress our work in this important arena.

As indicated above, the main reason for this case study is to stress the evolving RODIN ideas and tools in an area not covered by other case studies. We will deliberately push the case study into ambient computing problems.

6.6. Description of Functionality

The overall focus will be on developing a number of scenarios in which students, lecturers and other university staff members are involved in activities of several types (a lecture, a consultation, a group project, work in the library, etc.) spread over some period of time. The actors involved will coordinate their work using PDAs. This work will consist of steps in which mobile participants cooperate remotely and asynchronously, and those in which they come together to one location for face-to-face discussions (such as lectures). They will use available university desktop PCs where appropriate, but outside university PC pools they will be able to use the PDAs only because in the scenarios we envisage the use of mobile phones will be prohibited (e.g. in lecture rooms, the library, other university buildings).

Each type of activities will be structured as a number of smaller standard applications in which students, lecturers and staff members play specific roles, for example, a group meeting, questions & answers session after a lecture, the integration and testing of a system developed by a student group, etc. We have narrowed our work down to four possible scenarios.

- The first scenario is a support for students in a classroom (interactive lecture) situation, including downloading lecture material and uploading answers to exercises. This work will allow us to develop features for coping with users (students) coming into and out of the wireless field, for isolating information coming to and from individual students, for anonymous and asynchronous communication between students and lecturers.
- The follow-up scenario is a student support for programming exercises to help students both when they have difficulties understanding something and in submitting solutions for evaluation. Some of this student activity will be done in a geographically and time-concentrated way in a “laboratory”; some of it will be done away from a lab (possibly in the middle of the night in the student’s own room). Even in the former case, students will come and go; in the latter the most that can be provided is to queue requests for the next available tutor or run a “solution” against (undisclosed) test vectors.
- The third scenario is a system supporting joint work of postgraduate students on group software projects. This scenario will require the use of both desktop PCs and PDAs. There will be a need to provide asynchronous loosely-coupled communication patterns supporting coordination of students’ work on project tasks and joint work on documents as well as of a much more synchronised activity involving, for example, integration testing. A special role will be played by the project manager monitoring all this work and by a lecturer helping students when necessary.
- The fourth scenario is a support of the university library users. The idea is that every user entering the university library will get a PDA assisting him/her in his/her work. This will include navigation of the premises, search for books, checking the user loan

account, remote consultations with librarians, making notes and copying texts from books, arranging meetings to work in the library on joint presentations/papers.

It is our intention to finalise our choices within one month.

6.7. Evaluation Criteria

- Given the four AmI scenarios developed previously by the ISTAG group, analyse how much has the use of formal techniques improved the original ideas about this application?
- How clearly can we show that we have introduced dependability and fault tolerance?
- How well has the crystallisation of ideas via the formal specification improved early discussion of the system requirements, especially whether the requirements are OK?
- Evaluate the usefulness of the applications by using questionnaires
- Compare student performance using the parallel groups which work is organised with and without ambient campus systems"

7. Generic Case Study Measuring Criteria

- How much effect have the new tools had on the case study?
- How hard is the case study work to do before the tools arrive?
- How hard it is to learn to use the tools in the case study? (shape of learning curve)
- What is it like when you stop using the tool, once you're used to it?
- 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
- Evaluate expressive convenience of the formal notations used